ERROR RESILIENT VIDEO STREAMING ALGORITHMS BASED ON WYNER-ZIV CODING

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Liang Liang

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2009

Purdue University

West Lafayette, Indiana

To my grandmother, Xuerong Zhang

To my parents, Xingchao Liang and Xiaoyu Wang

In memory of my grandfather, Dazhang Liang

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Edward J. Delp, for his guidance and support along the way of my Ph.D study. I am very grateful to my advisor for giving me this valuable opportunity to study and research in the video processing area. I want to express my sincere thanks to his inspiring instruction, his confidence in me finishing my study. I am very honored to be his student and a member of the image and video processing lab (VIPER).

I would also like to thank my co-advisor Professor Paul Salama, for his insightful suggestions and advice along my way of conducting research in Wyner-Ziv based error resilience.

I would like to thank my other two committee members: Professor Mary Comer and Professor Zygmunt Pizlo for their advice and encouragement.

I would like to thank the Indiana Twenty-First Century Research and Technology Fund for supporting the fund for conducting this research on error resilient video streaming techniques.

I would like to thank Dr. Rita Saerens, for giving me an opportunity to work as a teaching assistant in the department of mathematics when the Indiana Twenty-First Century Research and Technology Fund was ended. I was very much enjoined teaching math class and at the mean time the assistantship supported my continue study in error resilience techniques.

I would like to thank all my friends and colleagues in VIPER. They have been a great support and encouragement to me during these years. I would like to thank Limin Liu, for her patience and valuable suggestions whenever I came with questions to ask. Her expertises in the area of Wyner-Ziv coding has benefited me a lot. Discussing with her was always very insightful. I want to express my sincere thanks to her for all her help. I would also like to thank Fengqing Maggie Zhu, Ying Chen,

Golnaz Abdollahian, Ka Ki Ng, for being great friends to me. They have been my great support and make my life in VIPER much happier. I would like to thank Dr. Hyung Cook Kim, for his help when I run into questions with unix. I especially want to thank him for having referred me to Qualcomm camera team. I would also like to thank my other lab mates for their great help during my study. They are: Dr. Eugene Lin, Dr. Yuxin Liu, Dr. Hwayoung Um, Marc Bosch, Oriol Guitart, Deen King-Smith, Ashok Mariappan, Anthony Martone, Aravind Mikkilineni, Nitin Khanna and Carlos Wang.

I would like to thank Dr. Piu Ong for offering me the summer intern opportunity in Cisco, 2007. This intern experience had been very beneficial to me for connecting my research work with the real products. I would like to thank my mentor Norman Cheng for his guidance in my intern project on video conferencing, where I first time utilized my knowledge in video for real products' design. I would like to thank Dr. Rui Zhang and Dr. Jim Chou for their insightful mentoring on my second summer internship in Cisco, 2008. By working on the project of scalable video coding, my experience on video coding had been widely expanded.

I would like to thank Stefanos Orfanos, who has given me great support and love during my study. I am grateful to his encouragement when I was struggling in my research work.

I am also thankful to my friends: Ligia-Varinia Barreto, Renate Mallus, Silvina Meza, Michelle Murphy, Julian Simon, Edie Cassell, Jing Cheng, Evie, Josh, Xiaoju Zheng, Maria Coburn, Joanne Webb, Jinxia Liu, , Ricardo, Sidharth, Ivan, Ling Huang, Elena, Cecilia, Yajie Zhang, Xiaojun Feng and my roommates mother Haiyan Liu, Mu Qiao, Fei Wang, Chang Liu, Zhenqiu Xie, Fang Wan, Beini Lin, Peng Gong, Chao Yin, Jiefu Chang, Haiyang Qian, Ali Yanic, Yihuang Shen, Nannan Sun, Peishan Liu, Yuerong Hu, Jemmie, Fenglun Pan, Peichun Chen, Yili Zheng, Zhi Qi, Xing Fang, , Zhengzheng Pan, Hong Huang, Hong Li, Haorong Li, Jianqi Wang and Chang Liu for their great friendship.

I have dedicated this document to my parents Xingchao Liang and Xiaoyu Wang, and my grandparents Dazhang Liang and Xuerong Zhang. Their love and encouragement have supported me throughout these years while I was pursuing my academic degrees at Purdue University. I thank my parents for giving me the chance to come to this world.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABBREVIATIONS

UEP	Unequal Error Protection
CAUEP	Content Adaptive Unequal Error Protection
FBUEP	Feedback Aided Unequal Error Protection
TE	Turbo Encoder
TD	Turbo Decoder
R-S	Reed-Solomon Coder
EEP	Equal Error Protection
SLEP	Systematic Lossy Forward Error Protection
MB	Macroblock
FEP	Forward Error Protection

ABSTRACT

Liang, Liang. Ph.D., Purdue University, August, 2009. Error Resilient Video Streaming Algorithms Based on Wyner-Ziv Coding. Major Professor: Edward J. Delp.

Compressed video is very sensitive to channel errors. A few bit losses can derail the entire decoding process. Therefore, protecting compressed video is always necessary for reliable visual communications. In recent years, WynerZiv lossy coding has been applied to error resilience and achieved superior improvement over conventional techniques.

In our work, we first proposed an unequal error protection method for protecting data elements in a video stream, via a Wyner-Ziv encoder that consists of a coarse quantizer and a Turbo coder based lossless Slepian-Wolf encoder. Data elements that significantly impact the visual quality of decoded video, such as modes and motion vectors as used by H.264, are protected more by assigning a higher parity data rate than the coarsely quantized transform coefficients. This results in an improvement in the quality of the decoded video when the transmitted data was corrupted by the transmission errors, than the results obtained by using equal error protection.

In our later work, we described an improved technique by adapting the parity bit rates of the protected video information to the video content. The parity data rates are assigned to the motion information and the transform coefficients according to their impact on the visual quality of each frame. This results in an efficient way of improving the quality of the decoded video when it has been corrupted by transmission errors.

We also proposed a feedback aided error resilience technique, based on Wyner-Ziv coding. By utilizing feedback regarding current channel packet-loss rates, turbo coders can adaptively adjust the amount of parity bits needed for correcting corrupted

slices at the decoder. This results in an efficient usage of the bit budget for Wyner–Ziv coding while maintaining good quality decoded video when the data has been corrupted by transmission errors.

Finally we studied the case of combining the content adaptive function at the encoder side and utilizing the feedback of the current channel packet losses conveyed from the decoder side to adjust the protection level of the protection video information at the same time. A new parity data rates assignment table is given. The experiment results showed the improvement on the rate distortion performance of this combined method, comparing to the case of utilizing the content adaptive function or the decoder side feedback information individually. An analysis and modeling of the system have also been studied, which aim to find the theoretical basis for finding the optimal data rate allocation between the primary coding and the Wyner-Ziv coding.

1. INTRODUCTION

Channel errors can result in serious loss of decoded video quality. Many error resilience and concealment schemes have been proposed [1]. However, when large errors occur, most of the proposed techniques are not sufficient enough to recover the loss. In recent years, error resilience approaches employing Wyner–Ziv lossy coding theory [2] have been developed and have resulted in improvement in the visual quality of the decoded frames [3]- [13]. Other works applied distributed source coding onto error resilience include [14], [15], [16], [17].

In 1976, Wyner and Ziv proved that when the side information is only known to the decoder, the minimum required source coding rate will be greater or equal to the rate when the side information is available at both encoder and decoder (see Figure (1.1)). Denoting the source data by X and the side information by Y, where X and Y are correlated, but the side information Y is only available at the decoder, the decoder manages to reconstruct a version of X, X', subject to the constraint that at most a distortion D is incurred. It was shown that $R^{WZ}(D) \geq R_{X|Y}(D)$ [2], where $R^{WZ}(D)$ is the data rate used when the side information is only available to the decoder and $R_{X|Y}(D)$ represents the data rate required when the side information is available at both the encoder and the decoder.

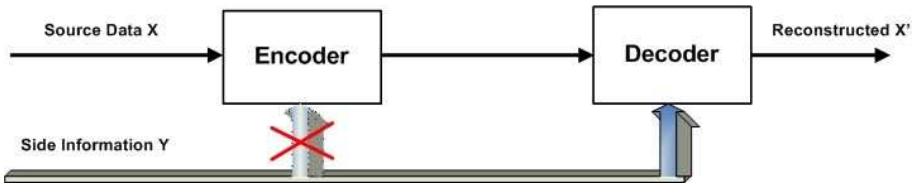

Fig. 1.1. Side Information Available at Decoder Only

Wyner and Ziv also proved that equality can be achieved when X is Gaussian memoryless source and D is mean square error distortion $D(X, X') = (X - X')^2$, as well as when the source data is the sum of an arbitrarily distributed side information Y and independent Gaussian noise U. In addition, they derived the rate boundary $R^{WZ} = R_{X|Y}(D) = \frac{1}{2}\log\frac{\sigma_U^2\sigma_X^2}{(\sigma_U^2+\sigma_X^2)d}$ that can be achieved when $0 < D < \frac{\sigma_U^2\sigma_X^2}{\sigma_U^2+\sigma_X^2}$, and where σ_U^2 and σ_X^2 are the variances of the Gaussian noise U and the source data X [2].

One of the earliest work of applying Wyner-Ziv lossy coding theory for error resilient video transmission is proposed in [3], 2003. The general approach is to use an independent Wyner–Ziv codec (as shown in Figure (1.3)) to protect a coarse-version of the input video sequence, which can be decoded together with the side information from the primary MPEG-x/H.26x decoder. The basic system structure is shown in Figure (1.2). The approach proposed in [3] is known as systematic lossy forward error protection (SLEP).

SLEP, in addition to an MPEG-2 encoder, uses a Wyner–Ziv encoder made up of a coarse quantizer and a lossless Slepian–Wolf encoder that utilizes Turbo coding. The input to the Wyner–Ziv encoder consists of the reconstructed frames obtained from the MPEG-2 encoder. These are initially coarsely quantized and then passed onto a Turbo encoder [18], [19], which outputs selected parity bits. At the receiving end, a Turbo decoder uses the output of the MPEG-2 decoder, as side information, and the received parity bits to recover the lost video data. In the absence of any channel errors, the output of the SLEP decoder will be the same as that of the MPEG-2 decoder. If however, channel errors corrupt the MPEG-2 stream, then SLEP attempts to reconstruct a coarse version of the MPEG-2 stream via the received parity bits, which may have also been corrupted. The quality of the reconstructed version depends on the quantization step used by the coarse quantizer as well as the strength of the Turbo code.

Improvements to SLEP have been proposed in [9], [12], and have resulted in a lower data rate for Wyner–Ziv coding as well as improved decoded video quality. It is noted that the SLEP method has been applied to H.264 in [12].

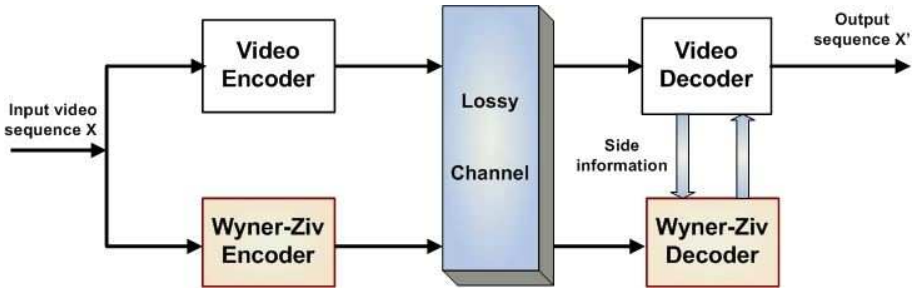

Fig. 1.2. Error Resilient Video Streaming Using Wyner–Ziv Coding.

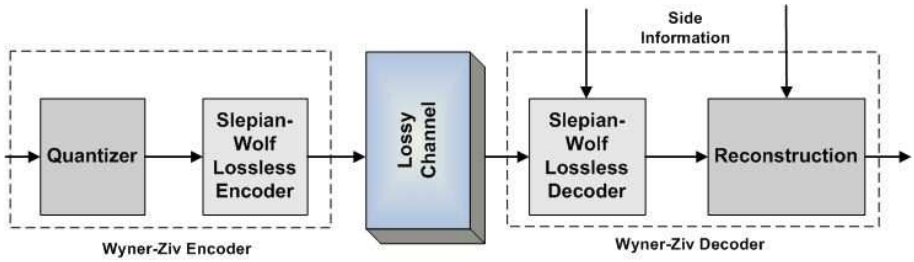

Fig. 1.3. Wyner–Ziv Codec

Another approach of using Wyner–Ziv coding for robust video transmission was proposed in [20], in which the Wyner–Ziv encoder consisted of a discrete cosine transform, a scalar quantizer and an irregular repeat accumulate code as the Slepian–Wolf coder.

Our approach to unequal error protection is also based on Wyner–Ziv coding and is motivated by the SLEP approach. The overall goal of our schemes is to correct errors in each frame by protecting motion information and the transform coefficients. The primary codec is an H.264/AVC codec and the Wyner–Ziv codec utilizes coarse quantization and a Turbo codec. Instead of protecting everything associated with the coarsely reconstructed frames, we separately protect motion information, and transform coefficients produced by the primary H.264 encoder. The idea being that since the loss of motion information impacts the quality of decoded video differently from the loss of transform coefficients, both should receive unequal levels of protection that are commensurate with their respective contributions to the quality of the video reconstructed by the decoder [21]. The motion information is protected via Turbo coding whereas the transform coefficients are protected via Wyner–Ziv coding. This approach is referred to as unequal error protection using Pseudo Wyner–Ziv (UEPWZ) coding.

We improve the performance of our unequal error protection technique by adapting the parity data rates for protecting the video information to the content of each frame. This is referred to as content adaptive unequal error protection (CAUEP) [22]. In this scheme, a content adaptive function was used to evaluate the normalized sum of the absolute difference (SAD) between the reconstructed frames and the predicted frames. Depending on pre-selected thresholds, the parity data rates assigned to the motion information and the transform coefficients were varied for each frame. This resulted in a more effective and flexible error resilience technique that had an improved performance compared to the original UEPWZ.

Another approach to improve the proposed unequal error protection is to send feedback regarding the current channel packet loss rates to the Pseudo Wyner–Ziv

encoder, in order to correspondingly adjust the amount of parity bits needed for correcting the corrupted slices at the decoder [23]. This approach is referred to as feedback aided unequal error protection (FBUEP). At the decoder, the current packet loss rate is estimated based on the received data and sent back to the Pseudo Wyner–Ziv encoder via the real-time transport control protocol (RTCP) feedback mechanism. This information is utilized by the Turbo encoders to update the parity data rates of the motion information and the transform coefficients, which are still protected independently. At the Wyner–Ziv decoder, the received parity bits together with the side information from the primary decoder are used to decode and restore corrupted slices. These in turn are sent back to the primary decoder to replace their corrupted counterparts. It is to be noted that simply increasing the parity bits when the packet loss rate increases is not applicable, since it will exacerbate network congestion [24]. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases, the primary data transmission rate should be lowered in order to spare more bits for parity bits transmission.

In out latest work [25], we propose to combine the use of a content adaptive function, implemented at the encoder, with the channel loss feedback provided by the decoder. The new method is named FBCAUEP. In FBCAUEP, both the content of the video sequence and the latest channel packet loss rate are known by the encoder, therefore, the Turbo encoder can do a more precise parity data rate allocation decision on transform coefficients and the motion information. The data rate ratio between H.264 primary data transmission and Wyner-Ziv parity bits transmission can reach $10:1$ to $11:1$ due to the efficient parity data rate allocation with assistance from both the video content and the channel packet loss rate. The experiment results also demonstrate that the rate distortion performance and the visual quality of the decode videos can both be further improved compared to those from CAUEP and FBUEP.

Our proposed error resilience schemes [21], [22], [23], [25], [26] aim to improve both the rate distortion performance as well as the visual quality of the decoded video frames when video has been streamed over data networks such as wireless

networks that experience high packet losses. In our experiments, we only consider packet erasures whether due to network congestion or uncorrected bit errors. The main focus of our scheme is for applications such as video conferencing, especially in a wireless network scenario, where serious packet losses will result in unpleasant distortion during real time video streaming.

1.1 Overview of H.264/AVC Video Standard

H.264/AVC(advanced video coding) is the latest video standard for the coded representation of visual information. It outperforms the earlier MPEG-4 and H.263 standards by providing higher compression of the video images [27].

H.264/AVC defines the syntax of an encoded video stream rather than explicitly defining a Codec. It also defines the decoding methods of the bit stream. A set of practical system structures of the encoder and the decoder are shown in Figure 1.4 and Figure 1.5 [28]. H.264/AVC includes the basic functionality of motion compensated estimation, transform, quantization, entropy coding as in previous MPEG and H.263 standards, but the details of each function block have been modified to achieve a higher compression gain. The main features of the H.264/AVC in achieving significantly coding efficiencies include multi-frame motion prediction, multiple macroblock partition size choices for motion estimation, generalized B frame concepts, quarter pixel motion estimation resolution, multiple choices for intra coding modes, deblocking filter and efficient entropy coding methods.

H.264/AVC includes a forward encoding path and a backward reconstruction path. The input video sequence is processed in units of macroblocks (a block of 16 pixels). On the forward encoding process, motion estimation is first applied to each marcoblock to evaluate the best encoding mode (intra or inter, sub-block sizes) based on a rate distortion algorithm. The mode achieves the lowest rate distortion is chosen as the best mode for the macroblock. After the best mode has been chosen, the difference between the predicted blocks and the reference blocks are computed to form

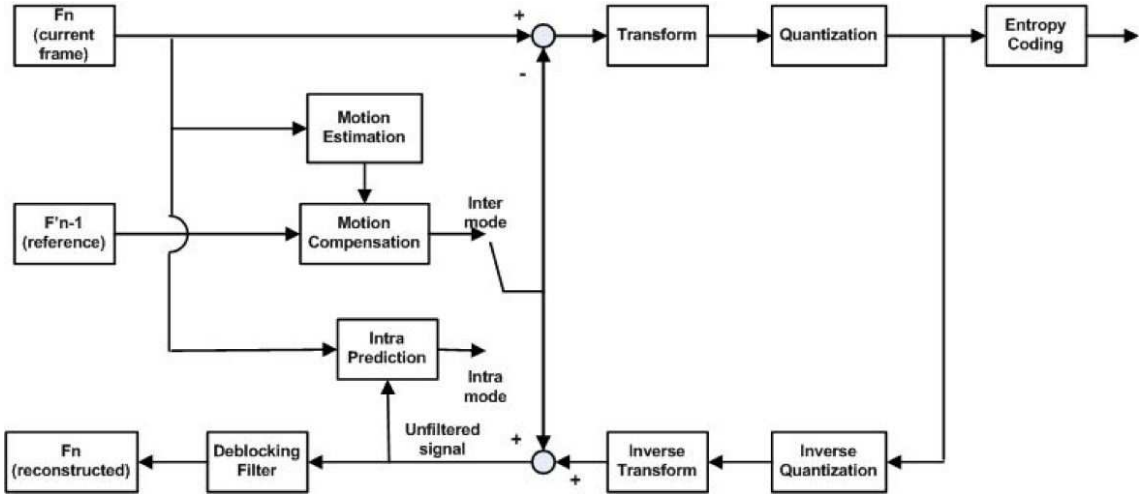

Fig. 1.4. System Structure of a Practical H.264 Encoder

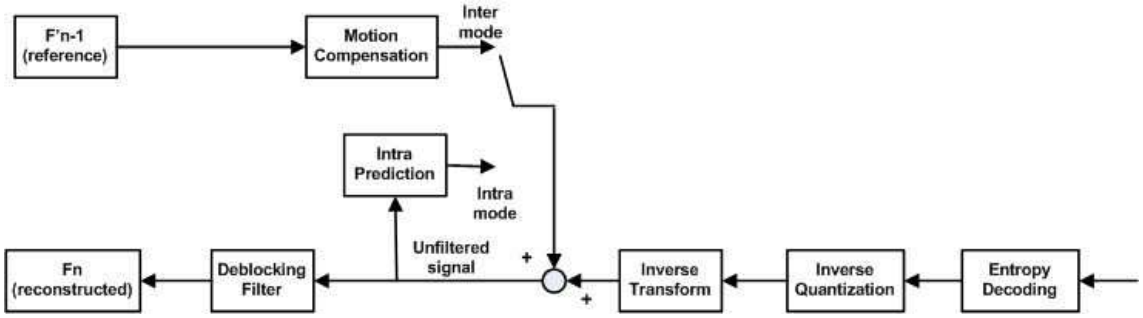

Fig. 1.5. System Structure of H.264 Decoder

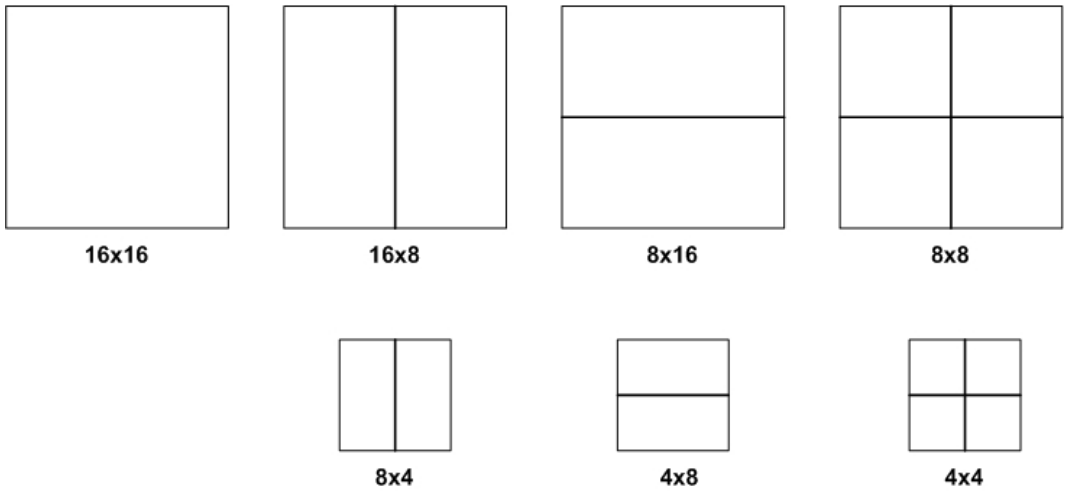

Fig. 1.6. Seven Macroblock partition Modes of H.264 Inter Prediction

the prediction error blocks which are then transformed and quantized. Different from previous video standards, H.264/AVC applies 4×4 integer transform, which can be carried out using integer arithmetic without loss of decoding accuracy. The quantized transform coefficients are reordered and entropy coded.

At the decoder side, the compressed data stream is entropy decoded, inverse quantized and inverse transformed, together with the decoded prediction block, to form the decoded frames.

1.1.1 Inter Prediction

H.264/AVC includes total of 7 macroblock partition modes: 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4, as shown in Figure 1.6. The data rate and the caused distortion of each mode is evaluated and compared. The one results in the lowest rate-distortion (evaluated in Equation 1.1)is chosen as the best partition mode. The motion vectors of each sub-blocks of the macroblock are then encoded and transmitted. High prediction accuracy can be achieved due to more flexible block-size choices, however it also results in high complexity process of motion estimation.

$$\alpha = argmin(D_i + \lambda R_i) \tag{1.1}$$

Where D is the distortion function which can be the sum of the squared prediction error or the absolute value of the prediction error. R is the total bits used for encoding in mode i. λ is a Lagrange parameter which depends on the quantization parameter in use.

It can be noticed that if the large partition size is chosen, fewer bits are needed to encode the choice of the motion vectors and the type of partition, however, the prediction error signal may contain high energy. If a smaller partition size is chosen, the prediction error signal may contain low energy while requiring more bits to signal the motion vectors and the type of partitions. The choice on the partition size therefore has large impact on the compression performance. For homogeneous region, large block size is more preferred and for the area containing high motion or edging details, a smaller partition size will result in more accurate prediction and therefore achieve better rate-distortion performance.

In H.264/AVC, the motion vector can have as high as quarter pixel resolution for luma components and one-eighth resolution for chroma components, which was not featured in the previous standards.

Due to the importance of the motion vectors, no transformation and quantization should be applied. When small partition sizes are chosen, it may cost significant amount of bits for encoding. As is known that neighboring motion vectors often have high correlation. It therefore can be employed to reduce the necessary bits used for encoding the motion information. H.264/AVC includes the feature of motion vector prediction: each motion vector is predicted from the neighboring motion vectors. The method of obtaining the prediction motion vector (MVp) depends on the partition size of the current and neighboring blocks, also the availability of the neighboring motion vectors. Two examples of current (16×16 case) and neighboring block partition sizes are shown in Figure 1.7.

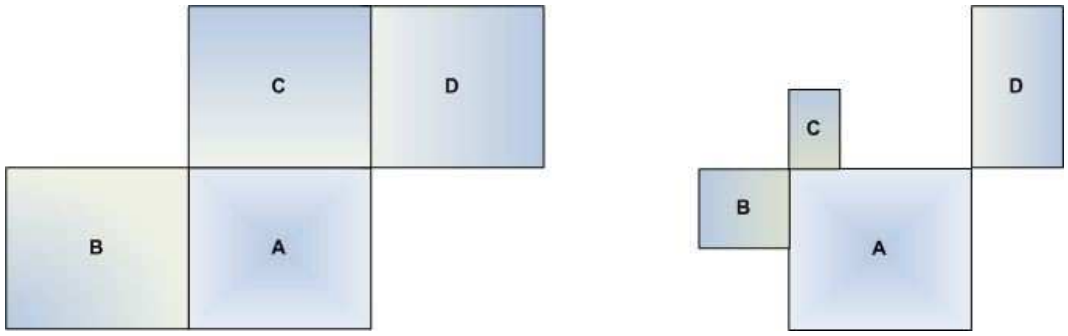

Fig. 1.7. Two Examples of Motion Vector Prediction in H.264/AVC

If the current block (**A**) has the same size as the neighboring blocks of left side (**B**), the one on the top (**C**) and the one at the position of top-right (**D**) are used to predict the motion vector of the current block. Otherwise, the first block (on the top of the other blocks) immediately to the left (**B**) of the current block, the first block (scanning order) on the top (**C**) of the current block and the one immediately on the top-right side (**D**) of the current block are used for prediction. Except the partition size of 16×8 and 8×16, motion vector prediction is computed as the median of the motion vectors of blocks **B**, **C** and **D**. If a macroblock has the partition size of 16×8 blocks, the motion vector predictor of the upper 16 block is predicted from **C** and motion vector predictor of the lower sub-block is predicted from **B**. If a macroblock is partitioned into two 8×16 blocks, the motion vector predictor of the left sub-block is predicted from **B** and the right side sub-block is predicted from **D**. For skipped mode macroblocks, the motion vector predictor is generated as the median of motion vectors of **B**, **C** and **D**. If one or more neighboring blocks (**B**, **C**, **D**) are not available, the choice of obtaining motion vector predictor is modified accordingly. The difference between the motion vector of the current block and the motion vector predictor are calculated and the resulting value is named motion vector difference (MVD). Instead of motion vectors, the motion vector difference of each block are encoded and transmitted which saves large amount of bits comparing to direct encoding of motion vectors. At the decoder side, it exploits the same pattern of motion vector prediction and therefore can decode the motion vectors of each block from received motion vector differences. For macroblocks encoded in skip mode, the motion vector predictors are directly used as the motion vectors to decode these macroblocks.

1.1.2 Intra Prediction

In H.264/AVC, there are total 9 intra prediction modes for 4×4 luma blocks and 4 modes for luma 16×16 blocks. At encoding process, all the prediction modes are compared and the one results in the lowest difference between prediction block and

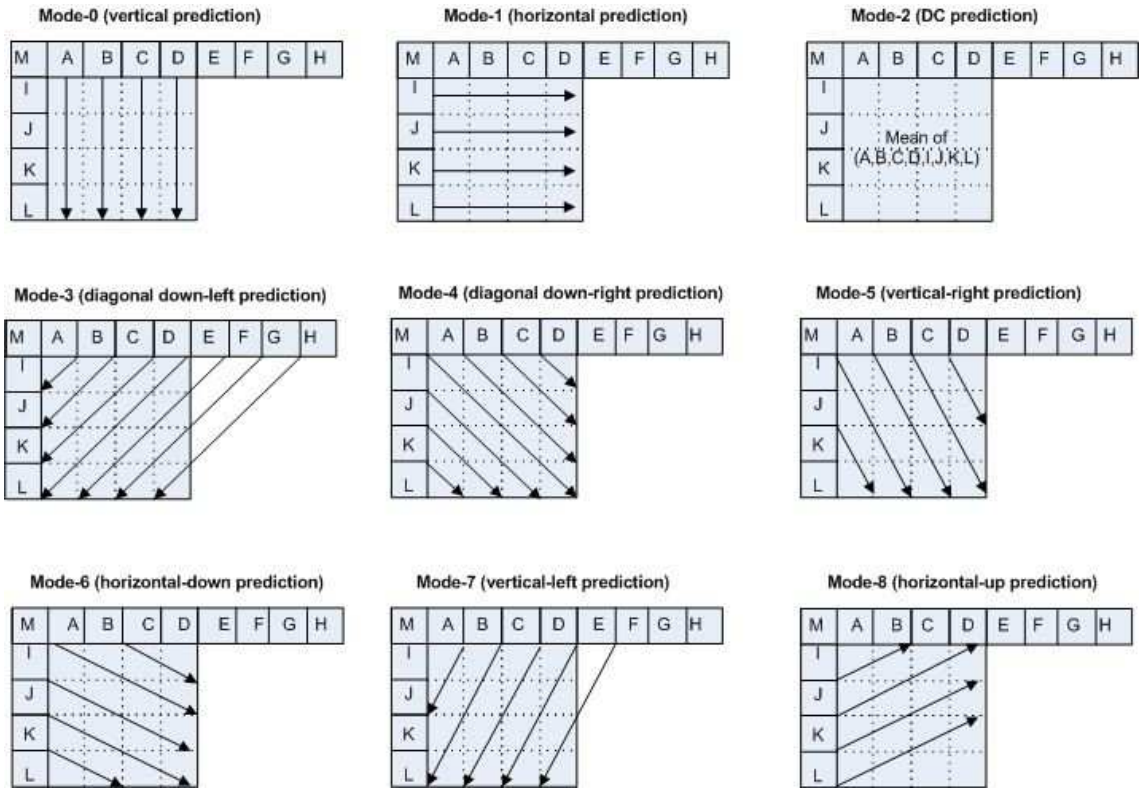

Fig. 1.8. Nine Intra Prediction Modes for 4×4 Blocks in H.264

the current block is selected. The nine prediction modes for 4×4 blocks are shown in Figure 1.8.

1. In mode 0 (vertical prediction mode), the upper samples (A, B, C, D) are extrapolated vertically.

2. In mode 1 (horizontal prediction mode), the left samples (I, J, K, L) are extrapolated horizontally.

3. In mode 2 (DC prediction mode), all samples in prediction block are predicted by the mean of samples A, B, C, D and I, J, K, L.

4. In mode 3 (diagonal down-left prediction mode), the samples are interpolated at 45 degree between lower-left and upper-right.

5. In mode 4 (diagonal down-right prediction mode), the samples are extrapolated 45 degree down and to the right.

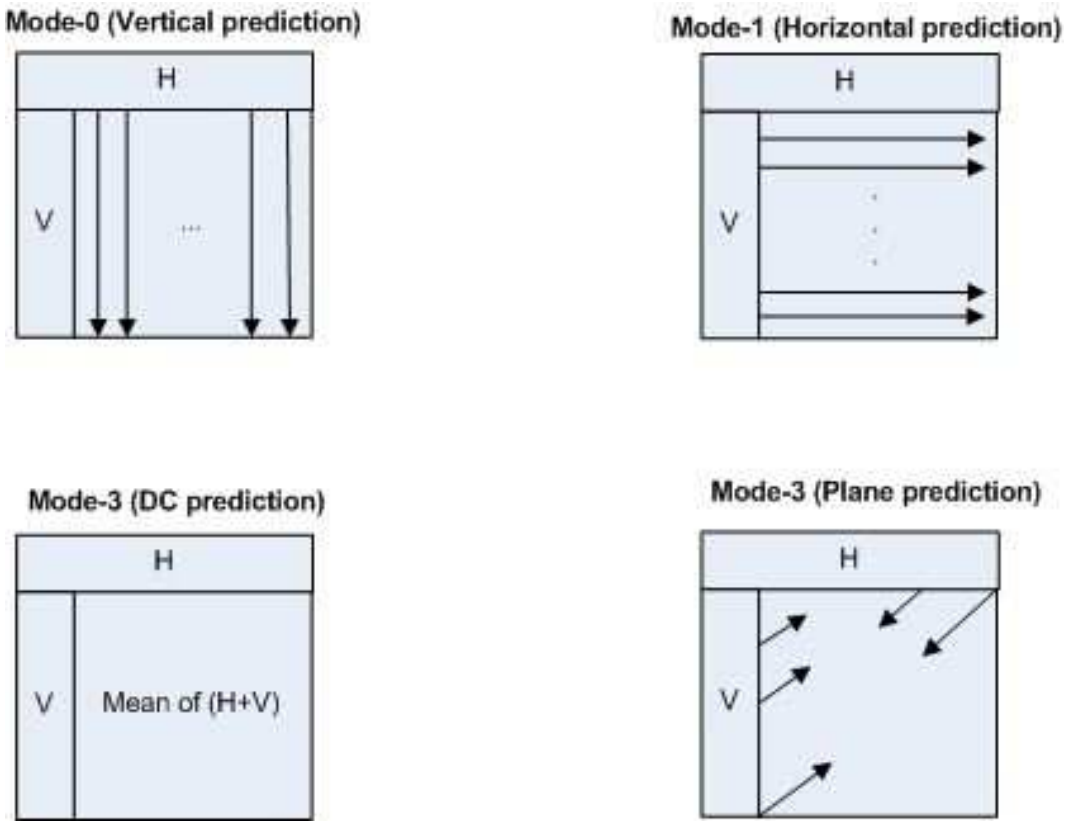

Fig. 1.9. Four Intra Prediction Modes for 16×16 Macroblocks in H.264

6. In mode 5 (vertical-right prediction mode), extrapolation is at approximately 26.6 degree to the left of vertical.

7. In mode 6 (horizontal-down prediction mode), extrapolation is at an angle of about 26.6 below horizontal.

8. In mode 7, the extrapolation is at an angle of about 26.6 degree to the right of vertical.

9. In mode 8 (horizontal-up prediction mode), the interpolation is at an angle of about 26.6 degree above horizontal.

The illustration for 4 types of 16×16 intra prediction modes are given in Figure 1.9.

1. In mode 0 (vertical prediction mode), the extrapolation is obtained from the upper samples

2. In mode 1 (horizontal prediction mode), the extrapolation is obtained from the left samples

3. In mode 2 (DC prediction mode), The prediction block is formed by the mean of the upper and left-hand samples.

4. In mode 3 (Plane prediction mode), A linear plane function is fitted to the upper and left-hand samples. (for smoothly varying luminance area)

1.1.3 Deblocking Filter

Different from previous video standards, H.264/AVC exploits deblocking filter to reduce the blocking distortion of the decoded macroblocks. It smooths block edges and aimed to improve the visual quality of the decoded frames. At the encoder side, a deblocking filter is used in the backward reconstruction path right after the inverse transform and before reconstructing and storing the decoded macroblocks for future reference. At the decoder side, the deblocking filter is used after inverse transform and before reconstructing and displaying the macroblocks. Because the deblocked macroblocks have better quality than the blocky ones, they are used for future motion prediction, which improves the compression performance.

The deblocking filter is applied to the edges of each 4×4 blocks in a 16×16 macroblock in an order of left-to-right for vertical boundaries and top-to-bottom for horizontal boundaries. The amount of the filtering depends on the current quantiser, coding modes of neighboring blocks and the gradient of image samples across the boundary. [28]

1.1.4 Transform and Quantization

While most of the previous video standards use DCT (discrete cosine transform) transform, two types of transforms are applied for luma components in H.264/AVC. In intra predicted macroblocks, a Hadamard transform is used for 4×4 array of luma DC coefficients. For all the other 4×4 residual blocks, a DCT-based integer transform is applied.

There are several advantages of using the integer transform. First of all, all operations can be carried out using integer arithmetic without losing decoding accuracy. Second, by using integer arithmetic, it is possible to ensure zero mismatch between encoder and decoder inverse transform. Thirdly, the core part of the transform can be implemented using only additions and shifts. Last, the scaling multiplication is integrated into the quantizer which reduces the total number of multiplications. At the decoder side, without loss of accuracy, the inverse quantization and inverse transform are carried out using 16-bit integer arithmetic with only a single multiply per coefficient.

The complete transform, quantization, inverse quantization and inverse transform process can be summarized as follows: At the encoding side:

1. Forward transform:

$$W = C_f X C_f^T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} [X] \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \tag{1.2}$$

   Where X is the input 4×4 residual data.

2. Quantization process:

$$Z = W.round(PF/Qstep) \tag{1.3}$$

   Where Qstep is the quantizer step size and PF's value depends on the pixel position as listed in Table(1.1).

Table 1.1

Position	PF
$(0,0)$, $(2,0)$, $(0,2)$, $2,2$	$a^2 = \frac{1}{4}$
$(1,1)$, $(1,3)$, $(3,1)$, $(3,3)$	$\frac{b^2}{4} = \frac{1}{10}$
Other positions	$\frac{ab}{2} = \frac{1}{4}\sqrt{\frac{2}{5}}$

At the decoder side:

1. Decoder scaling:

$$W^{'} = Z.Qstep.PF.64 \tag{1.4}$$

2. Inverse transform:

$$X^{'} = C_i^T W^{'} C_i \tag{1.5}$$

Where $C_i = \begin{bmatrix} 1 & 1 & 1 & \frac{1}{2} \\ 1 & \frac{1}{2} & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 1 & -\frac{1}{2} \end{bmatrix}$

3. Post scaling:

$$X^{"} = round(X^{'}/64) \tag{1.6}$$

Where $X^{"}$ is the output 4×4 residual data

1.1.5 Error Resilience Features in H.264/AVC

In H.264/AVC, several error resilience features are included.

In flexible macroblock ordering (FMO), macroblocks are allowed to be assigned into different slice groups according to certain mapping procedures, such as box-out, raster scan. Inside each slice group, the coding order is still in scanning order. Each slice group is encoded independently and can be decoded independently too. Effectively using flexible macroblock ordering feature can largely improve robustness

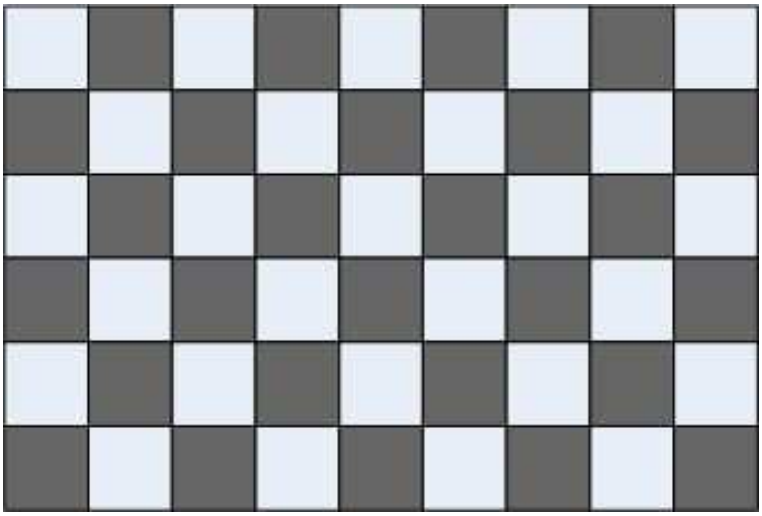

Fig. 1.10. Checker Board Mapping

to video data loss. An example is shown in Figure 1.10, where the checker-board mapping is used. The shaded macroblocks belong to one slice while the rest belong to the other slice group. It can be noticed that if one slice group got lost, each lost macroblock still has three to four decoded macroblocks that can be used to error conceal the lost one. However, the coding efficiency might be lowered due to the broken in-picture prediction along the slices edges.

Redundant picture is also a new error resilience feature used to enhance the robustness to data loss by sending redundant information of the important video data. Different from resending a same copy of the primary data, H.264/AVC allows the redundant slices to be encoded with different parameters and therefore it is not necessarily the same representation as the primary video encoded data. For example, it can be a coarse version of the primary encoded data by changing the quantization parameter which uses less bits.

Data partition is used to give higher protection to more important video data while video transmission. It allows the syntax of each slice to be partitioned into at most three partition groups. In H.264/AVC, header information, including macroblock mode, quantization parameters and motion vectors are put in the highest protected partition (partition **A**) since they are most important information for decoding the

video stream. The intra coded data is put into the second level partition (partition **B**), which is decodable based on the availability of partition **A**. The inter coded data is sent in partition **C**, which can only be useful if partition **A** is available. If data partition is used, the three different partitions are put into three individual bit buffers. At the decoder side, all three types partitions need to be available to start decoding process. If partition B or C got lost, partition A can still be useful by applying error concealment method to recover the lost slices.

Synchronization/switching pictures (SP) is a new feature in H.264/AVC which provides special types of pictures (SP/SI). Identical SP frames can be reconstructed when difference frames are used for predictions. This property makes SP frames similar as intra frames but with more compression efficiency. SP frames can be used for splicing, random access, and error resilience/concealment. SI frames are used in conjunction with SP frames [29], [30].

Other than the above techniques, H.264/AVC allows intra predicted macroblock in the P frames.

1.2 Structures of H.264/AVC data Stream

The output of the H.264/AVC encoder is a series of NAL units. On packet networks, each NAL can be sent in different packets. Each NAL unit contain a RBSP. The types of the RBSP are listed as follows:

- Parameter Set: global parameters for a sequence such as picture dimensions, video format, macroblock allocation map, e.t.c.
- Coded slice: header and data for a slice which contains the coded video data.
- Picture delimiter: boundary between video pictures. If it is not present, the decoder infers the boundary based on the frame number within each slice header.
- Supplemental enhancement information: Side messages that are not essential for correct decoding of the video sequence.

- Data partition: it is used for error resilience purpose. Partition A contains header data for all macroblocks in the same slice. Partition B contains intra coded data and partition C contains inter coded data.

- End of Sequence: it indicates that the sequence is end or in another word the next picture is an IDR picture.

- End of Stream: it indicates that there is no more video data in the bit stream.

The parameter sets defined in H.264/AVC contains information of a large number of coded frames.

- Sequence parameter set: it contains the parameters for a video sequence, which includes an identifier, limits on frame numbers, number of reference frames, picture width and height, choice of interlaced or progressive coding format.

- Picture parameter set: it contains parameters for one ore more decoded pictures, which includes an identifier, a flag for signaling VLC/CABAC, number of reference frames, quantization parameters, a flag signaling the modification of the deblocking filter parameters.

One or more sequence parameter sets and picture parameter sets are normally sent to the decoder prior to decoding of slice headers and encoded video data.The slice header activates a selected picture parameter set for encoding the coded video data within this slice. If a different picture parameter set is needed for encoding a different slice, the slice header activates the corresponding picture parameter set. In the same way, each picture parameter set in use activate a corresponding sequence parameter set in need. The advantages of using parameter set is to signal the important but infrequently changed parameters separately from the coded slices data and therefore save large amount of bits.

1.3 Transmission of H.264/AVC Data Stream over Internet

Video coding layer (VCL) and the network abstraction layer (NAL) are two conceptual layers included in H.264/AVC standards, where video coding layer contains the representation of the coded video data and network abstraction layer defines the interface between the video codes and the transmission channel. NAL units are sent in packets and therefore it fits packet based networks. Several NAL units can be put into one aggregation packet. Each NAL unit can also be sent in separate packet. One NAL unit can also be fragmented into several packets. NAL units are sent in decoding order. If there is any data loss happened during transmission, an error flag in the NAL unit header will indicate that [31].

There are two kinds of typical transmission errors: bit inversion errors and packet losses. Most of the multiplexing protocols include packet loss or bit-error detection and therefore it can be assumed that the erroneous transmission packets can be detected. In H.264/AVC, the bit-erroneous packets are considered as being discarded at the decoder. One reason for it is that handling the bit erroneous data normally complicates the decoding process.

When transmission loss happened, error resilience or error concealment techniques need to be used to minimize the distortion caused by the lost video data within the frame and the error propagation to other frames.

One way to reduce the packet loss rate is to reduce the packet size since the probability of a bit error hitting a short packet is normally lower than for large packets. Moreover, short packets reduce the amount of lost information. The slice or slice group feature in H.264/AVC is to divide a frame into several independent slices which means no intra prediction can be conducted along the slice edges. Each slice is sent in individual packet and the loss of one of them does not affect the decoding of the the other slices within the same frame. It both reduced the packet length and enhanced the decoding ability of the different area in one frame. It also facilitates successful error concealment. However, on the other hand, more packets means more

bits are used for header information of each packet. Also the restricted range of intra prediction reduced the compression efficiency. Therefore, the choice on the packet size plays an important role in dealing with transmission loss while keeping coding performance.

Flexible macroblock ordering is also one of the slice group methods. It allows different ways of mapping a group of macroblocks into one slice group, such as checker-board pattern, interleave pattern, sub-picture within a picture. Flexible macroblock ordering is known to be effective in conjunction use with error concealment techniques. For example, in checker-board pattern, if one slice group got lost during transmission, the macroblocks neighboring the lost macroblock may be received (in another slice group) successfully. Therefore, the neighboring decoded macroblocks can be used to error conceal the lost macroblocks.

The sub-picture mapping is useful if it is combined with a technique that can detect the background area and the motion area. The picture is then divided into several slice groups of different importance. Unequal error protection can be applied to different slice groups to enhance error resilience (motion content slice needs more protection than the background slice since homogeneous background area is easier to be error concealed than the motion area).

When transmission loss happened, the loss statistics are normally unknown to the encoder side. In order to resend the lost information for error resilience purpose, a low-bit-rate feed back channel from decoder to encoder is helpful. This feedback channel can be available for H.264/AVC video transmission using RTP/RTCP protocol. The feed-back channel, depending on the available data rate, can report a b-frame delayed channel loss information observed at the decoder side to the encoder. In RTP/IP protocol, the feedbacks are supported by RTCP messages.

At the decoder side, the successfully received packets are decoded while the error concealment has to be applied for the lost video data. If inter motion prediction is used for the frame with data loss but error concealed, and it is used as the reference for decoding the following frames, error propagation will then occur in those follow-

ing decoded frame, which causes a long-term visual quality degradation. Therefore packet loss has significant impact on the visual quality of the decoded sequence before intra coded frames are used. It has been mentioned that because of using motion compensated prediction, the decoded picture quality depends not only on the lost video data in the current frame but also on the entire lossy sequence decoded before it (if no intra frames have been inserted to stop the error propagation).

A robust decoder should be able to detect the transmission loss and apply the appropriate error concealment technique to recover the lost information. For H.264/AVC decoder, it assumed that before decoding, the erroneous slices are discarded and the correctly received video data are arranged in a correct decoding order. However, the decoder has to detect the the temporal and spacial location of the lost slices. These are usually signaled in the slice or frame header.

Two well-known error concealment methods are applied in H.264/AVC decoder: weighted pixel value averaging for intra frames and boundary matching based motion vector recovery for inter frames. In weighted pixel value averaging method, if a macroblock got lost, the neighboring correctly decoded macroblocks (at least Two) are used for error concealment. Otherwise previously error-concealed neighboring macroblocks will be used for error concealment too. The pixel values in the lost macroblocks are obtained by calculating a weighted sum of the available boundary pixels. The weight of each boundary pixel is relative to the inverse distance between the pixel to be concealed and the boundary pixels.

For inter predicted slices, the correctly decoded motion information is evaluated and if the average length of the motion vector is smaller than a predefined threshold, the lost slices are copied from the co-located positions of the reference frame. If it is greater than the threshold, the motion compensated error concealment method is used. If a block has adjacent blocks whose motion vectors are correctly decoded, these motion vectors are used to obtain the error concealed blocks. If none of the adjacent blocks are available, the neighboring concealed blocks are used instead. The candidate blocks has the smallest boundary matching error, which is the sum of absolute pixel

difference of the adjacent luminance pixels in the concealed block and its decoded or concealed neighboring block, is chosen.

Three types of video application can be categorized by using network protocols: conversational application such as video conferencing, which has high restriction on delays and always limited to point to point or small scale multi-points communication. It also requires real-time video coding and therefore prefer low complexity encoding and decoding process. The second application is downloading or storage pre-coded video streams. For this application, it is not necessary to consider delay and real-time coding requirement. The third application is named IP-based streaming and its delay requirement is somewhere in middle of the first two application.

Maximum transfer unit size (MTU) is an an important parameter when deciding on the packet size. The packet size should never be bigger than the MTU but close to it. In this way, it optimizes the header overhead while minimize the loss probability at the same time. For wired network, the MTU is normally assumed to be around 1500 bytes while only about 100 bytes for wireless transmission environment.

Error resilience method usually add redundancy in order to cope with the transmission loss. However, when error loss rate is high (high congestion network), the congestion control mechanism start to reduce the network load by decrease the encoding data rate by reducing frame rate, picture quality or completely dropping the encoded video data.

1.3.1 Feedback Information Used in H.264/AVC

Feedback information can be used in H.264/AVC video streaming. Decoder can send back the received video data information by acknowledging the correctly received slices and not-acknowledged message for lost slices. The feedback is normally a delayed information but a immediate feed back message is also possible depending on the available bandwidth for transmitting the feedback data. The feedback information can be assumed as error-free and the overhead is assumed to be negligible [32]. The lost data is therefore can be re-transmitted if the encoder received the message

of the loss, however, a delay may exist. Although the transport protocol is not the focus of the H.264/AVC, the flexibility of the H.264/AVC gives the motivation of using feedback in system design.

There are three feedback modes and three kinds of feedback messages defined in [32]. In feedback mode I, the encoder receives the messages about the acknowledged slices that have been successfully received. Encoder then restrict the reference areas of encoding the following slices to be those acknowledged slices. If not such a reference is available, the intra mode is used. In the feedback mode II, the reference slices in the encoder are those acknowledged area and also the not-acknowledged area with the same error concealment method applied on as the decoder used. In feedback mode III, the reference area is not restricted but an expected distortion is computed and updated at the encoder side. The channel is deterministic by receiving feedback messages.

1.4 Error Resilience Based on Wyner-Ziv Lossy Coding Theory

In 1970s, Wyner and Ziv proved that the minimum source encoding rate for a given distortion, when the side information is only known to the decoder, is greater or equal to the the rate obtainable when the side information is available at both encoder and decoder sides. Therefore Wyner-Ziv coding is also referred to as lossy compression with side information at the decoder. This distributed source coding theory implies two important applications, the low complexity encoding and error resilient transmission. In our work, we developed the application of the Wyner-Ziv theory on error resilience.

Conventional video transmission uses forward error correction (FEC) systems for error resilience. However, when the channel error rate exceed the error-correction capability, a highly unacceptable video quality, known as cliff effect, can be observed. One improvement on solving this problem is using unequal error protection method, which give higher error protection to more important video information. However,

this requires to use layered video coding, which is not used in practice because of the inefficient rate-distortion performance.

By adding an independent Wyner-Ziv encoder which processes the same video source as the primary video encoder, e.g, MPEG encoder, and using the decoded video signals from the primary decoder as the side information to the Wyner-Ziv decoder, an error-resilient transmission can be achieved. This achievement is based on the error correction capability of the Turbo Codec (or Reed-Soloman code, or LDPC code), which serves as the lossless Slepian-Wolf's codec block inside the Wyner-Ziv codec. Turbo code has been proved that can come close to the Slepain-Wolf bound in [33]. Another part inside the Wyner-Ziv codec is a coarser quantizer, comparing to the one used in the primary codec. Therefore the Wyner-Ziv description output from the coarser quantizer uses less data rate. It is then sent to the Turbo encoder and output the parity bits which use even less bite rate and is transmitted to the decoder side. The Turbo decoder uses the side information and the parity bits to recover the coarser description of the video signals. When the unacceptable large channel error happened, this coarser reconstructed signal could limit the upper bound of the error and therefore avoid the cliff effect. Plenty results have shown that this error resilience approach gives much better visual quality than using conventional forward error correction in the transmission of primary video sequence. The basic block diagram of this algorithm is given in Fig(1.4).

1.4.1 Improvements on Wyner-Ziv Error Resilience Algorithms

The first result in Wyner-Ziv error resilience application is presented in [3], 2003, in which the systematic lossy forward error protection (SLEP) coding scheme was first time proposed to mitigate the channel error effects in digital video broadcasting. The improvements on Wyner-Ziv error-resilient transmission based on SLEP system are summarized in the following sections.

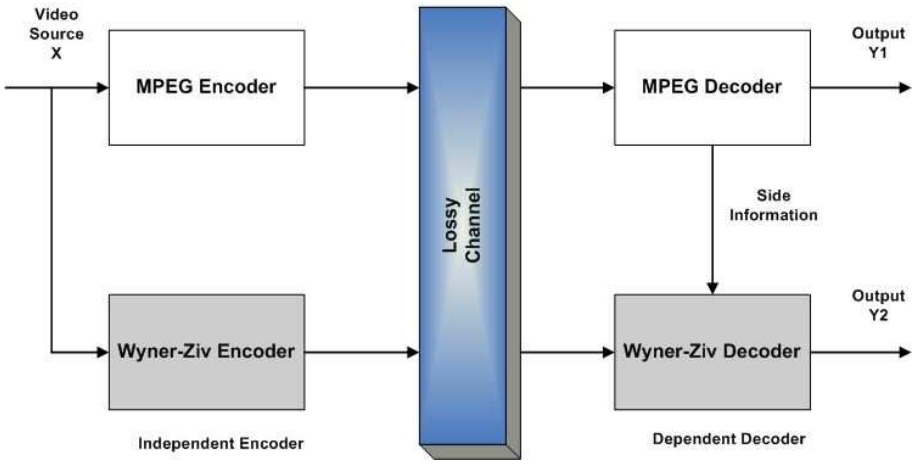

Fig. 1.11. Basic Wyner-Ziv Codec for Error Resilience

First Result on Wyner-ziv Error Resilient Video Transmission

The simplest method by using pixel domain Wyner-Ziv coding in SLEP system and the first results of applying Wyner-Ziv codec for error-resilient transmission were introduced in [3]. The system part contains the primary MPEG codec, which does not use conventional FEC during transmission over the error-prone channel. The Wyner-Ziv codec in this system consists of a coarse quantizer and a turbo encoder at the encoder side; a turbo decoder and a reconstruction block at the decoder side. The system structure is shown in Fig(1.4.1). Wyner-ziv encoder is independent of the MPEG encoder, both of which have the same input video sequence. Wyner-ziv decoder is dependent on the MPEG decoder by using its output as the side information to reconstruct the video information and limit the error up to a residual distortion. The results of this technique were shown in [3], which gave good visual quality in the decoded frames than those using the conventional FEC algorithm.

In no channel error case, the Wyner-Ziv codec outputs the same data as those from the MPEG decoder, therefore is completely redundant. Only when the channel error corrupts part of the bit stream, the Wyner-Ziv codec can do its help to limit the error to an upper bound which depends on the coarse quantizer.

The advantages of using SLEP system are: it can potentially achieve a much lower data rate than the conventional channel coder, e.g.FEC, since the SLEP system allows for some distortion by the channel errors. At the same time, Wyner-Ziv codec guarantees stronger error protection, especially for high distortion case.

The drawback is that since the pixel domain Wyner-Ziv coding does not exploit any spatial and temporal correlation in the input video source, the required Wyner-Ziv Data rate is still high.

Embedded Wyner-Ziv Codec

An improvement on the basic Wyner-Ziv coding for error resilience is proposed in [4], by exploiting the trade-off between the allowable transmission errors and Wyner-

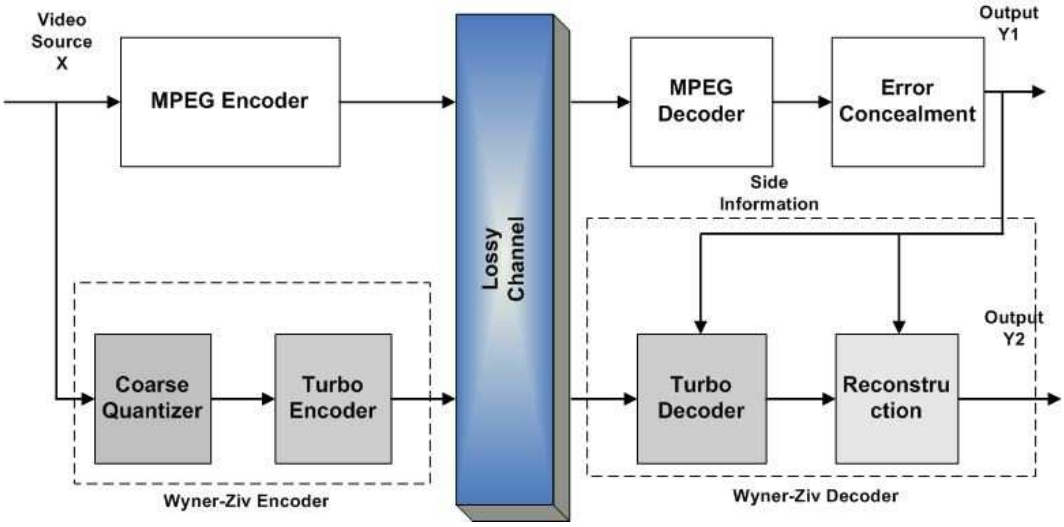

Fig. 1.12. Pixel Domain SLEP Based on Wyner-Ziv Coding

Ziv data rate, to construct an embeded Wyner-Ziv codec. The system block diagram is shown in Fig1.4.1.

At the encoder side, the system part set-up is the same as before. The Wyner-Ziv codec (I) used a coarser quantization and the Wyner-Ziv codec (II) used a finer quantizer. Wyner-Ziv codec (I) has stronger error protection capability due to that the coarsely quantized data stream is easier to decode. The concealed signal from primary decoder is sent to the Wyner-Ziv decoder (I) as the side information to reconstruct the output video sequence, whose distortion is upper bounded by the Wyner-Ziv codec (I). If the channel error is not too sever, the spare data rate can be used to decode the Wyner-Ziv data stream (II). The decoded signal from Wyner-Ziv decoder (I) and the primary decoder is served as the side information to Wyner-Ziv decoder (II) and yielded an improved reconstructed signal Y_2.

The simulation results shown in [4] demonstrated that when the channel error is not too sever and the data rate is more than enough to correctly decode all the frames by only using the first level Wyner-Ziv coder, the peak signal to noise ratio (PSNR) was improved limitedly due to the coarse quantizer. Therefore, for this case, the embedded system could be more effective by encoding and decoding a finer Wyner-Ziv description by Wyner-Ziv codec (II).

Wyner-ziv Coding Applied to Temporally Subsampled Source Sequence

This technique is simple but necessary to be mentioned. The basic system structure is the same as the one level pixel-domain Wyner-Ziv codec. The only difference is that only every n-th frame is transmitted by the Wyner-Ziv codec. The data rate used by Wyner-Ziv coding is decreased to $total - data - rate/n$. However, this is at the expense of an increase in the average distortion due to transmission errors. The simulation results were give in [4]. The trade off between the temporal subsampling versus Wyner-Ziv data rate should be an important consideration in the further improvement.

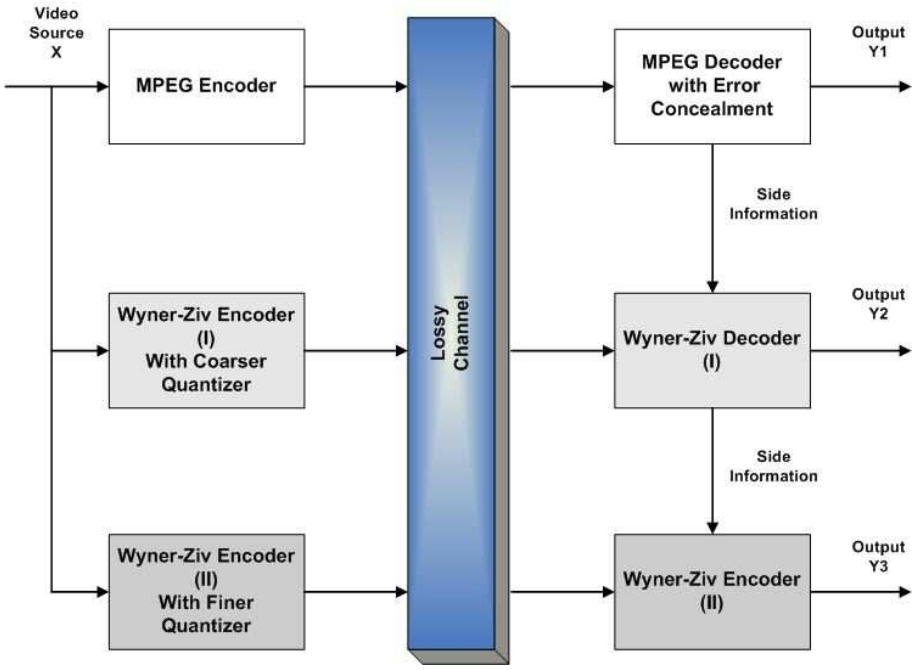

Fig. 1.13. Embeded SLEP Technique Based on Wyner-Ziv Coding

Hybrid Video Codec

Further improvement on SLEP system is using Reed-solomon (RS) codec instead of the Turbo codec in the Wyner-Ziv codec [5]. RS code is usually used in conventional FEC in MPEG video transmission. The name of hybrid video codec comes from this reason. In this system, the input video frames to the hybrid video code uses the same slice structure as that of MPEG video code. The Wyner-Ziv description output from the coarser quantizer is then sent to RS coder which applies systematic RS codes across the slices of an entire frame. Only the RS parity symbols were transmitted to the receiver. RS code also has error correct capability. It uses the received RS parity bits and the error-prone Wyner-Ziv description from the MPEG decoder to obtain an error-free Wyner-Ziv description. Since the location of the lost slices is known, the RS decoder can perform erasure decoding across the error-prone slices.A fall back mechanism substitutes the lost slices in the main video sequence with their correct but coarser versions. The coarse fall back causes some prediction mismatch which propagates to the subsequent frame but the visual examination of the decoded sequence showed that this small error is imperceptible [5]. To prevent prediction mismatch between the two coarse video encoders (one at encoder side and the other on in the decoder side as shown in Fig(1.4.1)), it is necessary to ensure that they use the same reference frame for predictive coding. This is done by requiring the coarse encoder inside the Wyner-Ziv encoder to use the locally decoded reference frame from the main MPEG encoders as a reference frame for predictive coding.

It is the first time mentioned in [5] that the redundant slice feature in H.264 software can be used to generate the second Wyner-Ziv description.

The simulation results showed that this scheme could achieve a much lower data rate used for Wyner-Ziv encoding and a better visual performance in the decode frames than the conventional FEC.

The trade off between the Wyner-Ziv data rate and the residual distortion can be exploited to add an embedded Wyner-Ziv coder that helps to achieve a graceful

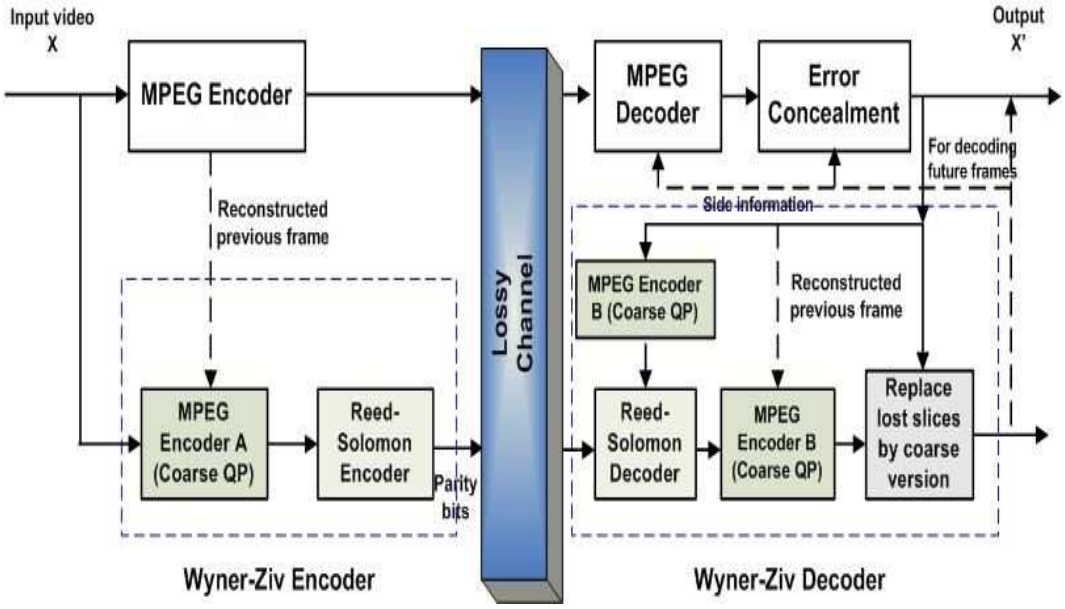

Fig. 1.14. Hybrid SLEP System Using Reed-Solomon Coder

degradation of the decoded video frames. This idea resulted in the pixel domain embedded SLEP system in [4], which could also be implemented for hybrid video codec as given in [5].

1.5 Overview of The Thesis

1.5.1 Contribution of The Thesis

In this thesis, we developed error resilience techniques based on Wyner-Ziv coding [21], [22], [23], [25], [26]. The main contribution of the thesis are:

- Unequal Error Protection Based on Wyner–Ziv Coding

  Our work on unequal error protection [21] using Wyner–Ziv coding is motivated by the SLEP system. However, there are two major differences from SLEP. First, instead of protecting everything associated with coarsely reconstructed frames, we separately input motion information and the transform coefficients from a primary H.264 encoder to our Wyner–Ziv encoder and protect them independently. Second, since motion information and the transform coefficients are processed independently, different parity data rates can be assigned according to their impact on the quality of the decoded frames. This results in an efficient way of protecting the video elements based on their importance level. Experimental results show that proposed scheme significantly improved the visual quality of corrupted frames at the expense of a small increase in data rate.

- Content Adaptive Unequal Error Protection Based on Wyner–Ziv Coding

  In this work, we improved the performance of our unequal error protection technique by adapting the parity data rates of the protected video information to the content of each frame [22]. A content adaptive function was used to evaluate the sum of the absolute difference (SAD) between the reconstructed frame and the predicted frame. Depending on pre-selected thresholds, the parity data rates assigned to the motion information and the transform coefficients

were varied for each frame. This resulted in a more effective and flexible error resilience technique that had an improved performance compared to our original work.Experimental results show that the proposed scheme significantly improved the quality of corrupted frames and efficiently utilized the available data rate.

- Feedback Aided Unequal Error Protection

  In this work, we developed a feedback aided error resilience technique based on the previously proposed unequal error protection method. At the decoder, the current packet loss rates are estimated based on the received data and sent back to the Wyner–Ziv encoder via the real-time transport control protocol (RTCP) feedback mechanism. This is utilized by the Turbo encoder, as a Slepian–Wolf lossless coder inside the Wyner–Ziv codec, to update the parity data rates of the motion information and the transform coefficients, which are still protected independently. At the Wyner–Ziv decoder, the received parity bits together with the side information from the primary decoder are used to decode the corrupted slices. These in turn are sent back to the primary decoder to replace their corrupted counterparts. It is to be noted that simply increasing the parity slices when the packet loss rate increases is not applicable, since it will exacerbate network congestion [24], [34], [35]. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases the primary data transmission rate should be lowered in order to spare more bits for parity bit transmission. Also, considering the advantage of unequal error protection, when the packet loss rate is high, more data rate should be allocated to motion information parity data rate since motion vectors mostly have higher importance than the transform coefficients. Whenever feedback is used in a communication system, the amount of delay it incurs needs to be taken in consideration. In our feedback system, an RTCP packet is sent back to the Wyner–Ziv encoder by using the immediate/early RTCP feedback mode,

whenever possible. The immediate RTCP feedback mode is the mode that is capable of reporting every loss event immediately back to the receiver. However, the success of sending the feedback message in such a timely manner also depends on the available bandwidth [36], [37]. The proposed method resulted in an efficient allocation of the data rate by the Wyner–Ziv codec for the purpose of error resilience and consequently provided good quality decoded video when data had been corrupted by transmission errors.

- Feedback Aided Content Adaptive Unequal Error Protection

  In this work, we studied the case of combining the content adaptive function in the encoder side and the channel loss feedback aided from the decoder side. In this method, a new content adaptive function directly analyzing the statistics of the motion information and the transform coefficients is implemented in the Wyner-Ziv encoder. The improved rate distortion performance is shown from the experiment results. Further more, the analysis and the modeling of the whole system is studied in order to find an optimal data rate allocation. The experiment results showed the improved performance on both the rate distortion and the visual quality of the decode video frames.

1.5.2 Organization of The Thesis

In Chapter 2, we presented an error resilient scheme that utilizes a Wyner-Ziv codec to protect mode data, motion vectors and reference frames' indices produced by an H.264/AVC coder more than coarsely quantized transformation coefficients. Experimental results show that proposed scheme significantly improved the visual quality of corrupted frames at the expense of a small increase in data rate.

In Chapter 3, we presented an error resilient scheme that utilizes a Wyner-Ziv codec to protect important video information produced by an H.264/AVC coder with unequally assigned parity data rates. The motion information and the transform coefficients are protected independently. The parity data rate settings are updated

for each frame according to the frame content. Experimental results show that the proposed scheme significantly improved the quality of corrupted frames and efficiently utilized the available data rate.

In Chapter 4, we presented a feedback aided unequal error protection technique that is based on our previous unequal error protection method. Both used Wyner–Ziv coding for error resilience. In this technique the parity data rates allocated to motion information and the transform coefficients were adjusted adaptively depending upon feedback received from the decoder regarding packet loss rates. The proposed method resulted in an efficient allocation of the data rate by the Wyner–Ziv codec for the purpose of error resilience and consequently provided good quality decoded video when data had been corrupted by transmission errors.

In Chapter 5, we presented the technique of combing the content adaptive function in the encoder end and the feedback on packet loss rate from the decode end. The content adaptive function is also improved by analyzing both the residual frames and the sum of the difference on motion vector lengths.

Chapter 6 concludes the thesis.

2. UNEQUAL ERROR PROTECTION BASED ON WYNER–ZIV CODING

Our work on unequal error protection [21] using Wyner–Ziv coding is motivated by the SLEP system. However, there are two major differences from SLEP. First, instead of protecting everything associated with the coarsely reconstructed frames, we separately input motion information and the transform coefficients from a primary H.264 encoder to our Wyner–Ziv encoder and protect them independently. Second, since motion information and the transform coefficients are processed independently, different parity data rates can be assigned according to their impact on the quality of the decoded frames. This results in an efficient way of protecting the video elements based on their importance level.

To further reduce the data rate of the Wyner–Ziv encoder's output, it is necessary to use unequal error protection. This can be achieved by increasing the parity data rate assigned to the data elements that have the strongest impact on the visual quality of the decoded video. Experimental results have indicated that correctly decoded motion vector differences with poorly reconstructed residual transform coefficients have more effect on the visual quality of decoded video frames than properly reconstructed residual transform coefficients with corrupted motion vector differences. Thus, more parity data rate should be used for motion vector differences than those used for the coarsely quantized transform coefficients. The same was observed for mode differences, that is mode differences impact the visual quality of a decoded frame more than the transform coefficients. An exception however, occurs when there is a scene change or the video sequence contains motion that is not properly accounted for by the motion vectors. In this case, the residual transform coefficients can adversely effect the quality of the decoded video if not properly reconstructed, and hence they should be allocated more parity data rate.

This chapter is organized as follows: Section(2.1) provides a detailed description of the unequal error protection system. Experimental results are provided in Section (2.2), followed by the conclusion in Section (2.3).

2.1 Unequal Error Protection Using Wyner–Ziv Coding

In this work, rather than using a Wyner–Ziv encoder as an independent supplementary video codec, we propose to use it as a second layer encoder to protect the important video information produced by an H.264/AVC encoder as shown in Figure 2.1. The Wyner–Ziv codec uses a Turbo coder to perform Slepian–Wolf coding.

The primary codec of the system is an H.264/AVC codec. Associated with it is a Wyner–Ziv codec that utilizes coarse quantization and two pairs of Turbo codecs. Instead of protecting everything associated with the coarsely reconstructed frames, we separately protect motion information and transform coefficients produced by the primary H.264 encoder. The idea being that since the loss of motion information impacts the quality of decoded video differently from the loss of transform coefficients, both should receive unequal levels of protection that are commensurate with their respective contributions to the quality of the video reconstructed by the decoder [21]. The block diagram depicting the unequal error protection system is shown in Figure 2.1.

In H.264/AVC, there are 9 modes used for predicting a 4×4 block in an I frame and 4 modes for predicting a 16×16 block from its neighbors [28], [27]. The mode index and the transform coefficients are critical for proper frame reconstruction at the decoder. In the case of P and B frames, the H.264/AVC standard allows the encoder the flexibility to choose among different reference frames and block sizes for motion prediction. In particular, the standard permits block sizes of 4×4, 4×8, 8×4, 8×8, 8×16, 16×8, and 16×16. Since motion vectors belonging to neighboring blocks are highly correlated, motion vector differences (MVD) are encoded and transmitted to the decoder side, together with the reference frame index, mode information and the residual transform coefficients.

In our unequal error protection scheme, important video information is protected through a Pseudo Wyner–Ziv coder. In the case of I frames, mode information (MI) as well as the transform coefficients are protected whereas motion vector differences, mode information and reference frame index (RI) are protected for P and B frames. These are used to create long symbol blocks that are sent to the Turbo encoder.

In order to mitigate the mismatch between the transform coefficients input to the Wyner–Ziv encoder and the corresponding side information at the Wyner–Ziv decoder, an inverse quantizer, identical to the one used in the H.264/AVC decoder, is initially used to de-quantize the coefficients. These are then coarsely quantized by a uniform scalar quantizer with 2^N levels ($N \leq 8$), and used to form a block of symbols that is passed onto the Turbo encoder. The quantization step size for processing the transform coefficients is therefore $2^{(8-N)}$. In all cases, the output of the Turbo encoder is punctured to reduce the overall data rate.

Due to the importance of maintaining its accuracy the motion information is not quantized, instead, the Turbo encoder accepts the motion information directly and outputs the selected parity bits. It can be noticed that without using quantization, the processing of Turbo coding motion information itself is not strictly speaking Wyner-Ziv coding. Therefore, we name the whole secondary encoder a Pseudo Wyner-Ziv encoder instead of Wyner-Ziv encoder, and we refer to this scheme as unequal error protection using Pseudo Wyner–Ziv coding (UEPWZ). However, the application of Turbo coding in our schemes is different from straight forward error control coding. In our application, only the parity bits p produced by the Turbo encoder are transmitted to the decoder. The output data stream u from the first branch is not transmitted to the decoder side. This is illustrated in Figure 2.2. The corresponding decoded error prone primary video data from the H.264 decoder will be used to co-decode the parity bits received by the Turbo decoders.

The Turbo encoder used consists of two identical recursive systematic encoders (see Figure 2.2) [38], each having the generator function: $H(D) = \frac{1+D^2+D^3+D^4}{1+D+D^4}$. The input symbols sent to the second recursive encoder are interleaved first in a permuter

before being passed to it. A puncture mechanism is used to delete some of the parity bits output from the two recursive encoders, in order to meet a target parity data rate. Only parity bits are transmitted to the decoder side. The first branch of data, indicated by the dashed line in Figure 2.2, is not transmitted. The error correction capability of the Turbo coder also depends on the length of the symbol blocks. In our scheme, the symbol block length is proportional to the size of a frame instead of a slice. Each packet contains the video data corresponding to one slice. Therefore, if a packet is lost, the video information for this slice is dropped.

For the transform coefficients, the symbol block length is 25344 for a QCIF sequence. In the proposed scheme the motion vectors are obtained for each 4x4 blocks, which makes the symbol block length of 3168. The experimental results also show that the Turbo encoder maintains strong error correction ability for such a symbol block length.

The Turbo decoder utilizes the received parity bits and the side information from the H.264/AVC decoder, to perform iterative decoding using two BCJR-MAP (maximum a posteriori) decoders [38]. The error corrected information is then sent back to the H.264/AVC decoder to replace the error corrupted data. In this process, the decoded error-prone transform coefficients are first sent to a coarse quantizer, which is the same as the one used at the Pseudo Wyner–Ziv encoder side. The reason is that at the encoder side, in order to save data rate usage by the Wyner-Ziv coding, a coarse version of the transform coefficients is Turbo encoded. However, only the output parity bits are transmitted to the decoder side. The video data u output from the Turbo encoder is not transmitted. Instead, the H.264 decoded error prone transform coefficients are used as is, together with the received parity bits of the Turbo encoded coarse-version transform coefficients, to decode the error corrected coarse version of the transform coefficients. Therefore, it is necessary to coarsely quantize the H.264 decoded error prone transform coefficient before passing them to the Turbo decoder.

When using the real-time transport protocol (RTP), packet loss can be inferred at the decoder easily by checking the sequence number field in the RTP headers, thus

Wyner-Ziv decoding is only performed when the decoder detects packet losses. When no packet loss happens, the H.264 decoded transform coefficients are used for decoding the residual frames. However, when packet loss happens, the coarser version of the transform coefficients decoded by the Turbo decoder is used to limit the maximum degradation that can occur.

Similarly, the error corrupted motion information received by the H.264/AVC decoder is sent directly to the corresponding Turbo decoder, together with the received corresponding parity bits, to decode the error corrected motion information. This is then sent back to the H.264/AVC decoder to replace the error-corrupted motion information. The reconstructed frames can be further used as reference frames to decode subsequent frames. Therefore, the final version of the decoded video sequence is obtained based on the error corrected motion information and the transform coefficients.

In the case of serious channel loss and/or limited available data rate for error protection, the Pseudo Wyner–Ziv coder might not be strong enough to recover all the lost video information. At this point, UEPWZ takes advantage of allocating different protection levels to different protected video data elements depending on their overall impact on the decoded video sequence. Experiments have shown that by assigning unequal data rates for protecting motion information and the transform coefficients, the rate distortion performance can be improved compared to the equal parity data rate allocation case.

A similar approach is adopted for B frames except that in this case the indices of the reference frames have to be passed onto the Wyner–Ziv encoder as well.

The total data rate allocated for transmitting the parity data is obtained as:

$$WZDR_{uep} = \sum_{i=1}^{N} TC_i + \sum_{j=1}^{N} MVD_j \tag{2.1}$$

$$= PDR_{TC} \times H_{TC} \times W_{TC} \times PN_{symbol} \times FR \tag{2.2}$$

$$+ PDR_{MVD} \times H_{MV} \times W_{MV} \times PN_{symbol} \times FR \tag{2.3}$$

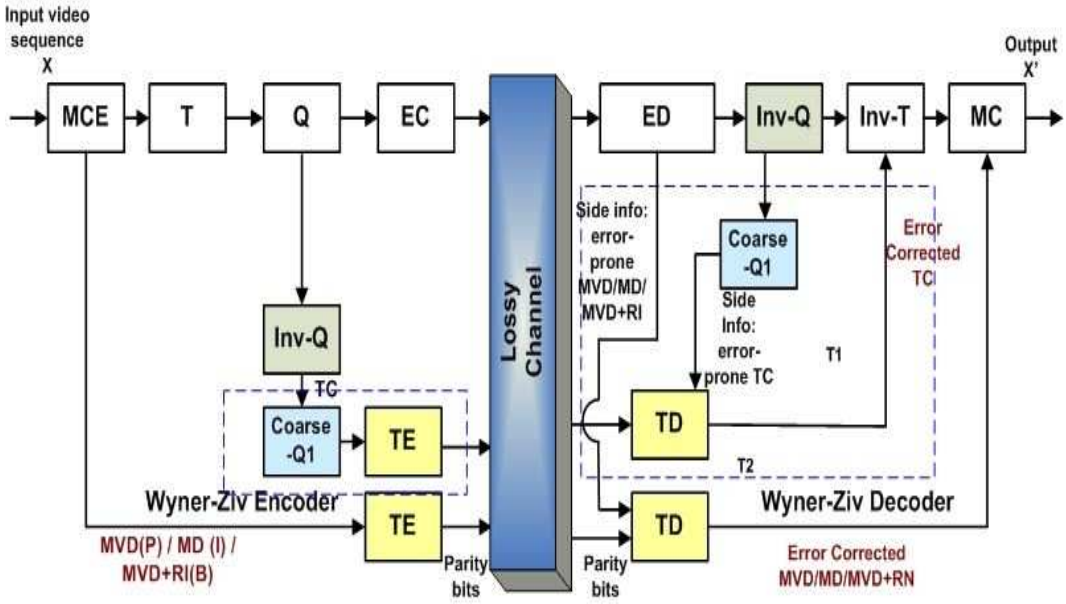

Fig. 2.1. Unequal Error Protection Using Wyner-Ziv Coding For Error Resilience

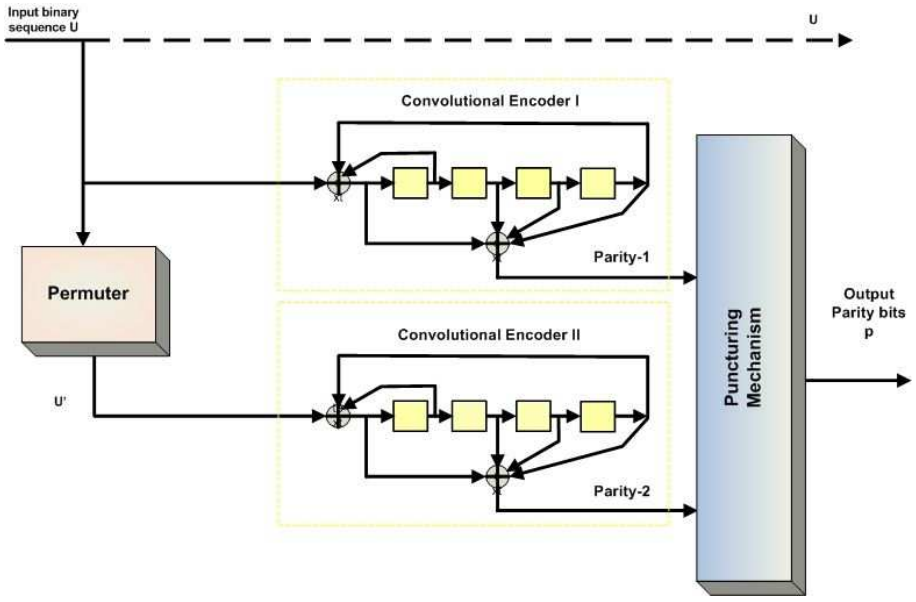

Fig. 2.2. Parallel Turbo Encoder

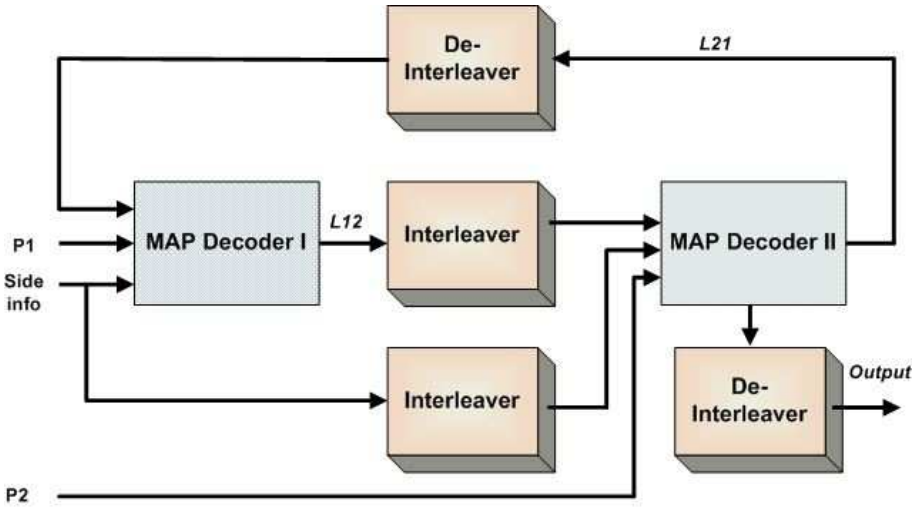

Fig. 2.3. Iterative Turbo Decoder

where H_{TC} and H_{MVD} are the height of the TC and MVD blocks, respectively. The W_{TC}, W_{MVD} is the width of the TC and MVD blocks, PN_{symbol} is the number of the symbol plane and the FR is the current frame rate for transmitting the video packets. N is the total number of the frames in the video sequence. PDR_{TC} is the parity data rate allocated to transform coefficients and PDR_{MVD} is the parity data rate allocated for protecting motion vector differences. For UEPWZ, the parity data rate is fixed for processing the whole video sequence. In UEPWZ, the data rate ratio between transmitting H.264 encoded video data and the parity bits can go as low as 14% to correct enough lost data so that the visual quality of the decoded video looks well. Depending on the total data rate limit, the ratio between TC and MV can be different. At very low total data rate range, there are not enough data rate allocated to both TC and MV parity bits transmission. In these cases, since protection on MV is more important than TC, the data rate allocated to TC can be zero. When the total data rate is higher, the parity rate ratio between TC and MI can be $1:2$ to $1:3$ or higher depending on the total available data rate.

2.2 Experimental Results

To evaluate the current approach, experiments were carried out using version 10.2 of the H.264/AVC reference software on various sequences. All sequences were compressed at 15 frames per second with a GOP structure *I-P-P-P....* Each frame in a sequence was divide into 9 slices that were packetized into individual packets. The parity bits corresponding to each slice were also placed into individual packets, and both data packets and parity packets were sent over a channel and subjected to random losses ranging from 1 packet/frame to 8 packets/frame.

Data networks suffer from two types of transmission errors: random bit error and packet drop. In our experiments, we only consider the case of packet erasures, whether due to network congestion or uncorrected bit errors. Lowering the total data rate to reduce the network congestion is a realistic solution when packet loss is very high. However, since our main application is for video streaming over wireless networks in

which case the packet loss situation is more complicated, we did not consider it in our current experiments. It is to be noted that simply increasing the parity bits when the packet loss rate increases is not applicable, since this will exacerbate network congestion. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases, the primary data transmission rate should be lowered in order to spare more bits for parity bits transmission. Finally, it was assumed that packet losses were uniform.

In the experiments, the parity data rates were fixed. The parity data rate assigned to Turbo encoding the transform coefficients is 0.5 bpp and the parity data rate assigned for protecting motion information is 0.125 bpp. The total data rate used to transmit the parity bits was 71.28 kbps when the data rate used for transmission of the primary H.264 data was 408 kbps. This amounts to a ratio of 1 : 5 for the data rate used in transmitting the parity bits and the primary H.264 video data.

It was observed after extensive testing that a ratio of 1 : 5 achieved the best performance, in PSNR, for the various packet loss rates that were simulated.

We compare the performance of our technique using both equal error protection (EEP) and unequal error protection (UEPWZ) with an implementation of SLEP [3] that uses Turbo codes, and operates at the same data rates. In addition, we use error concealment (any lost slice is replaced by the co-located slice from a previously decoded frame) to circumvent the effect of channel errors.

Figure 2.4 depicts the PSNR curve of the first 100 frames of the *foreman* sequence when 4 packets/frame were lost. In this case the data rate of the H.264/AVC encoder was 408 kbps, with a corresponding Wyner-Ziv data rate of 71.28 kbps. The SLEP encoder was operated at the same data rate, and when error concealment was to be performed the H.264/AVC encoder alone was operated at the combined rate of 479.28 kbps. As can be seen using UEPWZ outperforms EEP, SLEP, and the use of error concealment only. This is because more parity data was allocated to the mode differences, motion vector differences, and reference frame indices than were allocated to the transform coefficients for a given target data rate. In addition,

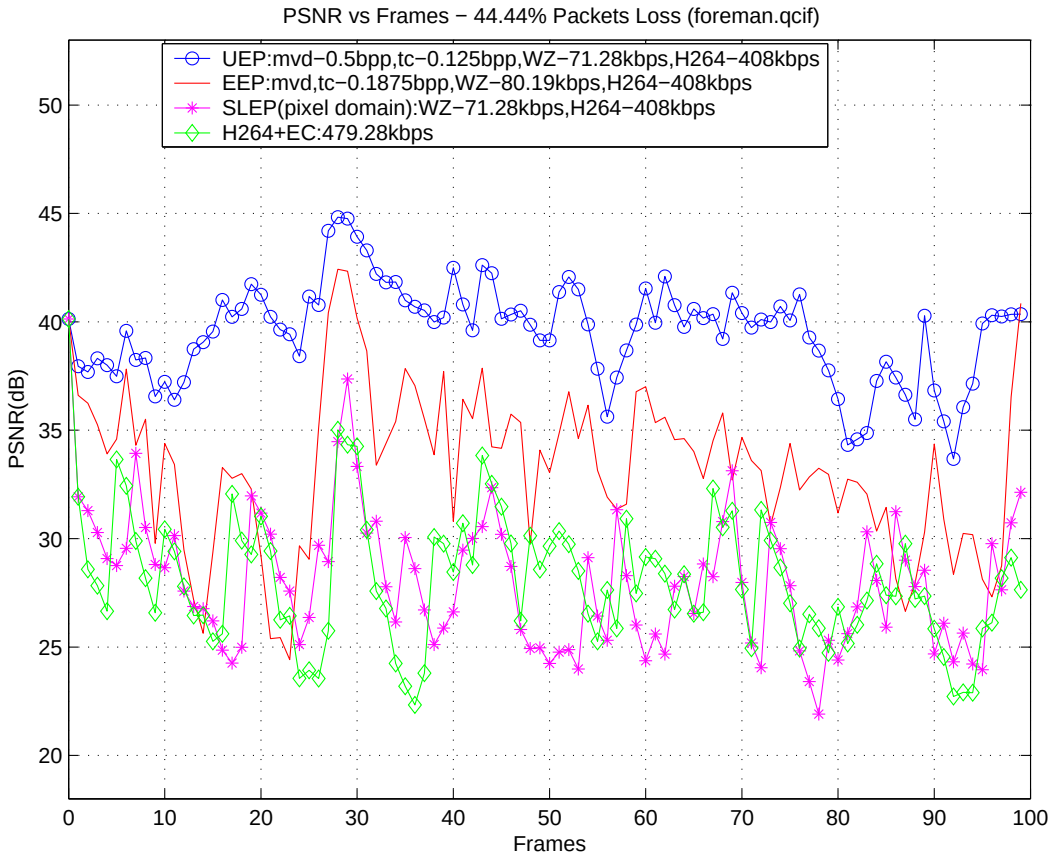

Fig. 2.4. PSNR vs Frames at a loss rate of 4 packets/frame for the *foreman* sequence

using the proposed technique in an EEP mode performed better than SLEP. This is because the Wyner-Ziv encoder in the particular implementation of SLEP used accepted coarsely quantized pixels as input, whereas the proposed technique utilizes mode and motion data which have a more compact representation.

Figure 2.5 and 2.6 exhibit the performance of the four strategies when the packet losses range from 1 packet/frame to 8 packets/frame, in the case of *foreman* and *carphone* respectively. Again, UEPWZ outperforms the other three techniques. It is interesting to note that when the H.264/AVC coder data rate is increased, the use of error concealment results in a performance close, in the PSNR sense, to that of SLEP even though for the most part SLEP generated better looking images than could be obtained using error concealment. It was also observed that none of the four methods could yield visually pleasing images at loss rates exceeding 7 packets/frame. In this case more data was lost than could be corrected by the Turbo decoder.

Figure 2.19, 2.20, 2.21, and 2.22 depict a frame from *foreman* and *table tennis* respectively, that has been decoded by the four strategies. As can be seen unequal error protection scheme achieves the best visual quality compared to other methods. It was also observed that using unequal error protection resulted in a more graceful degradation, compared to the others, even when the packet loss rate was as severe as 6 packets/frame.

Figure 2.7-2.18 show the results of rate distortion performance for QCIF and CIF video sequences at packet loss of 1, 2, 3 packets per frame. It can be seen that UEPWZ always performance better than EEP and H264EC due to effective unequal parity data rates allocation for transform coefficients and the motion information.

2.3 Conclusion

In this chapter we presented an error resilient scheme that utilizes a Wyner-Ziv codec to protect motion vectors and reference frames' indices produced by an H.264/AVC coder more than coarsely quantized transformation coefficients. Experimental results show that proposed scheme significantly improved the visual quality of

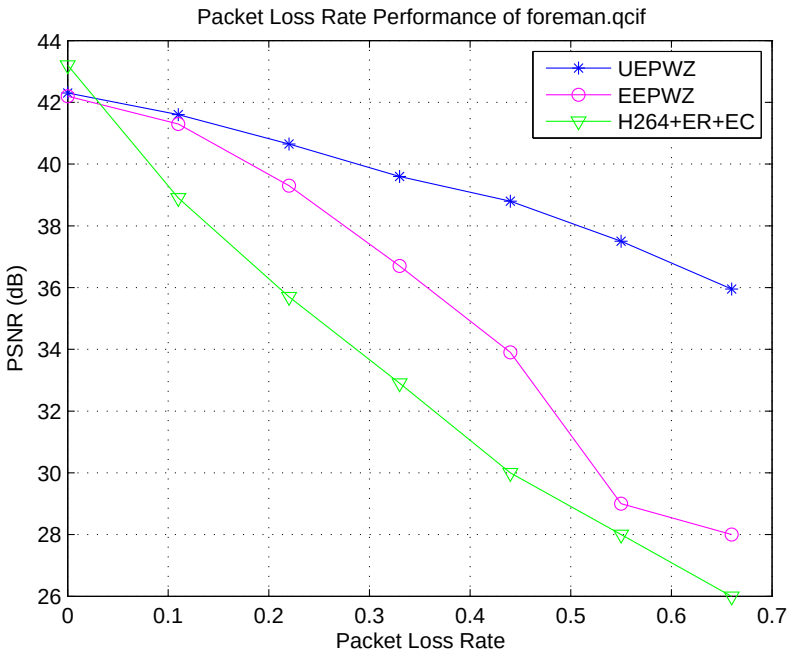

Fig. 2.5. PSNR vs Packets Loss for the *foreman* sequence

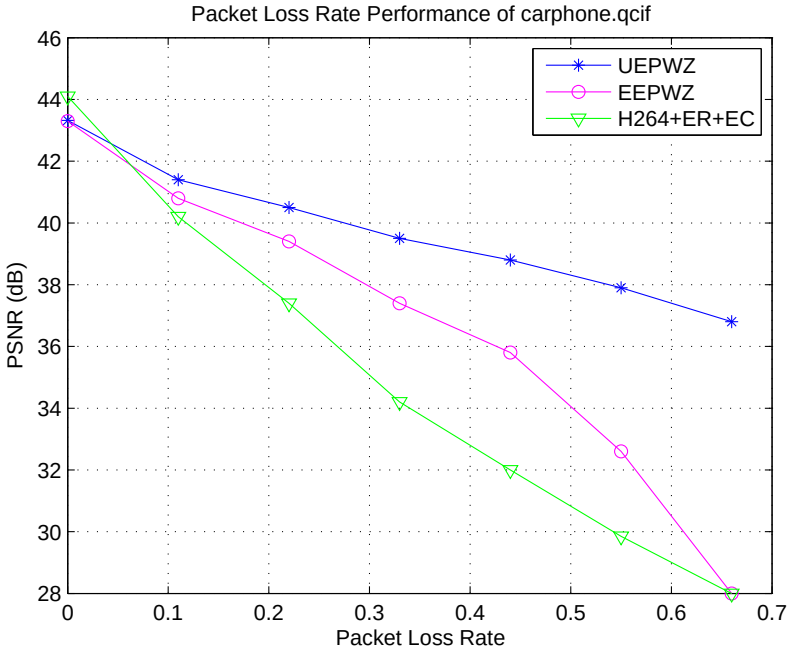

Fig. 2.6. PSNR vs Packets Loss for the *carphone* sequence

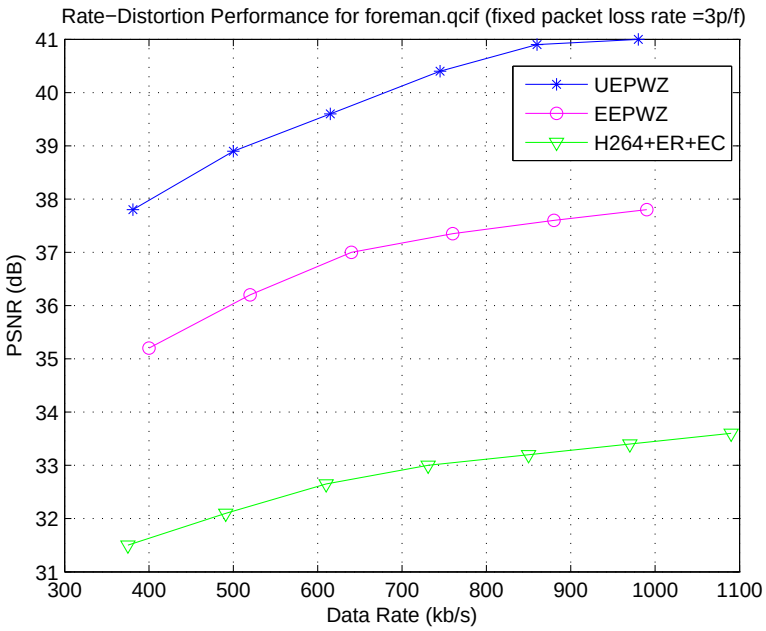

Fig. 2.7. Rate Distortion Performance for *foreman.qcif* sequence at packet loss = 3ppf

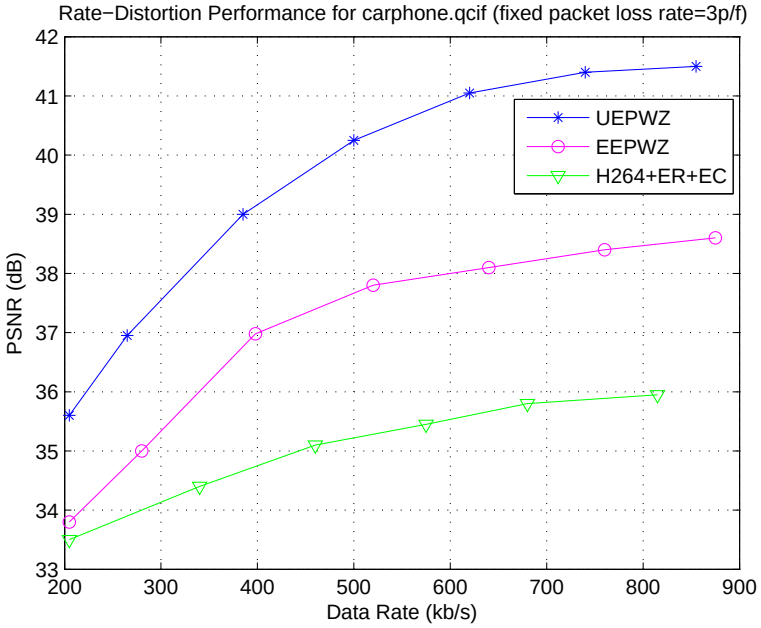

Fig. 2.8. Rate Distortion Performance for *carphone.qcif* sequence at packet loss = 3ppf

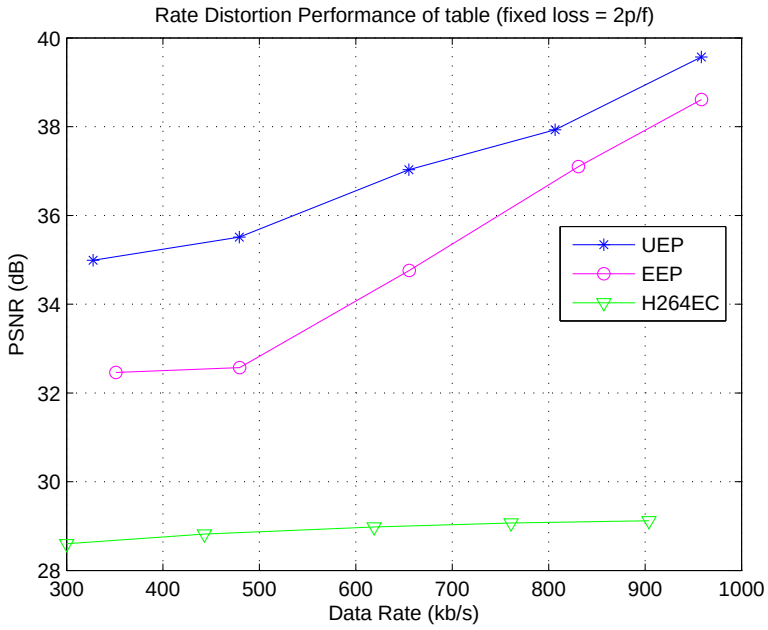

Fig. 2.9. Rate Distortion Performance for *table.qcif* sequence at packet loss = 2ppf

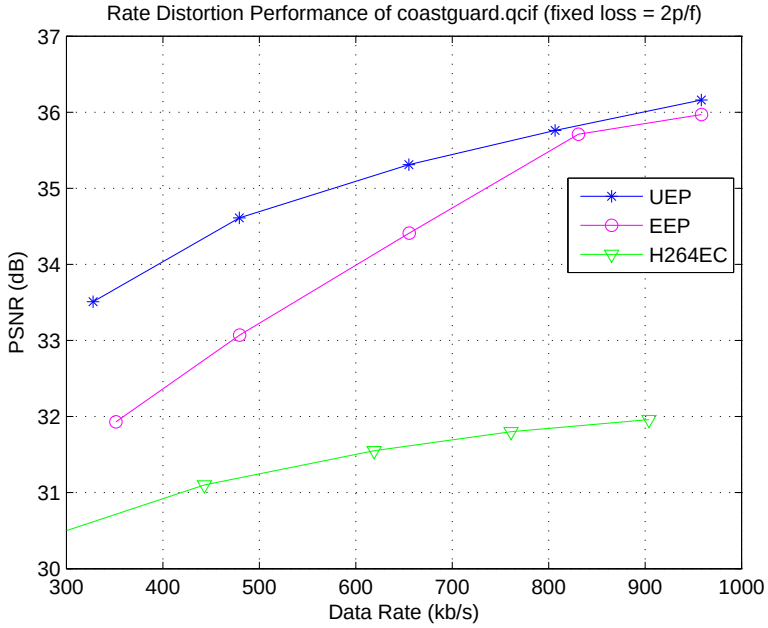

Fig. 2.10. Rate Distortion Performance for *coastguard.qcif* sequence at packet loss = 2ppf

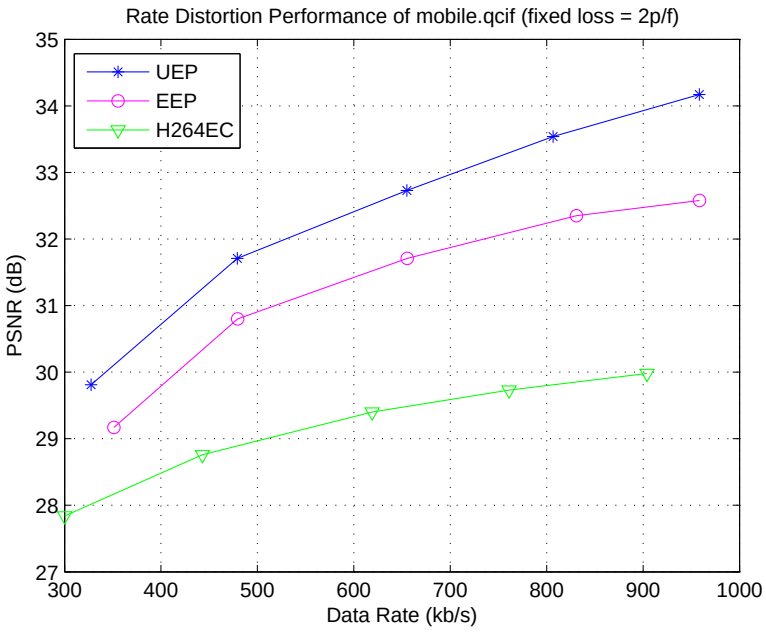

Fig. 2.11. Rate Distortion Performance for *mobile.qcif* sequence at packet loss = 2ppf

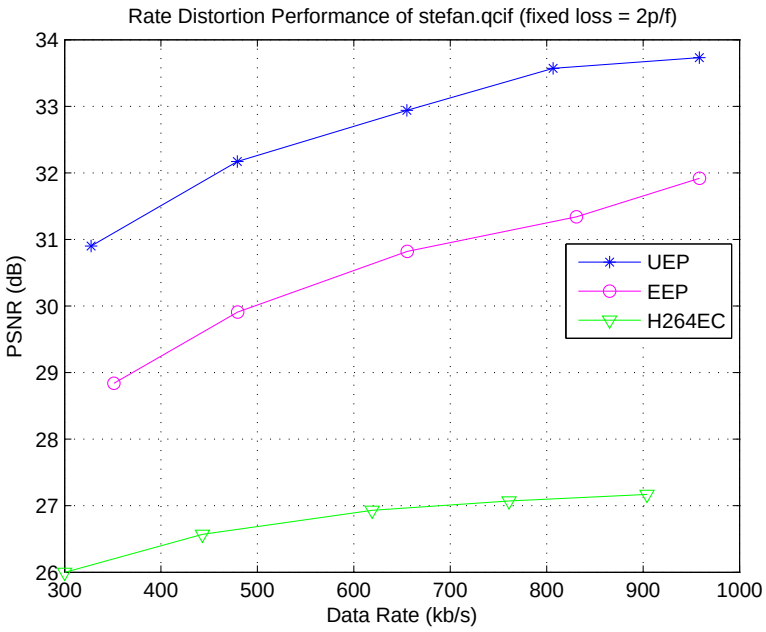

Fig. 2.12. Rate Distortion Performance for *stefan.qcif* sequence at packet loss = 2ppf

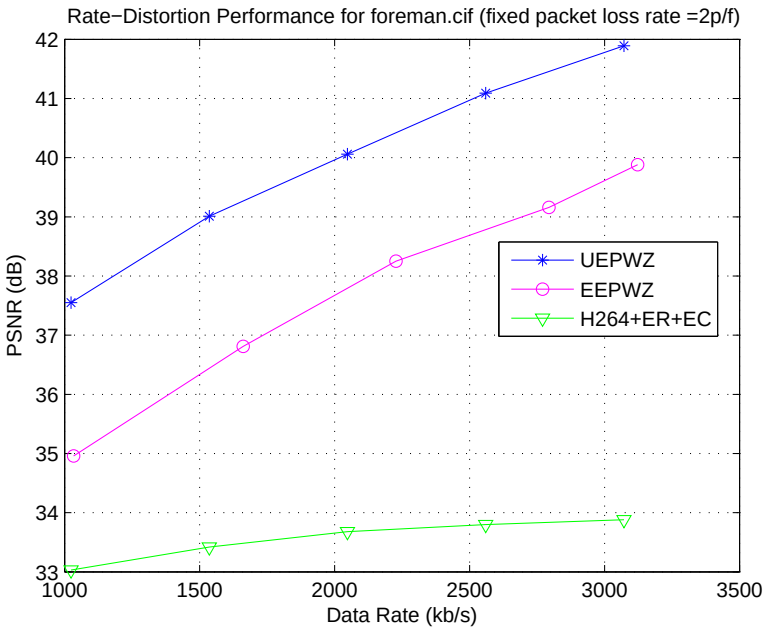

Fig. 2.13. Rate Distortion Performance for *foreman.cif* sequence at packet loss = 2ppf

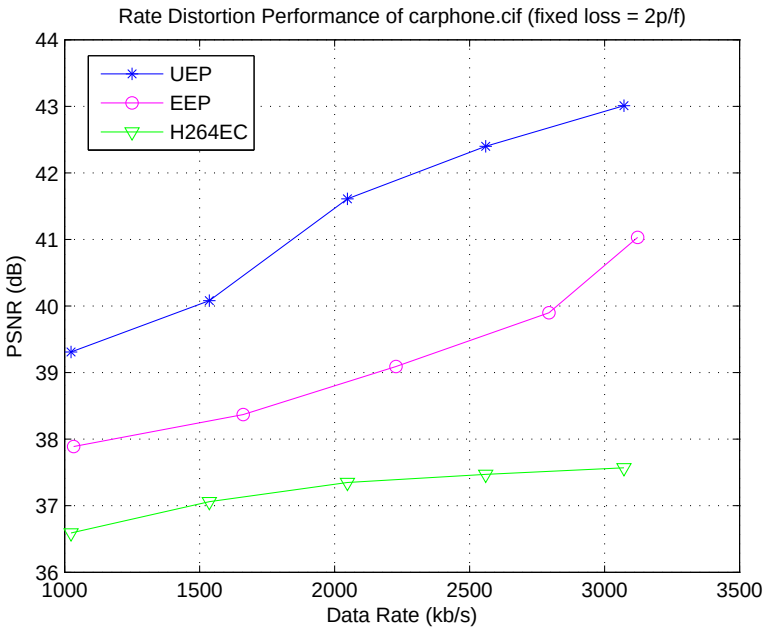

Fig. 2.14. Rate Distortion Performance for *carphone.cif* sequence at packet loss = 2ppf

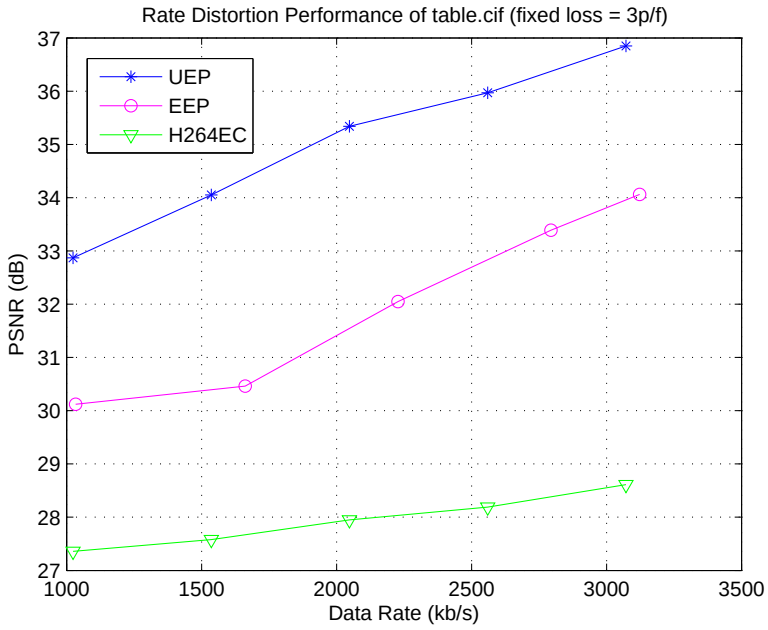

Fig. 2.15. Rate Distortion Performance for *table.cif* sequence at packet loss = 3ppf

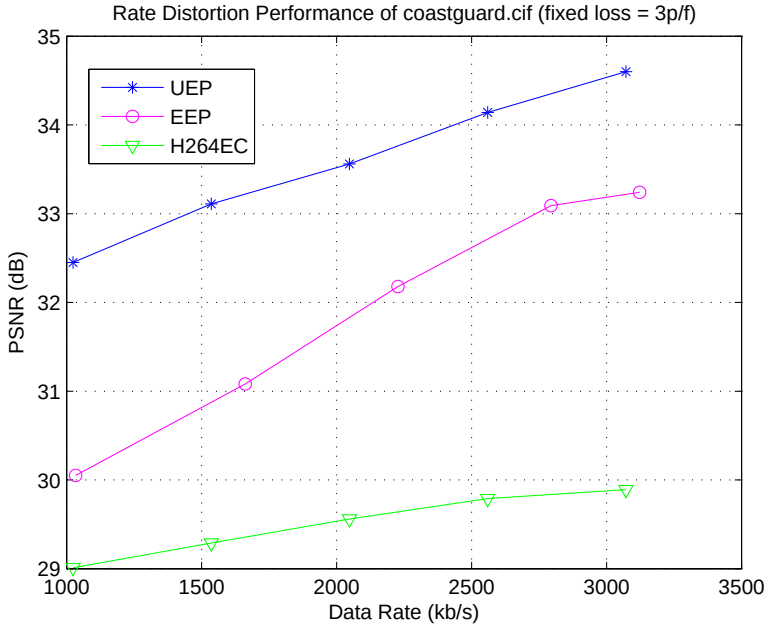

Fig. 2.16. Rate Distortion Performance for *coastguard.cif* sequence at packet loss = 3ppf

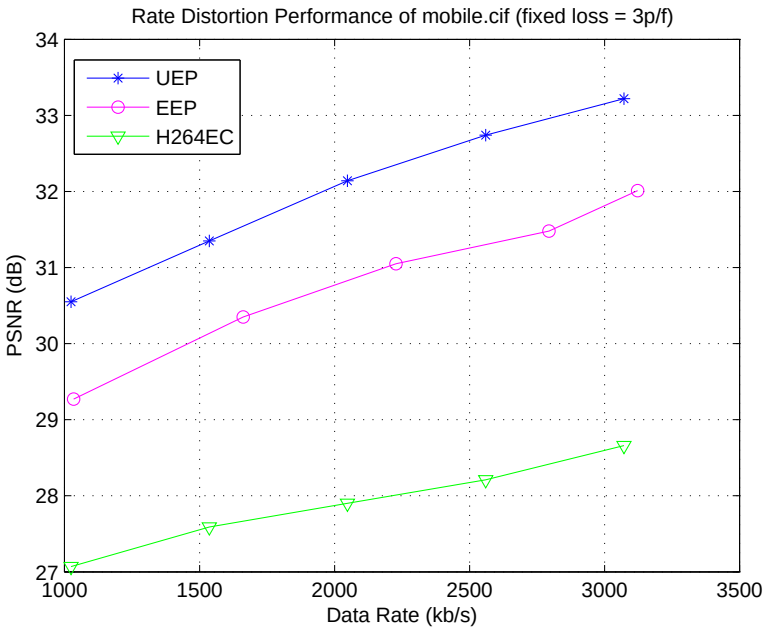

Fig. 2.17. Rate Distortion Performance for *mobile.cif* sequence at packet loss = 3ppf

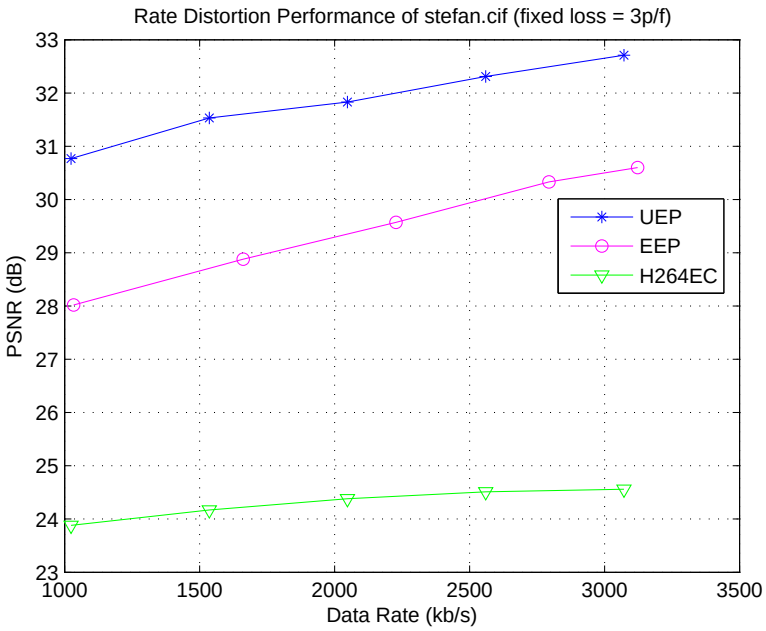

Fig. 2.18. Rate Distortion Performance for *stefan.cif* sequence at packet loss = 3ppf

(a) (b)

Fig. 2.19. Visual depiction of a frame from the *foreman* sequence when 4 packets/frame are lost. (a) Error Concealment performance when the sequence was compressed at a data rate of 479.28 kbps; corresponding PSNR = 24.185dB. (b): SLEP with 4 level, parity data rate WZ rate = 71.28 kbps, H.264/AVC rate = 408 kbps; corresponding PSNR = 27.94dB)

(a) (b)

Fig. 2.20. Visual depiction of a frame from the *foreman* sequence when 4 packets/frame are lost. (a) Equal Error Protection, WZ rate = 80.19 kbps; H.264/AVC rate = 408 kbps; corresponding PSNR = 33.33 dB). (b)Unequal Error Protection, WZ rate = 71.28 kbps, H.264/AVC rate = 408 kbps; corresponding PSNR = 39.55 dB)

(a) (b)

Fig. 2.21. Visual depiction of a frame from the *table tennis* sequence when 5 packets/frame are lost. (a) Error Concealment performance when the sequence was compressed at a data rate of 479.28 kbps; corresponding PSNR=22.83dB. (b): SLEP with 2 level, parity bit rate WZ rate = 71.28 kbps, H.264/AVC rate = 408 kbps; corresponding PSNR=23.82dB)

(a) (b)

Fig. 2.22. Visual depiction of a frame from the *table tennis* sequence when 5 packets/frame are lost. (a): Equal Error Protection,WZ rate = 80.19 kbps; H.264/AVC rate=408 kbps; corresponding PSNR=29.11dB). (b)Unequal Error Protection, WZ rate = 71.28 kbps, H.264/AVC rate=408 kbps; corresponding PSNR=35.08dB)

corrupted frames. Over the quality of competing techniques such as equal error protection, SLEP, and H.264 with error resilience and error concealment. This, however, came at the expense of a small increase in data rate.

3. CONTENT ADAPTIVE UNEQUAL ERROR PROTECTION BASED ON WYNER–ZIV CODING

In this chapter, we improve the performance of unequal error protection technique described in Chapter 2 by adapting the parity data rates of the protected video information to the content of each frame [22]. A content adaptive function is used to evaluate the sum of the absolute difference (SAD) between the reconstructed frame and the predicted frame. Depending on pre-selected thresholds, the parity data rates assigned to the motion information and the transform coefficients are varied for each frame. This results in a more effective and flexible error resilience technique that has an improved performance compared to the original work.

This chapter is organized as follows: Section 3.1 provides a detailed description of the content adaptive unequal error protection (CAUEP) system. Experimental results are provided in Section 3.2, followed by the conclusion in Section 3.3.

3.1 Adaptive Unequal Error Protection

In the previous method UEPWZ, the parity data rates for Turbo coding motion information and the transform coefficients are always set in advance and fixed throughout. However, in a video sequence, different video content in each part of the sequence may require different amounts of protection for the corresponding video data elements. The amount of the motion contained in each frame may change over time, which means part of the video sequence may contain a large amount of motion while some other parts may only contain slow motion content. For this type of video sequences, fixed parity data rate assignment may result in inefficient error protection. When motion content increases in the video sequence, the pre-assigned parity

data rate may become insufficient to correct the errors or it may result in sending redundant parity bits when the motion content decreases in the same video sequence.

The goal of developing an efficient error resilience technique is to make the algorithm applicable to all types of video sequences. Therefore, a mechanism needs to be embedded in the Wyner–Ziv coder to analyze the video content, such as the amount of the motion, in each frame. This improves UEPWZ by adapting the protection levels of different video data element, to the content of each frame.

At the encoder side, a content adaptive function (CAF) is implemented in order to help make decisions regarding the parity data rate allocation for protecting different video data elements. As shown in the system structure diagram (See Figure 3.1), the motion information and the residual frame information are passed to the CAF block. An estimation on the average SAD of the residuals in each frame and the average SAD of the motion vector lengths in each P(B) frame is conducted, both of which indicate the motion content in each frame. The description of the two estimates is given as follows:

$$\overline{SAD_n^{Res}} = \frac{1}{N \times M} \sum_{i=0,j=0}^{i=N,j=M} \left| X_{i,j} - X_{p(i,j)} \right| \tag{3.1}$$

where $X_{i,j}$ denotes the reconstructed pixel value at position (i, j), $X_{p(i,j)}$ is the value of the predicted pixel at position (i, j), and $\overline{SAD_n^{Res}}$ represents the average value of SAD of the n-th frame in the sequence. M and N are the width and the height of each frame.

The SAD of the motion vector length is given by:

$$\overline{SAD_n^{MV}} = \frac{1}{Q \times P} \sum_{i=0,j=0}^{i=Q,j=P} \sqrt{(MV_{x(i,j)})^2 + (MV_{y(i,j)})^2} \tag{3.2}$$

where $MV_{x(i,j)}$ and $MV_{y(i,j)}$ denote the motion vector's x-axis and y-axis values at block position (i, j). Q and P are the coordinates in vertical and horizontal direction of the block index in each frame.

Based on these estimates, the suggested parity data rates assignment are sent to the two Turbo encoders.

The $\overline{SAD^{Res}}$ and $\overline{SAD^{MV}}$ of each frame is compared to three pairs of the predefined thresholds T_1^{res}, T_1^{MV}, T_2^{res}, T_2^{MV} and T_3^{res}, T_3^{MV} (see Table (3.1)), in order to decide the importance level between the motion information and the transform coefficients. The thresholds and the corresponding sets of parity data rates assignments were chosen experimentally.

The experimental results given in section 3.2 demonstrated that by using the parity data rate allocation and the threshold decisions in Table (3.1), content adaptive unequal error protection can provide a better rate distortion performance and visual quality of the decoded video sequences, when compared to our previously proposed unequal error protection UEPWZ. Both techniques outperform the equal error protection case and H.264 with error concealment case as shown in Section (3.2). However, depending on the channel conditions, it may not guarantee perfect recovery of the lost data in all cases. The calculation of the SADs and the comparisons to the thresholds are straight forward, therefore do not add much complexity to the system. The rate allocated to the transmission of the second layer parity data is kept always equal to or less than one fifth of the amount allocated to the primary video information transmission.

In content adaptive unequal error protection (CAUEP), the total data rate allocated for transmitting the parity data is obtained as in Equation3.3. The data rate allocated to error protection is always kept around 14% of the total data rate. Among this 14%, the data rate is allocated again for protecting TC and MI according to the following equations.

$$WZ - DR_{cauep} = \sum_{i=1}^{N} TC_i + \sum_{j=1}^{N} MI_j \tag{3.3}$$

$$= (\frac{1}{N} \sum_{i=1}^{N} PDR_{TC_i}) \times H_{TC} \times W_{TC} \times PN_{symbol} \times FR \tag{3.4}$$

$$+ (\frac{1}{N} \sum_{j=1}^{N} PDR_{MI_j}) \times H_{MI} \times W_{MI} \times PN_{symbol} \times FR \tag{3.5}$$

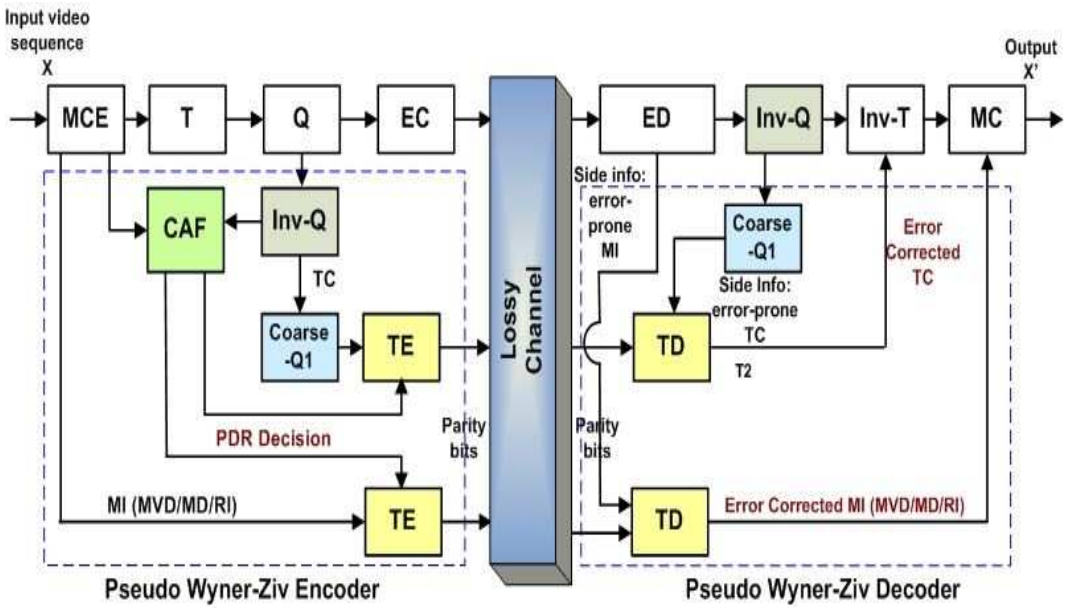

Fig. 3.1. Content Adaptive Unequal Error Protection Based on Wyner-Ziv Coding

Table 3.1
Setting of Parity Data Rate(PBR)

SAD range	PBR assignment
$SAD^{Res} \leq T_1^{Res}; SAD^{MV} \leq T_1^{MV}$	$PDR_{MI} = \frac{1}{4}, PDR_{TC} = 0$
$T_1^{Res} < SAD^{Res} \leq T_2^{Res}; T_1^{MV} < SAD^{MV} \leq T_2^{MV}$	$PDR_{MI} = \frac{1}{2}, PDR_{TC} = 0$
$T_2^{Res} < SAD^{Res} \leq T_3^{Res}; T_2^{MV} < SAD^{MV} \leq T_3^{MV}$	$PDR_{MI} = \frac{1}{2}, PDR_{TC} = \frac{1}{8}$
$SAD^{Res} > T_3^{Res}; SAD^{MV} > T_3^{MV}$	$PBR_{MI} = \frac{1}{2}, PDR_{TC} = \frac{1}{4}$

where H_{TC}, H_{MI} are the height of the transform coefficients and the motion information data packet. W_{TC}, W_{MI} are the width of the transformation coefficients and motion information data packet. PN_{symbol} is representing number of the symbol plane, FR is the current frame rate for transmitting the video packets, and N is the total number of the frames in the video sequence.

Two pairs of the Turbo coders are used for encoding the coarse version of the transform coefficients and the motion information output from the H.264 encoder. The parallel Turbo encoder consists of two same recursive systematic encoders, each

having the following generator matrix: $H(D) = \frac{1+D^2+D^3+D^4}{1+D+D^4}$. The input symbols to the second recursive encoder are interleaved before being passed to it. Puncturing is used to delete some of the parity bits from two encoders, in order to meet a target parity data rate. The turbo decoder utilizes the received parity bits and the side information from the H.264/AVC decoder, to perform the iterative decoding using two BCJR-MAP decoders (see Figure 2.3) [38]. The error corrected information is then sent back to the H.264/AVC decoder to replace the erroneous data.

For I frames in H.264/AVC, there are 9 modes used for predicting a 4×4 block and 4 modes for predicting a 16×16 block from its neighbors. The mode index and the TC are therefore critical for proper frame reconstruction at the decoder.

In the case of P and B frames, since motion vectors belonging to neighboring blocks are highly correlated, motion vector differences (MVD) and TCs are encoded and transmitted. In addition, the H.264/AVC standard allows the encoder the flexibility to choose among reference frames and block modes of 4×4, 4×8, 8×4, 8×8, 8×16, 16×8, and 16×16. In a Wyner-Ziv encoder, the MVD are obtained for each 4×4 block in order to get a fixed block length of symbols, which are then passed onto the turbo encoder. Also this way knowing the mode used for each block is not necessary for decoding the MVD in the Wyner–Ziv decoder. For all three kinds of frames (I, P and B), whether the TCs are protected or not depends on the content adaptive function. The Turbo decoder then utilizes the received parity bits for mode/MVD+RI and/or quantized TC, together with the side information (the channel corrupted mode/MVD+RI and/or TCs) from the H.264/AVC decoder to correct for transmission errors. The Turbo decoder corrects any errors and then transmits the corrected data back to the H.264/AVC decoder.

3.2 Experimental Results

To test the efficiency of the described method, the experiments were carried out using the same $JM10.2$ H.264/AVC reference software used previously on various sequences. As in Chapter 2 all sequences were compressed at 30 frames per second

with a GOP structure *I-P-P-P....* Each QCIF frame in a sequence was divided into 9 slices that were packetized into individual packets. Total number packet per frame is 9packets/frame for QCIF sequence and 18packets/frame for CIF. In another words, each slice of the frame belongs to one packet, so the total packet number is (N-slices) packets/frame. The parity bits of the MI and TCs of each slice were also placed into individual packets. The packets were sent over a channel and subjected to random losses ranging from 1 packet/frame to 8 packets/frame.

The probability of the packet being corrupted and therefore discarded is proportional to the length of the packet [39]. Assume the length of the H.264 data packet is l_h, and the lengths of the Wyner–Ziv MI and TC packets are l_{wm} and l_{wt}, respectively. If the probability of packet loss of H.264 data is r_h, then the probabilities of the packet loss of the MI and TC packets are $r_{wm} = r_h l_{wm}/l_h$ and $r_{wt} = r_h l_{wt}/l_h$, respectively. This is implemented in our packet loss simulation.

We compare the performance of the adaptive unequal error technique (CAUEP) with the original unequal error protection (UEPWZ) and equal error protection (EEP). The performance of CAUEP, UEPWZ and the EEP methods are also compared to that of H264 when the error resilience feature of using slice groups and error concealment using the co-located slice from a previously decoded frame to replace the lost slice are applied. For EEP the MI and the TC are assigned equal parity data rates.

All four methods use the same total transmission data rate, which means, for Wyner–Ziv based error resilience methods, the total data rate is divided into two parts, one part for the primary encoder and the other part for Wyner-Ziv coding. Figure 3.2 through Figure 3.19 depict the rate distortion performance for the QCIF format *foreman*, *carphone mobile*, *coastguard*, *table-tennis*, *stefan* sequences at the loss rates of 1, 2, 3 packets per frame.

It can be seen that CAUEP outperforms UEPWZ by around 0.2-0.4 dB for *foreman* at a packet loss rate of 1 packet per frame, by around 0-0.4 dB at a packet loss rate of 2 packets per frame, and by around 0.2-0.3 dB at a packet loss rate of 3

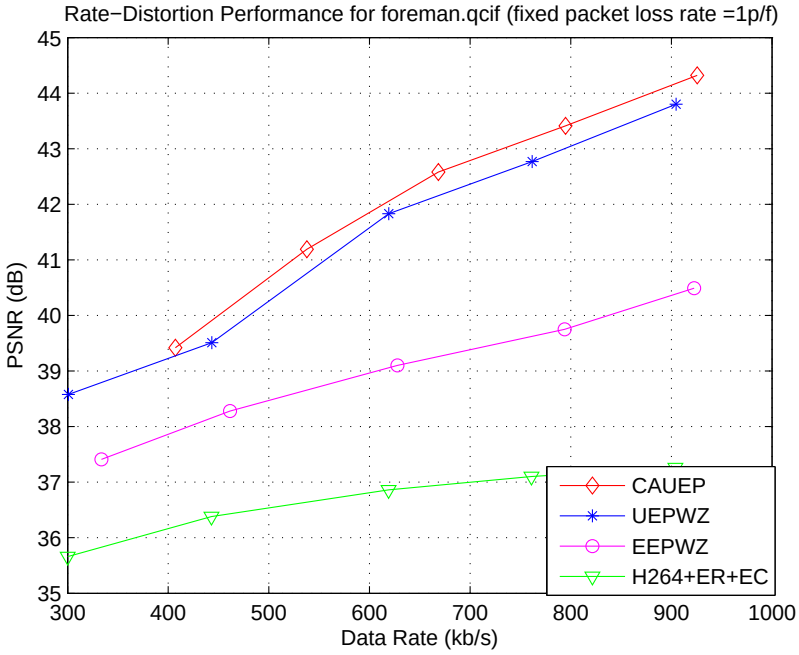

Fig. 3.2. Rate Distortion Performance for the *foreman* sequence

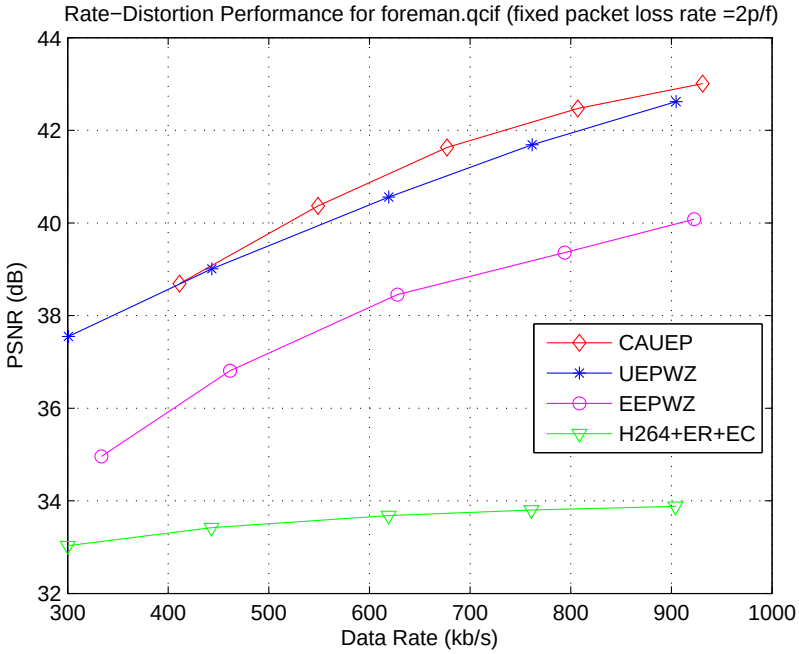

Fig. 3.3. Rate Distortion Performance for the *foreman* sequence

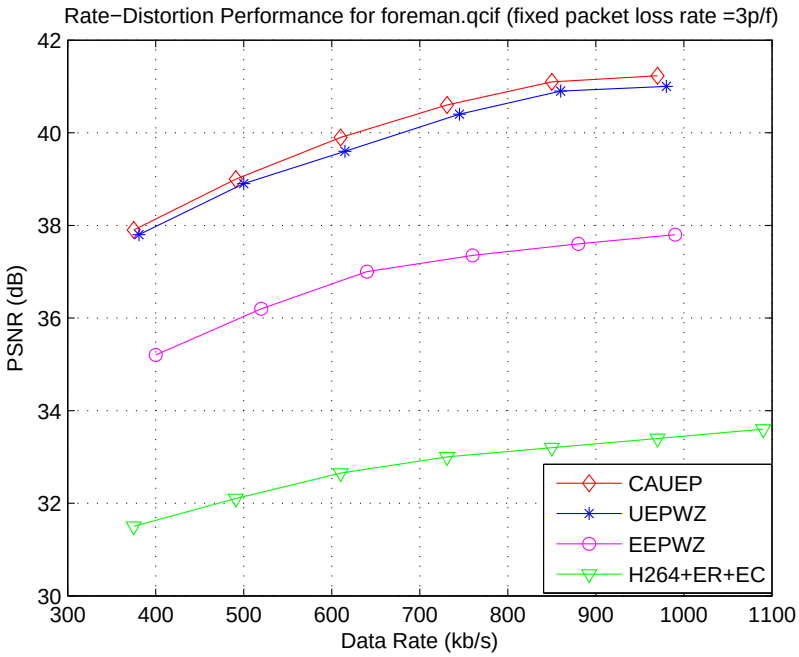

Fig. 3.4. Rate Distortion Performance for the *foreman* sequence

packets per frame. CAUEP outperforms UEPWZ by around 0.1-1 dB for *carphone* at a packet loss rate of 1 packet per frame, by around 0.4-0.6dB at a packet loss rate of 2 packets per frame and around 0.4-0.5dB at a packet loss rate of 3 packets per frame. For *mobile*, CAUEP outperforms UEPWZ by around 0-0.4 dB at a packet loss rate of 1 packet per frame, by around -0.1-0.4 dB at a packet loss rate of 2 packets per frame, by around 0-0.4 dB at a packet loss rate of 3 packets per frame. For *coastguard*, CAUEP outperforms UEPWZ by around 0.1-0.2 dB at a packet loss rate of 1 packet per frame, by around 0.1-0.3 dB for a packet loss rate of 2 packets per frame and by around 0.3-0.5 dB at a packet loss rate of 3 packets per frame. For *table*, CAUEP outperforms UEPWZ by around 0-0.1 dB at a packet loss rate of 1 packet per frame, by around 0.3-1 dB at a packet loss rate of 2 packets per frame and by around 0.3-0.8 dB at a packet loss rate of 3 packets per frame. For *stefan*, CAUEP outperforms UEPWZ by around 0.1-0.3 dB at a packet loss rate of 1 packet per frame, by around 0.2-1.5 dB at a packet loss rate of 2 packets per frame and by around 2-2.4 dB at a packet loss rate of 3 packets per frame.

For *foreman*, most of the frames contain high motion, and therefore require protection of both MI and TC. This corresponds to the third entry in Table(3.1). In this case the overall performance is very close to that of UEPWZ. In the case of *carphone*, the number of frames containing fast motion and the relatively slow motion are evenly distributed. This results in the even distribution of the parity data rate (PBR) over the second and third entries of Table(3.1). This flexible decision makes it outperform UEPWZ. More experiments have been carried out for different sequences and CAUEP always attains a better rate-distortion performance than UEPWZ, especially for sequences that have a mix of both fast and slow motions. As can be noticed, the CAUEP outperforms EEP by around 1.5-2 dB for *foreman* and 2.7-3.5 dB for *carphone*.

Figure 3.20 and Figure 3.21 exhibit the performance of the four strategies when the packet loss ranges from 1 to 7 packets/frame for *foreman* and *carphone*. Again, CAUEP outperforms the other three techniques. However, compared to UEPWZ,

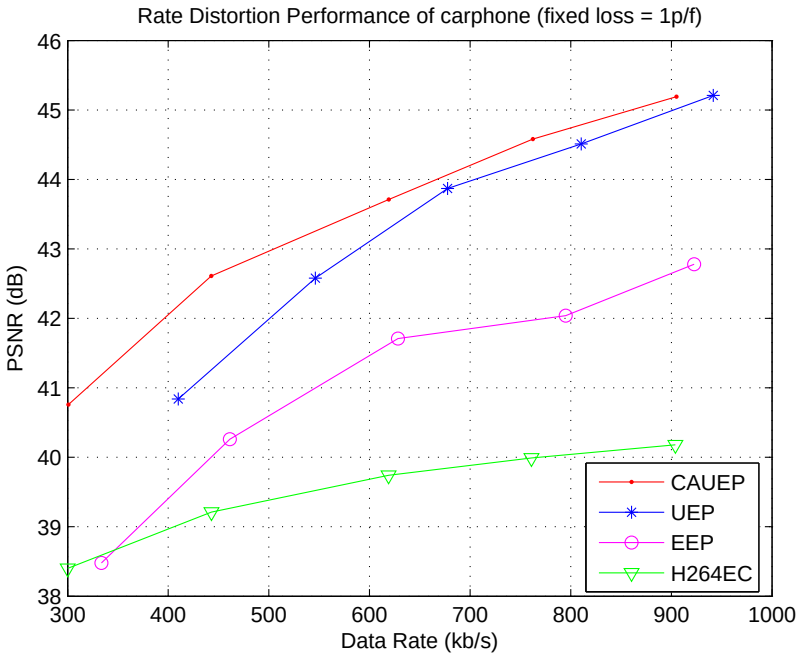

Fig. 3.5. Rate Distortion Performance for the *carphone* sequence

CAUEP gains about 0.2-0.3 dB only for *foreman* and 0.5-1 dB for *carphone*, and its performance converges to that of UEPWZ as packet losses become severe.

Figure 3.22-3.26 depict the rate distortion performance for CIF format sequences. Again it can be seen that CAUEP performs better than UEPWZ because the parity data rate allocation can be adjusted due to the content of each frame and therefore makes the scheme more efficient for error protection.

For visual quality comparison, the decoded frames of *table-tennis* sequence are provided in Figures 3.28- 3.39. As can be seen CAUEP improves the performance of UEPWZ by adapting the parity data rate allocation to the content of the frame.

3.3 Conclusion

In this chapter we presented an error resilient scheme that utilizes a Wyner-Ziv codec to protect important video information produced by an H.264/AVC coder with

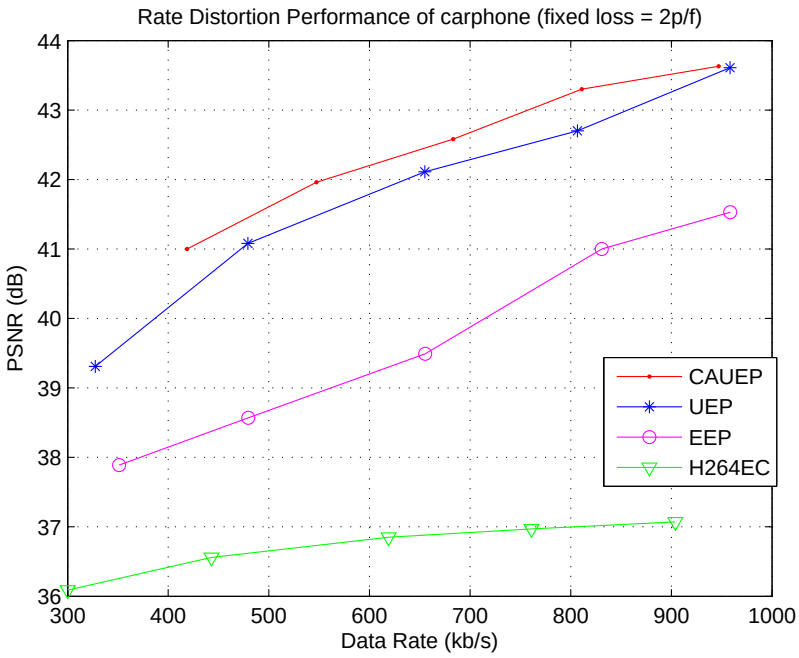

Fig. 3.6. Rate Distortion Performance for the *carphone* sequence

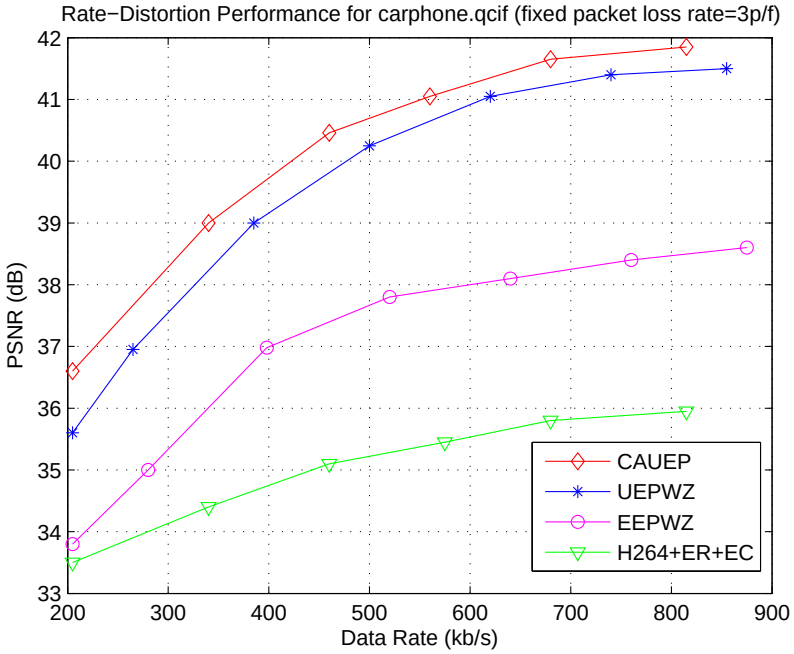

Fig. 3.7. Rate Distortion Performance for the *carphone* sequence

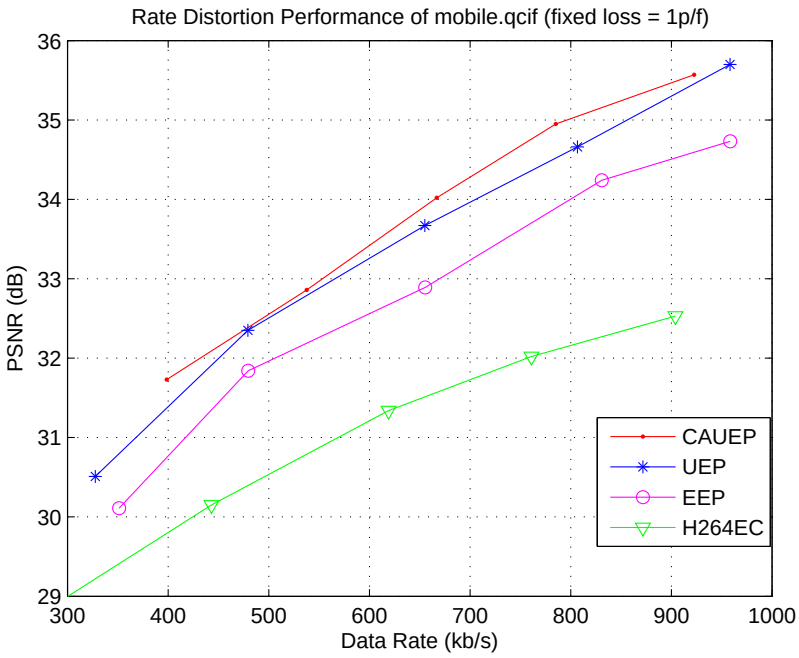

Fig. 3.8. Rate Distortion Performance for the *mobile* sequence

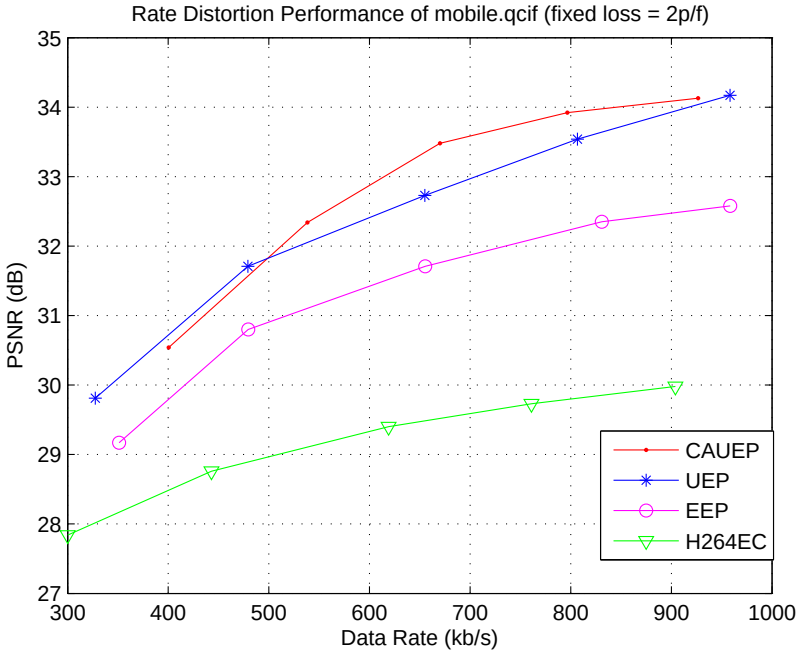

Fig. 3.9. Rate Distortion Performance for the *mobile* sequence

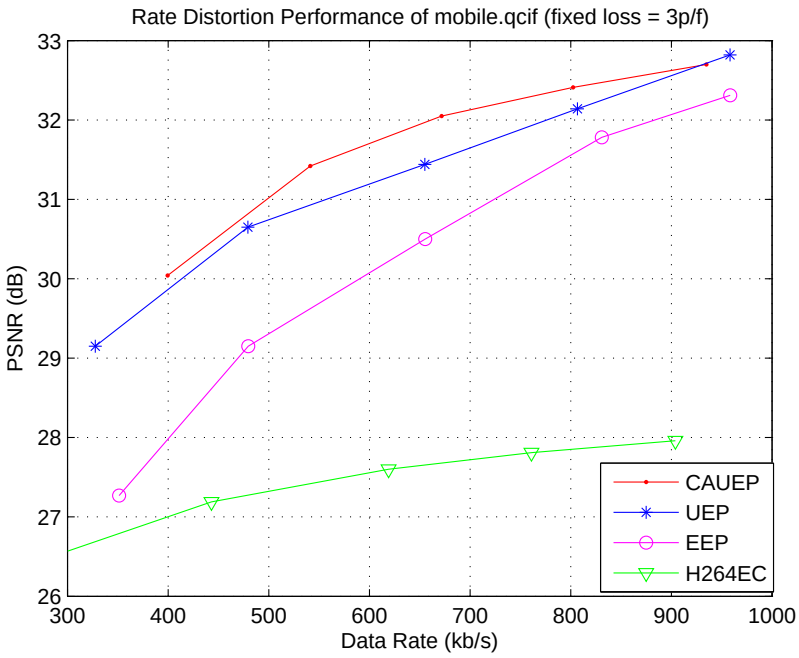

Fig. 3.10. Rate Distortion Performance for the *mobile* sequence

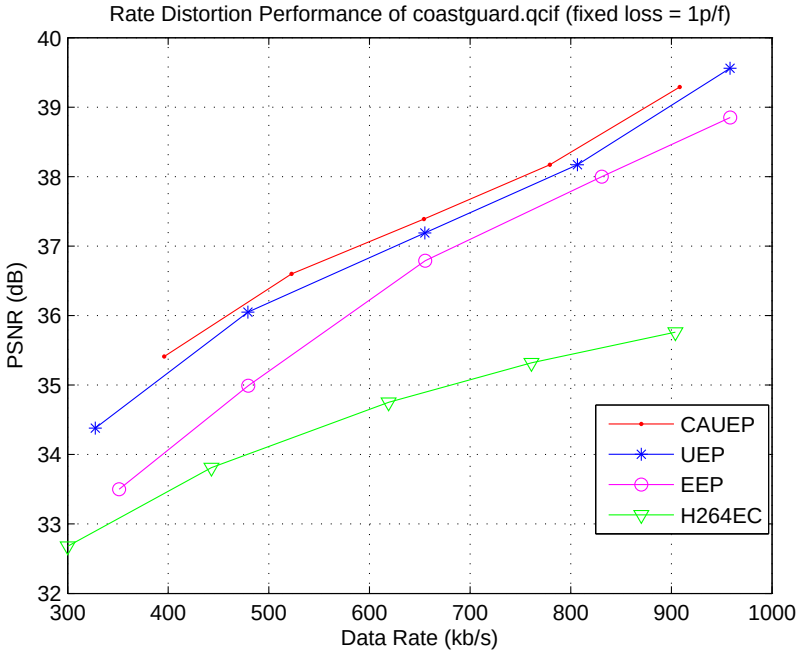

Fig. 3.11. Rate Distortion Performance for the *coastguard* sequence

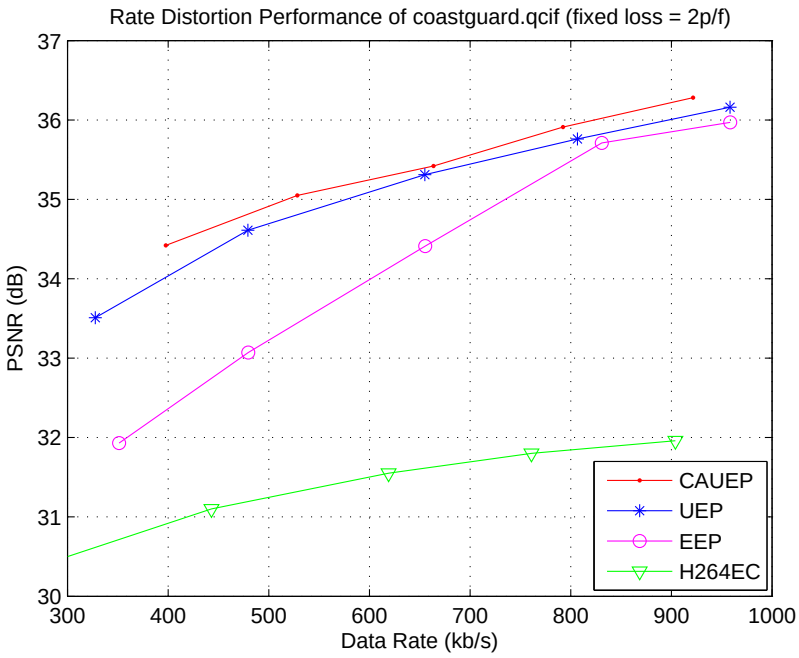

Fig. 3.12. Rate Distortion Performance for the *coastguard* sequence

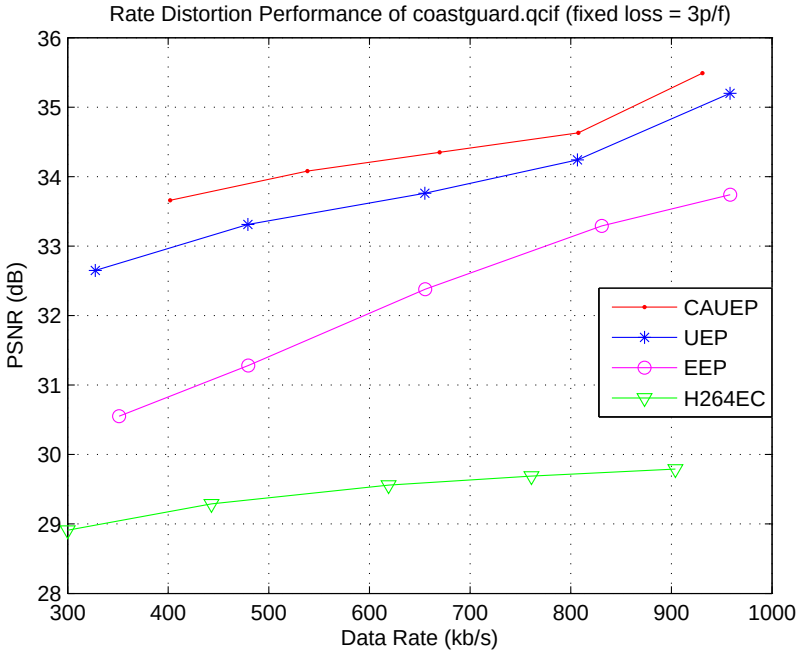

Fig. 3.13. Rate Distortion Performance for the *coastguard* sequence

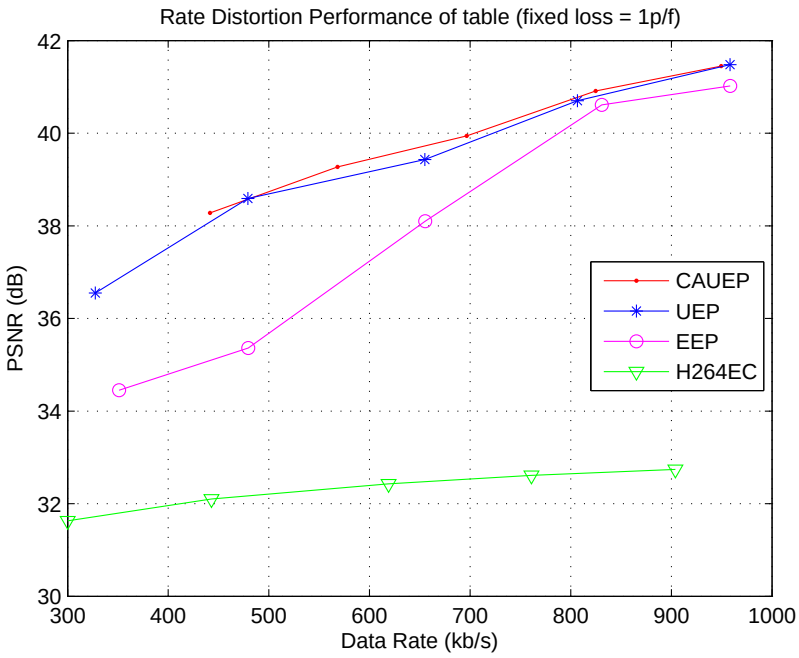

Fig. 3.14. Rate Distortion Performance for the *table* sequence

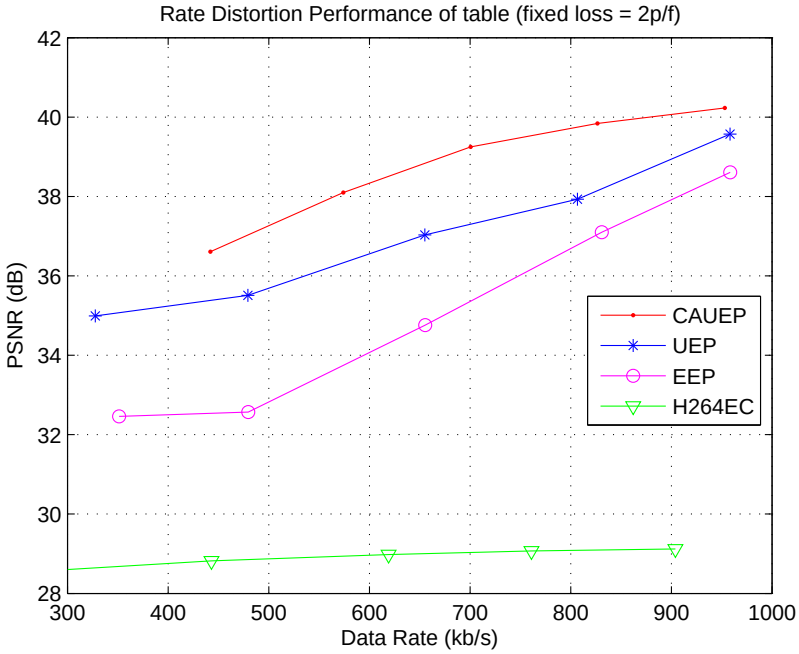

Fig. 3.15. Rate Distortion Performance for the *table* sequence

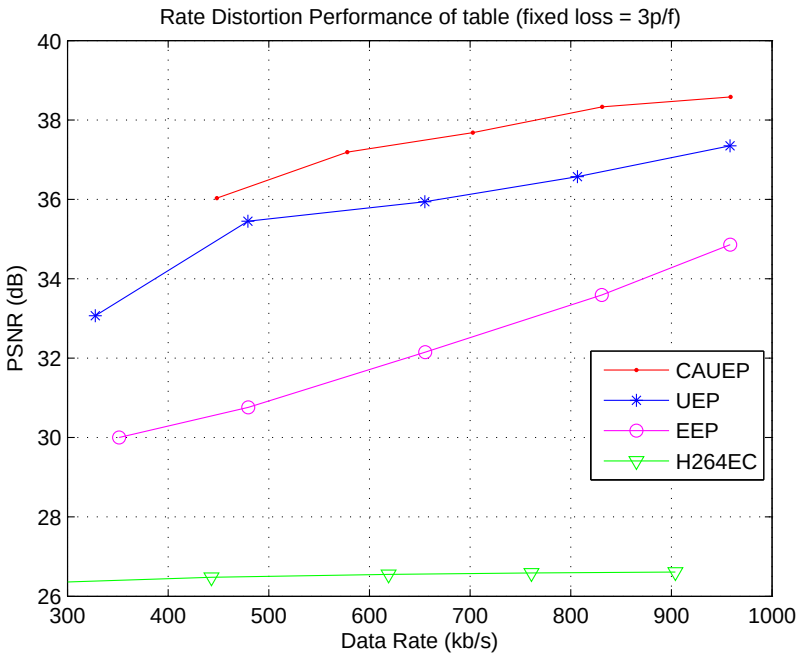

Fig. 3.16. Rate Distortion Performance for the *table* sequence

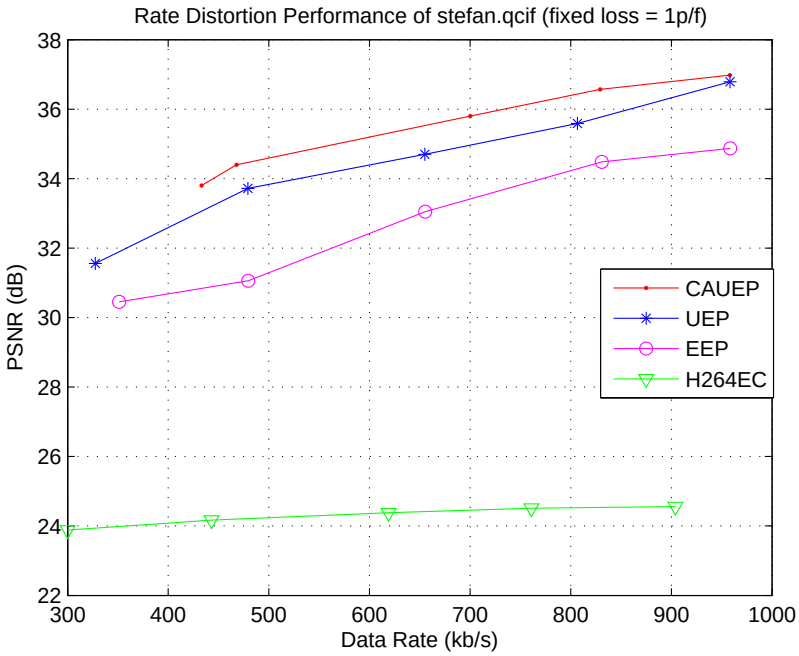

Fig. 3.17. Rate Distortion Performance for the *stefan* sequence

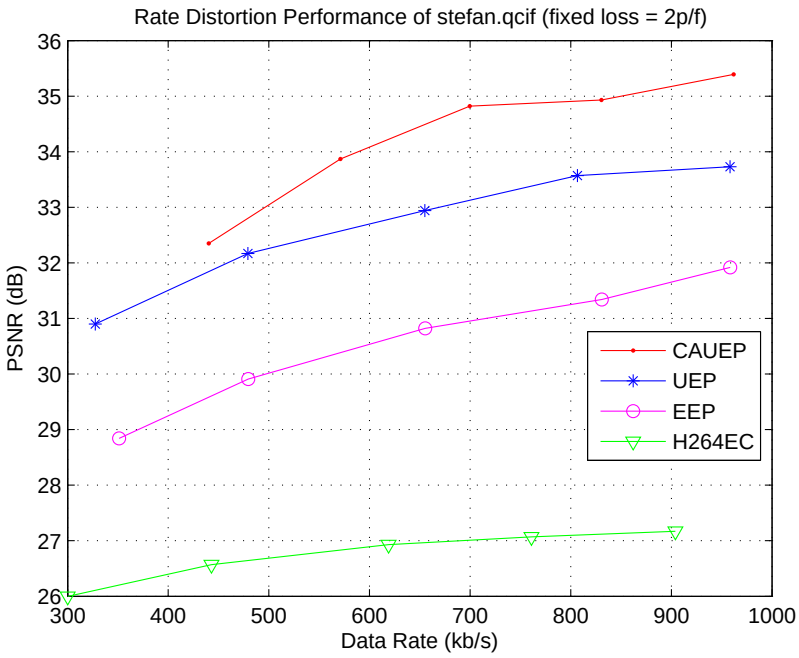

Fig. 3.18. Rate Distortion Performance for the *stefan* sequence

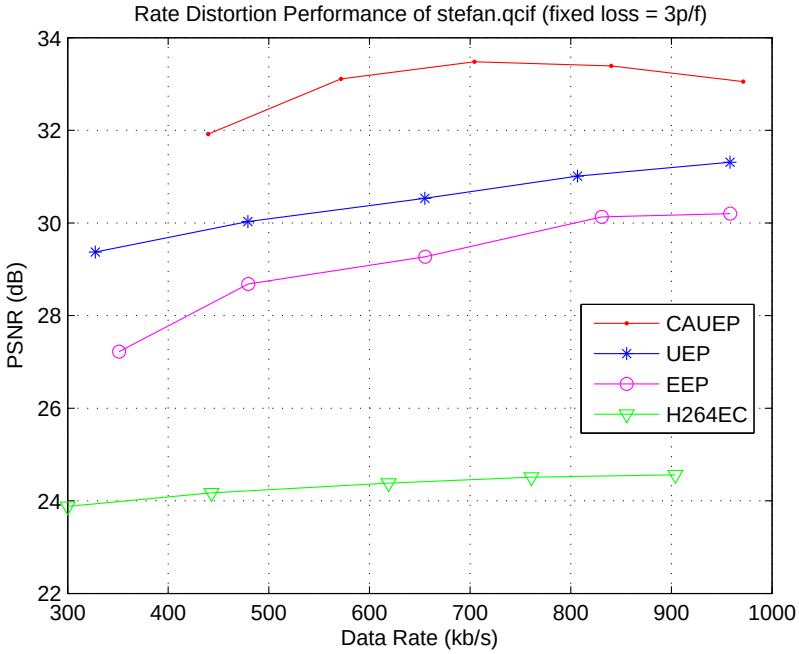

Fig. 3.19. Rate Distortion Performance for the *stefan* sequence

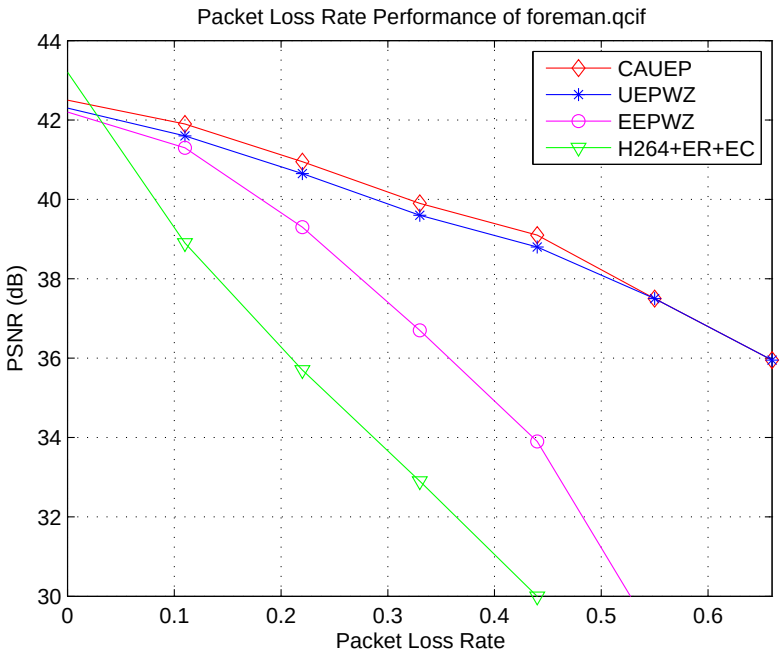

Fig. 3.20. PSNR vs Packet Loss Performance for the *foreman* sequence

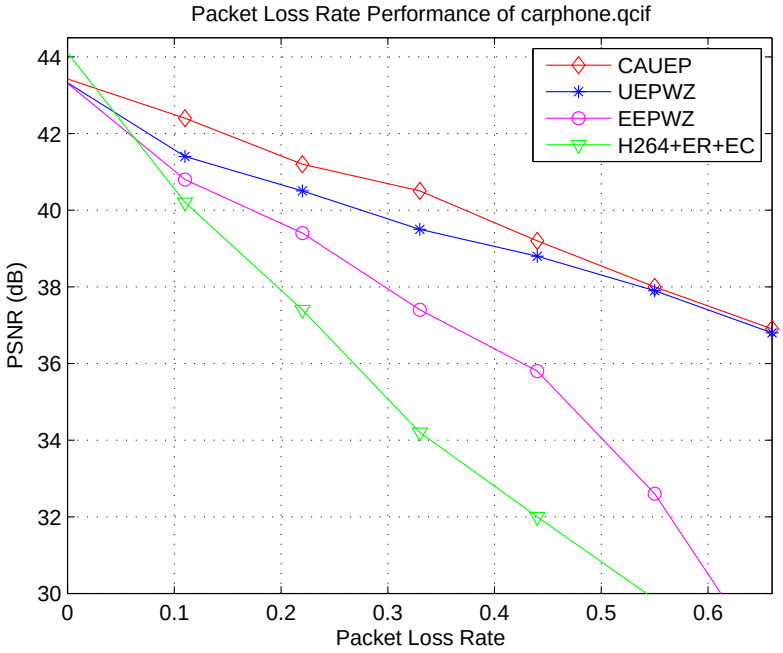

Fig. 3.21. PSNR vs Packet Loss Performance for the *carphone* sequence

unequally assigned parity data rates. The motion information and the transform coefficients are protected independently. The parity data rate settings are updated for each frame according to the frame content. Experimental results show that the proposed scheme significantly improved the quality of corrupted frames and efficiently utilized the available data rate.

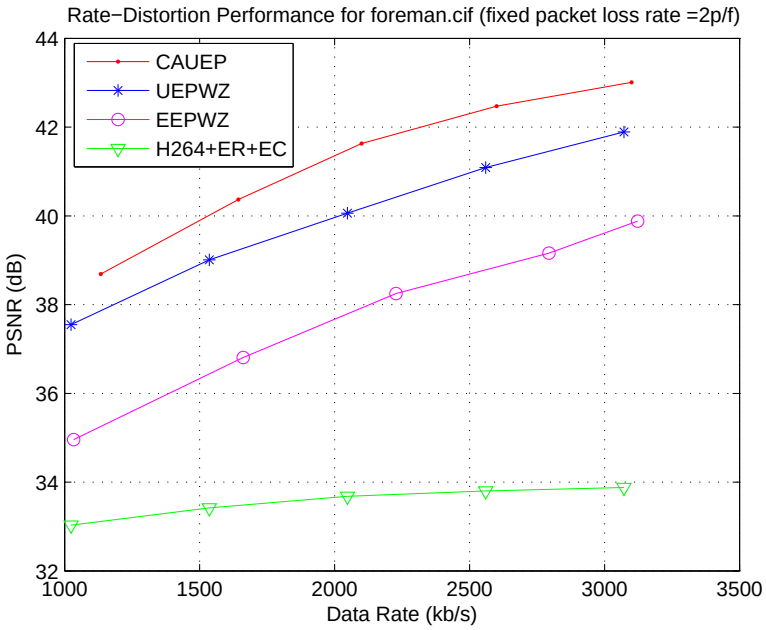

Fig. 3.22. Rate Distortion Performance for the *foreman.cif* sequence

Figure 3.27 shows the ten runs of the PSNR curves of the *foreman.qcif* sequence for the first 100 frames. The channel packet loss is at a dynamic packet loss pattern in which case, the packet loss is randomly varied from 0 to 5 packets per frame. The straight line represents the average PSNR of all the runs.

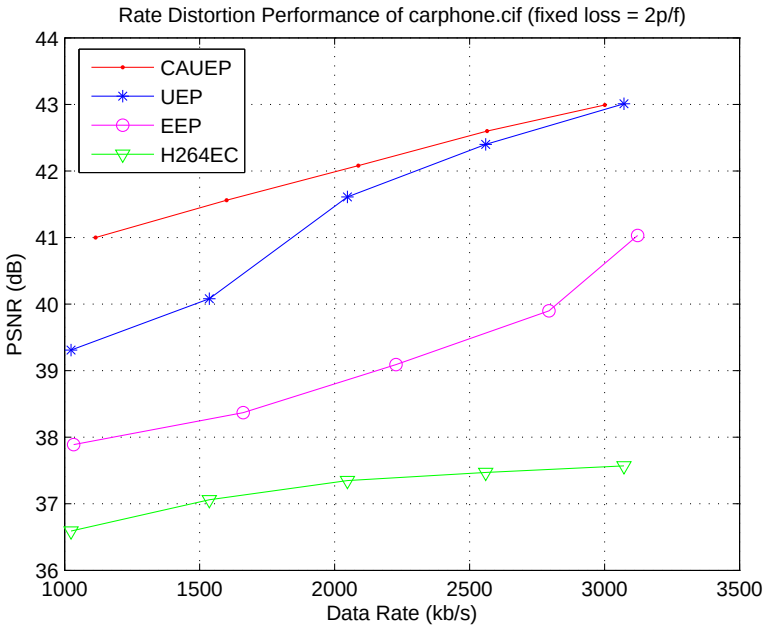

Fig. 3.23. Rate Distortion Performance for the *carphone.cif* sequence

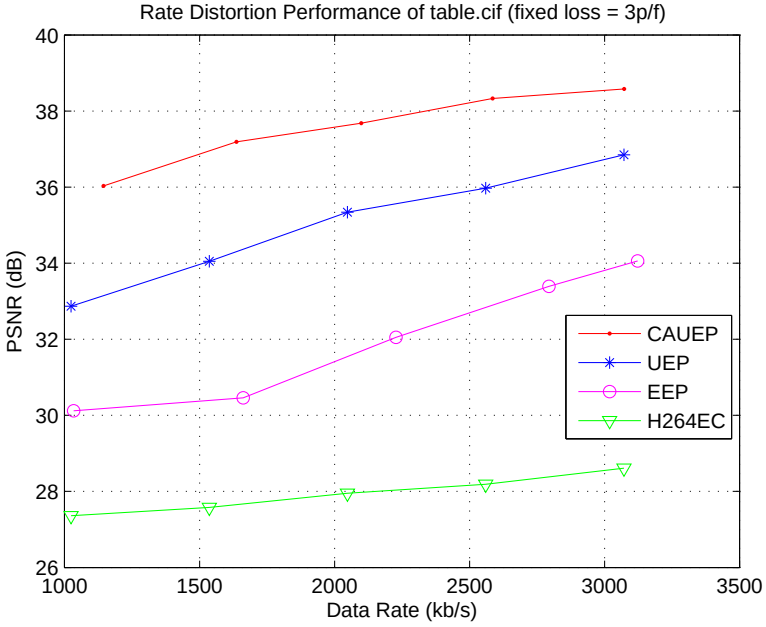

Fig. 3.24. Rate Distortion Performance for the *table.cif* sequence

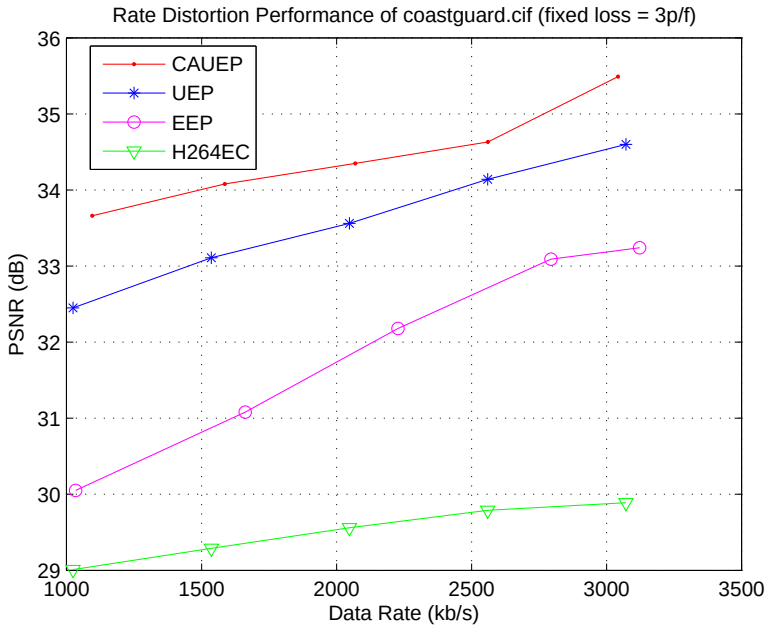

Fig. 3.25. Rate Distortion Performance for the *coastguard.cif* sequence

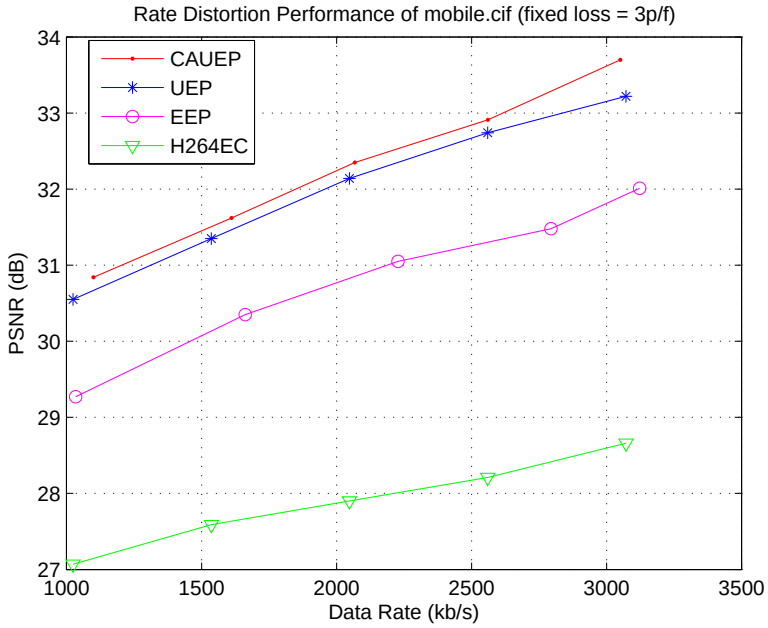

Fig. 3.26. Rate Distortion Performance for the *mobile.cif* sequence

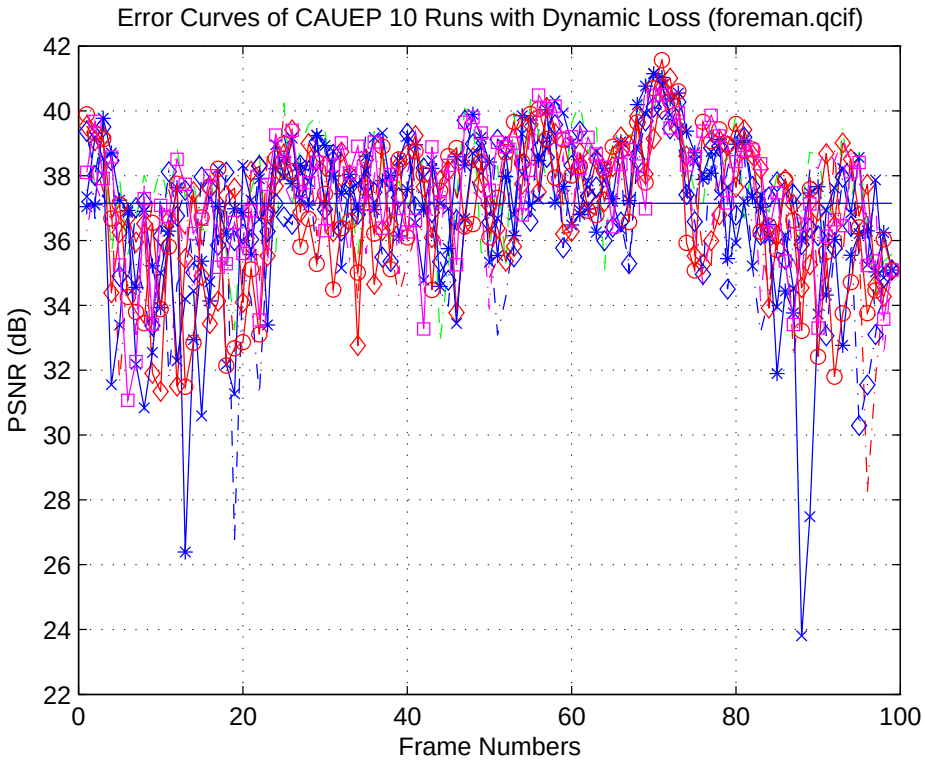

Fig. 3.27. Error Curves of CAUEP 10 Runs with Dynamic Loss (foreman.qcif)

Fig. 3.28. CAUEP: Visual Quality of the 73th frame in *table* sequence at 3 packets per frame loss

Fig. 3.29. UEPWZ: Visual Quality of the 73th frame in *table* sequence at 3 packets per frame loss

Fig. 3.30. CAUEP: Visual Quality of the 67th frame in *table* sequence at 3 packets per frame loss

Fig. 3.31. UEPWZ: Visual Quality of the 67th frame in *table* sequence at 3 packets per frame loss

Fig. 3.32. CAUEP: Visual Quality of the 66th frame in *table* sequence at 3 packets per frame loss

Fig. 3.33. UEPWZ: Visual Quality of the 66th frame in *table* sequence at 3 packets per frame loss

Fig. 3.34. CAUEP: Visual Quality of the 58th frame in *table* sequence at 3 packets per frame loss

Fig. 3.35. UEPWZ: Visual Quality of the 58th frame in *table* sequence at 3 packets per frame loss

Fig. 3.36. CAUEP: Visual Quality of the 57th frame in *table* sequence at 3 packets per frame loss

Fig. 3.37. UEPWZ: Visual Quality of the 57th frame in *table* sequence at 3 packets per frame loss

Fig. 3.38. CAUEP: Visual Quality of the 50th frame in *table* sequence at 3 packets per frame loss

Fig. 3.39. UEPWZ: Visual Quality of the 50th frame in *table* sequence at 3 packets per frame loss

4. FEEDBACK AIDED UNEQUAL ERROR PROTECTION

In this chapter, we present a feedback aided error resilience technique based on the previously proposed unequal error protection method. At the decoder, the current packet loss rates are estimated based on the received data and sent back to the Wyner–Ziv encoder via the real-time transport control protocol (RTCP) feedback mechanism. This is utilized by the Turbo encoder, as a Slepian–Wolf lossless coder inside the Wyner–Ziv codec, to update the parity data rates of the motion information and the transform coefficients, which are still protected independently. At the Wyner–Ziv decoder, the received parity bits together with the side information from the primary decoder are used to decode the corrupted slices. These in turn are sent back to the primary decoder to replace their corrupted counterparts.

It is to be noted that simply increasing the parity slices when the packet loss rate increases is not applicable, since it will exacerbate network congestion [24], [34], [35]. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases the primary data transmission rate should be lowered in order to spare more bits for parity bit transmission. Also, considering the advantage of unequal error protection, when the packet loss rate is high, more data rate should be allocated to motion information parity data rate since motion vectors mostly have higher importance than the transform coefficients.

Whenever feedback is used in a communication system, the amount of delay it incurs needs to be taken in consideration. In our feedback system, an RTCP packet is sent back to the Wyner–Ziv encoder by using the immediate/early RTCP feedback mode, whenever possible. The immediate RTCP feedback mode is the mode that is capable of reporting every loss event immediately back to the receiver. However, the

success of sending the feedback message in such a timely manner also depends on the available bandwidth [36], [37].

The chapter is organized as follows. Section(4.1) details the proposed feedback aided error resilience technique. Experimental results are presented in section(4.2), followed by the conclusion in section (4.3).

4.1 Feedback Aided Unequal Error Protection

Another approach to improve the unequal error protection is to exploit the feedback information of the channel loss rate from the decoder side. The parity data rates assigned for Turbo encoding the protected video information can accordingly be adjusted.

It is to be noted that data networks suffer from two types of transmission errors, namely random bit errors due to noise in the channels and packet losses due to network congestion. When transmitting a data packet, a single uncorrected bit error in the packet header or body may result in the whole packet being discarded [31]-[42]. In the current work, we only consider packet losses, whether due to network congestion or uncorrected bit errors. When using the real-time transport protocol (RTP), determining which packets have been lost can be easily achieved by monitoring the sequence number field in the RTP headers [24], [32]. Therefore, the packet loss rate of each frame can be easily obtained at the decoder.

Figure 4.1 depicts a block diagram of the FBUEP. At the H.264/AVC encoder, each frame is divided into several slices. Both the motion information and the transform coefficients of each slice are sent to the Pseudo Wyner–Ziv encoder to be encoded independently by the two Turbo encoders. As for UEPWZ, the parity data rates allocated to protecting the different elements of the video sequence are assigned independently.

At the decoder, the packet loss rate of each frame is evaluated based on the received video information. It is then sent back to the two Turbo encoders via the RTCP feedback packets. Depending on the channel packet loss rates conveyed, the

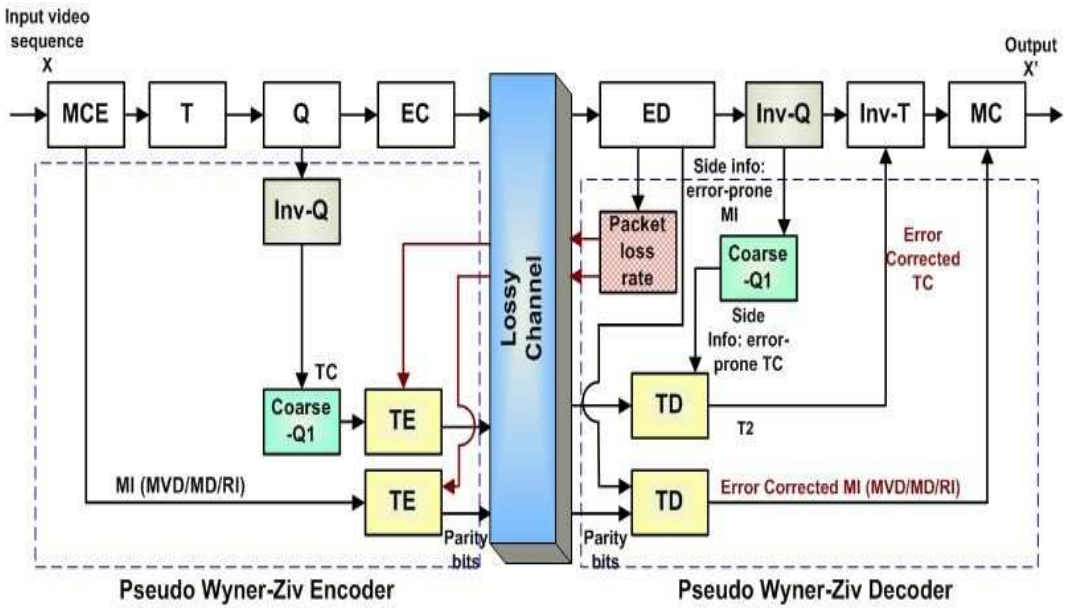

Fig. 4.1. Feedback Aided Unequal Error Protection Based on Wyner–Ziv Coding.

two Turbo encoders adjust the parity data rates for encoding the motion information and the transform coefficients of the current frame.

RTCP Feedback

In the decoder, the channel packet loss rate is obtained based on the received data and sent back to the Pseudo Wyner–Ziv encoder. If the available bandwidth for transmitting the feedback packets is above a certain threshold then an immediate mode RTCP feedback message is sent, otherwise the early feedback RTCP mode is used [43]. The two Turbo encoders update the parity data rates for encoding the motion information and the transform coefficients based upon the received RTCP feedback conveying the packet loss rates. This way the Pseudo Wyner–Ziv encoder attempts to adapt to the decoder's needs, while avoiding blindly sending a large number of parity bits that may not be needed when the packet loss is low or zero. In the case of high channel packet loss rate, the Pseudo Wyner–Ziv encoder enhances

the protection by allocating more data rates to the Turbo encoded data, especially the motion information, while decreasing relatively the data rate used for encoding the main data stream by the H.264/AVC encoder. In this way, the total data rate is kept as a constant so that it won't exacerbate the possible congestion over the network transmission.

According to the RTCP feedback profile that is detailed in [43], when there is sufficient bandwidth, each loss event can be reported by means of a virtually immediate RTCP feedback packet. In the RTCP immediate mode, feedback message can be sent for each frame to the encoder. In our scheme an initial parity data rate value is set at the beginning of transmitting a video. When the channel loss condition changes, the immediate mode RTCP feedback packet sends the latest channel packet loss rate to the Turbo encoders to adjust the parity data rate assignment. If we let N_L denote the average number of loss events to be reported every interval T by a decoder, B the RTCP bandwidth fraction for our decoder, and R the average RTCP packet size, then feedback can be sent via the immediate feedback mode when:

$$N_L \leq \frac{B * T}{R} \tag{4.1}$$

In the RTCP protocol profile [43], it was assumed that 2.5 percent of the the RTP session bandwidth is available for RTCP feedback from the decoder to the encoder. For example, for a 512 kbits/s stream, 12.8 kbits are available for transmitting the RTCP feedback. If we assume an average of 96 bytes (768 bits) per RTCP packet and a frame rate of 15 frames/second, then by Equation4.1, we can conclude that $N_L \leq \frac{12800*\frac{1}{15}}{768} = 1.11$. In this case, the RTCP immediate mode can be used to send one feedback message per frame to the encoder. 2.5% is the maximum that can be allocated for feedback information from the decoder side to the Wyner-Ziv encoder. Depending the channel packet loss rate, the percentage can be different but always under 2.5%.

When $N_L > \frac{B*T}{R}$, the available bandwidth is not sufficient for transmitting a feedback message via the immediate mode. In this case, the early RTCP mode is

turned on. In this mode, the feedback message is scheduled for transmission to the encoder at the earliest possible time, although it can not necessarily react to each packet loss event. In this case, a received feedback message at the encoder side may not reflect the latest channel loss rate. We therefore propose to send an estimate average channel packet loss rate based on packet loss rates of the previous k frames. It gives a better estimate of the recent channel packet loss rate. This scheme is detailed in Section(4.1).

When the Pseudo Wyner–Ziv encoder does not receive feedback regarding the current packet loss rates (the feedback packet got lost during transmitting back to the Turbo encoders or the available bandwidth is not sufficient for immediate mode feedback), the Turbo encoders keep using the last received channel packet loss rate to decide the parity data rates for encoding the motion information and the transform coefficients of the current encoded frame.

Delay Analysis

Delay must be considered when feedback is used. In our system, a RTCP feedback message is transmitted via the immediate mode, if the available RTCP transmission data rate is above the threshold as defined in Equation4.1. Through this mechanism the decoder reports the packet loss rate associated with each received frame to the encoder. The Pseduo Wyner–Ziv encoder then utilizes this information to select the parity data rates for encoding the motion information and the transform coefficients of the current encoded frame.

In early feedback mode, rather than sending feedback on a frame by frame basis, we propose to send the feedback packets to the Pseudo Wyner–Ziv encoder every k frames ($k = 1, 2, ..., n$). The feedback in this case is the average channel packet loss rate (L_k^m) evaluated based on the history of the received video information of the past k frames, as given in Equation4.2. m represents the m-th set of the k frames received at the decoder. In this equation $S_{i,j}$ is a counter counting the number of the error corrupted slices in the i-th received frame. i is counted in terms of every k

frames ($i = 0, ..., k$). j is the index of the received slice and each frame is assumed to be partitioned into n slices. The parity data rates assignment, for Turbo encoding the motion information and the transform coefficients of the next k frames, is then updated once every k frames and therefore has higher resilience to the delay problem.

$$L_k^m = \frac{1}{K} \sum_{i=1}^{k} \sum_{j=1}^{n} S_{(i,j)} \tag{4.2}$$

$$S_{(i,j)} = \begin{cases} 0 & \text{the error free packet is received;} \\ 1 & \text{the error corrupted packet is received.} \end{cases}$$

Furthermore, in the frame by frame based feedback strategy, if the packet loss rate of the current decoded frame is the same as the previous frame's, no feedback message needs to be sent back to the encoder. In the same way, if the average channel packet loss rate of the current received k frames (L_k^m) is equal to the average packet loss rate of the past k frames ($L_k^{(m-1)}$), no feedback is needed to be sent back to the Pseudo Wyner–Ziv encoder. In other words, the feedback message is only sent back to the encoder when the packet loss rate is changed. Therefore, there are three scenarios when no feedback is received by the Turbo encoders. One is that the channel packet loss rate is kept as a constant at the moment. Another case is that the feedback packet got lost during transmitting back to the Turbo encoders. The third case is that the available bandwidth is not sufficient for immediate mode feedback. Accordingly, the Turbo encoders only update the parity data rates for encoding the motion information and the transform coefficients when they received the updated feedback regarding the latest packet loss rate.

Data Rate Assignment

When packet loss rates increase, simply increasing the parity data rates for Turbo encoding the motion information and the transform coefficients while keeping the same data rate for the primary video data coding would only exacerbate channel

congestion [24]. A better way would be to reduce the data rate allocated to the primary video data transmission slightly and correspondingly increase the data rate allocated to the transmission of parity bits, so that the total transmission data rate at any packet loss rate is kept constant.

Furthermore, more efficient use of the data rate can be achieved by assigning different protection levels to the motion data and the transform coefficients in the Pseudo Wyner–Ziv encoder at different channel packet loss rate. For example, when the packet loss rate is very high and the available data rate for Psudo Wyner–Ziv coding is not enough for transmitting the required amount of the total parity bits, it is always more effective to allocate enough data rate for protecting the parity bits of the motion information first since the accuracy of the motion information normally plays a more important role in decoding the video sequence compared to the transform coefficients.

In our scheme, the parity data rates assigned to the motion information and the transform coefficients were evaluated experimentally. The parity data rates settings at different range of channel packet loss rate were tested by extensive experiments on different video sequences. The experiment results showed that the enough lost information can be corrected for reconstructing a visually good quality decoded frames (See Table(4.1)).

In the FBUEP, the total data rate allocated for transmitting the parity data is calculated as in Equation4.3

$$WZ - DR_{fbuep} = \sum_{i=1}^{N} TC_i + \sum_{j=1}^{N} MVD_j \tag{4.3}$$

$$= (\frac{1}{N}\sum_{i=1}^{N} PDR_{TC_i}) \times H_{TC} \times W_{TC} \times PN_{symbol} \times FR \tag{4.4}$$

$$+ (\frac{1}{N}\sum_{j=1}^{N} PDR_{MVD_j}) \times H_{MV} \times W_{MV} \times PN_{symbol} \times FR \tag{4.5}$$

Where H_{TC}, H_{MVD} is the height of the TC and MVD data packet. The W_{TC}, W_{MVD} is the width of the TC and MVD data packet. PN_{symbol} is the number of

Table 4.1
Setting of Parity Data Rate(PDR) for FBUEP Method

Packet Loss Rate	**Parity Data Rate Assignment**
$0 < N \leq 11\%$	$PDR_{MI} = \frac{4}{16}$, $PDR_{TC} = \frac{2}{16}$
$11\% < N \leq 22\%$	$PDR_{MI} = \frac{6}{16}$, $PDR_{TC} = \frac{2}{16}$
$22\% < N \leq 33\%$	$PDR_{MI} = \frac{7}{16}$, $PDR_{TC} = \frac{3}{16}$
$33\% < N \leq 44\%$	$PDR_{MI} = \frac{8}{16}$, $PDR_{TC} = \frac{4}{16}$
$44\% < N \leq 55\%$	$PDR_{MI} = \frac{10}{16}$, $PDR_{TC} = \frac{4}{16}$
$55\% < N$	$PDR_{MI} = \frac{12}{16}$, $PDR_{TC} = \frac{8}{16}$

the symbol plane and the FR is the current frame rate for transmitting the video packets. N is the total number of the frames in the video sequence. In FBUEP technique, parity data rate assigned for protecting TC and the MVD are different for each frame. Therefore the parity data rate assigned for TC and MVD are summed up for all frames.

4.2 Experimental Results

To evaluate the proposed techniques, experiments were carried out using the $JM10.2$ H.264/AVC reference software. The frame rate for each sequence was set at 15 frames per second with an *I-P-P-P...* GOP structure. In our experiment, each QCIF frame is divided into 9 slices. The primary encoded video data output from the H.264 encoder are packetized into 9 packets per frame, each containing the video information of one slice. The Turbo encoded parity bits of the motion information and the transform coefficients corresponding to each slice are also sent in separate packets. All three types of the packets are subjected to random losses over the transmission channel. We did not attempt to make all packets the same size. Since the packets containing the parity bits of the motion information or the transform coefficients are much smaller in size comparing to the H.264 packets, the possibility of getting lost

over a wireless network transmission is therefore much smaller. All the experiments results were averaged over 30 lossy channel transmission realizations.

Data networks suffer from two types of transmission errors: random bit error and packet drop. In our experiments, we only consider the case of packet erasures, whether due to network congestion or uncorrected bit errors. Lower the total data rate to reduce the network congestion is a realistic solution when packet loss is very high. However, since our main application is for video streaming over wireless networks in which case the packet loss situation is more complicated, we didn't consider it in our current experiments. It is to be noted that simply increasing the parity bits when the packet loss rate increases is not applicable, since it will exacerbate network congestion. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases, the primary data transmission rate should be lowered in order to spare more bits for parity bits transmission.

In our experiments, channel packet loss is simulated by using uniform random number generators. Our algorithm focuses on wireless network application in which case severe packet loss could happen. In the case of wireless network transmission, the probability that the packet arrives in error is approximately proportional to its length [44]. Assume the length of the H.264 data packet is l_h, and the lengths of the parity bits packets containing the motion information and the transform coefficients are l_{wm} and l_{wt}, respectively. If the probability of the packet loss of H.264 data is r_h, then the probabilities of the packet loss of the motion information and the transform coefficients packets are $r_{wm} = r_h l_{wm}/l_h$ and $r_{wt} = r_h l_{wt}/l_h$, respectively. This is implemented in our packet loss simulation.

For the UEPWZ technique, the protection level provided for the motion information and the transform coefficients was fixed for the entire video sequence, and was not changed to cater for different video content or varied channel packet losses. For CAUEP, the protection level of the motion information and the transform coefficients is always adapted to the content of each frame, that is the parity data rates allocated to the protected video information in the Pseudo Wyner–Ziv coder were adjusted

according to the content of each frame. The rate, however, was not adjusted to deal with variable channel losses.

In the case of FBUEP, the packet loss rates were evaluated by the H.264/AVC decoder and sent back to the Pseudo Wyner–Ziv encoder via RTCP feedback packets. In the case that there is not enough bandwidth for immediate mode feedback transmission, feedback message is transmitted to the encoder for every k frames only ($k > 1$).

In the event that the RTCP feedback packets got lost during transmission, the Turbo encoders will keep using the last received channel loss rate to decide the parity data rate allocation for the next k frames. It is noted that the protection level in the Pseudo Wyner–Ziv encoder is not adjusted according to the frame content in the case of FBUEP. In all three proposed schemes, the error concealment is not applied on the H.264 decoded data before being passed to the Wyner-Ziv decoder.

In our Wyner-Ziv based schemes, different parity data rate settings have been tested extensively for different types of video sequences. For the tested sequences, the final decision on the parity data rate assignments that are given in the paper can achieve a better rate distortion performance, the visual quality of the decoded frames and the overall data rate usage comparing to other values of the parity date rates.

Figure 4.14 shows the ten runs of the PSNR curves of the *foreman.qcif* sequence for the first 100 frames. The channel packet loss is at a dynamic packet loss pattern in which case, the packet loss is randomly varied from 0 to 5 packets per frame. The straight line represents the average PSNR of all the runs.

In general, channel conditions change over time, resulting in variable packet loss rates. In the following experiments, the channel packet loss rates were varied during the transmission time of the video sequences. In our simulation, the lowest packet loss rate is 0 while the highest possible packet loss rate is 55%. The mean of the overall channel packet loss is at 23.2%. The parity data rates allocated to the motion information and the transform coefficients, in the case of FBEUP, are shown in Table(4.1).

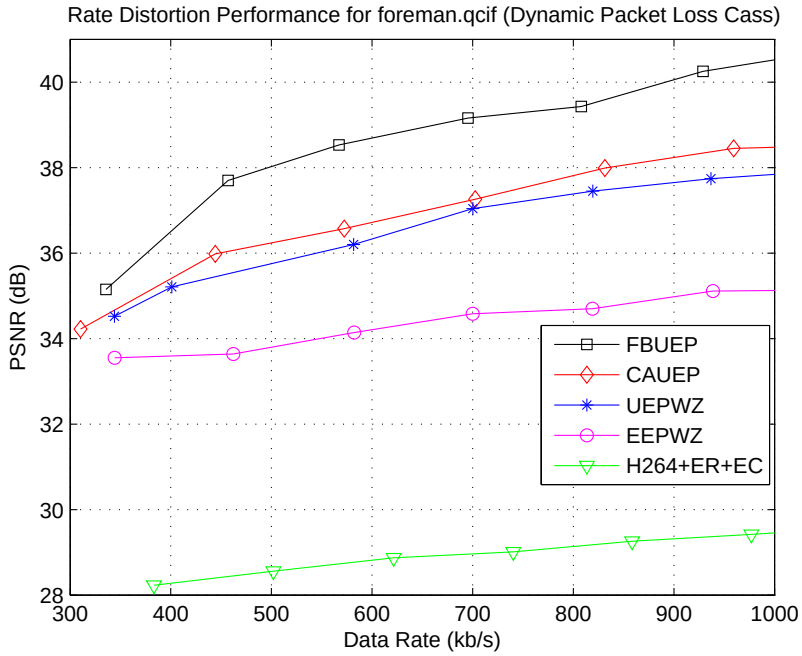

Fig. 4.2. Rate Distortion Performance (foreman-qcif) (Dynamic Packet Loss Case)

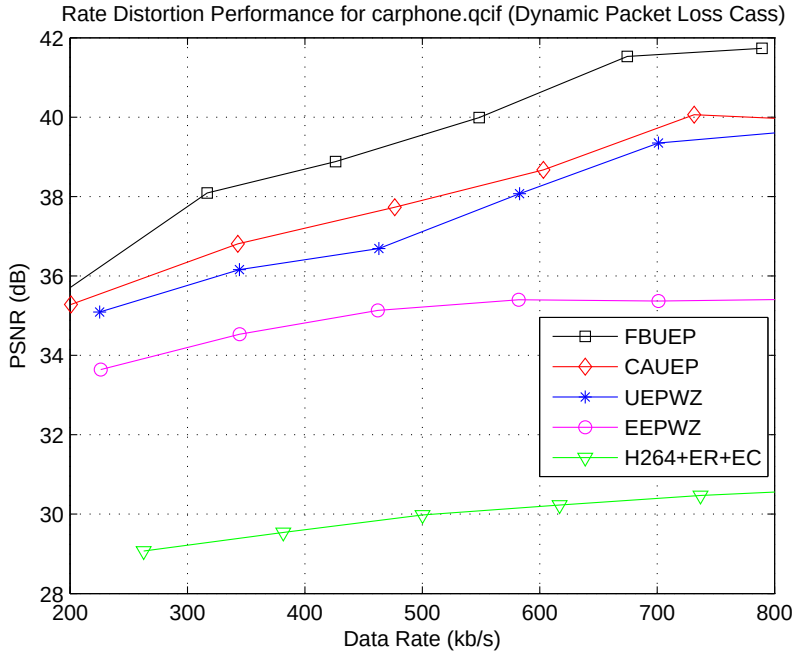

Fig. 4.3. Rate Distortion Performance (carphone-qcif) (Dynamic Packet Loss Case)

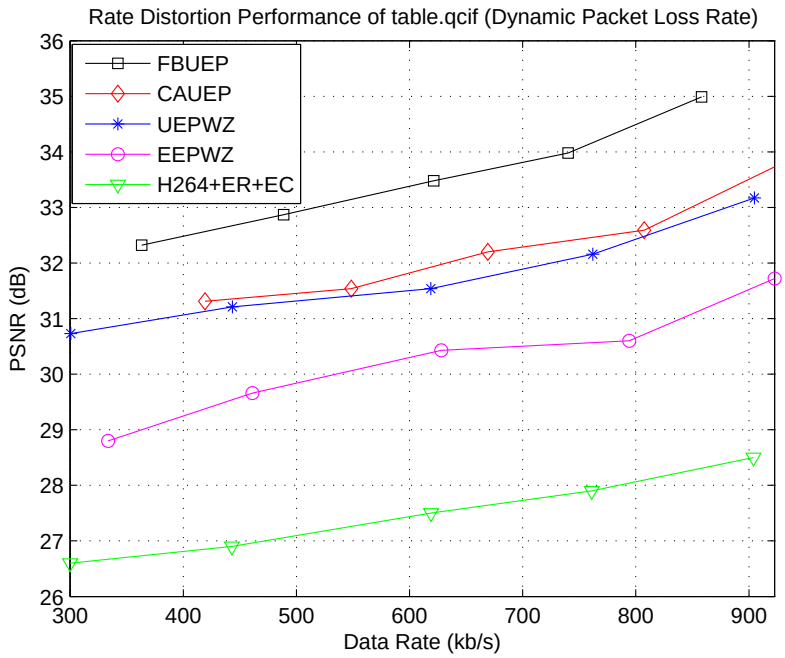

Fig. 4.4. Rate Distortion Performance (table.qcif) (Dynamic Packet Loss Case)

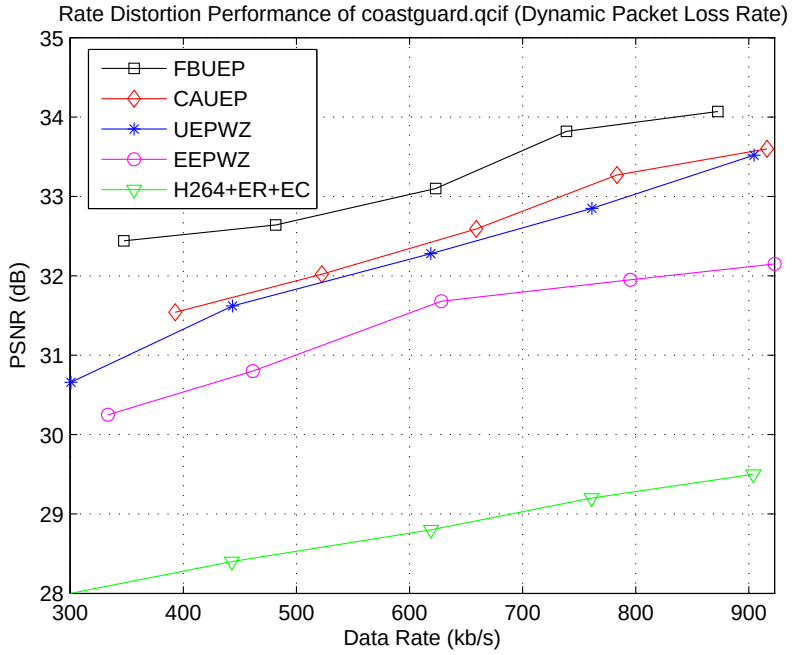

Fig. 4.5. Rate Distortion Performance (coastguard.qcif) (Dynamic Packet Loss Case)

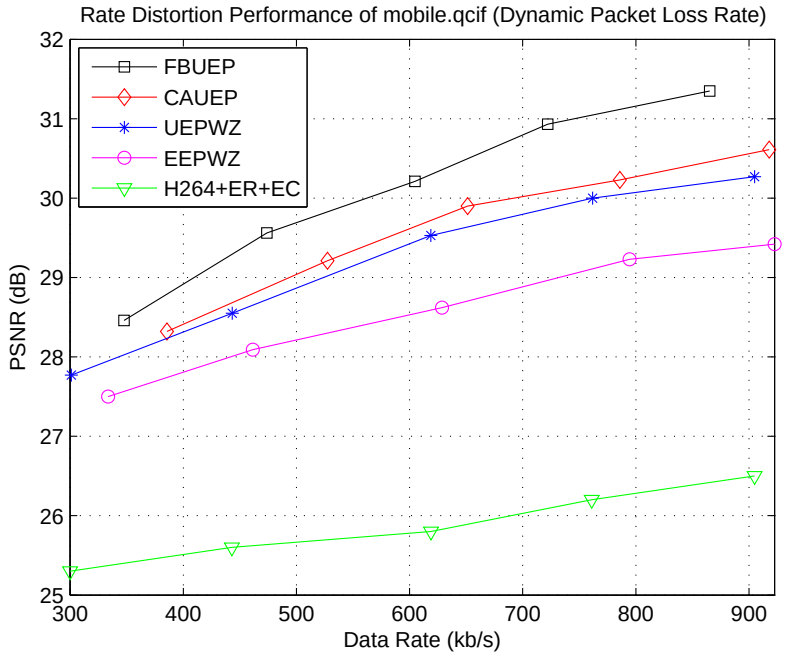

Fig. 4.6. Rate Distortion Performance (mobile.qcif) (Dynamic Packet Loss Case)

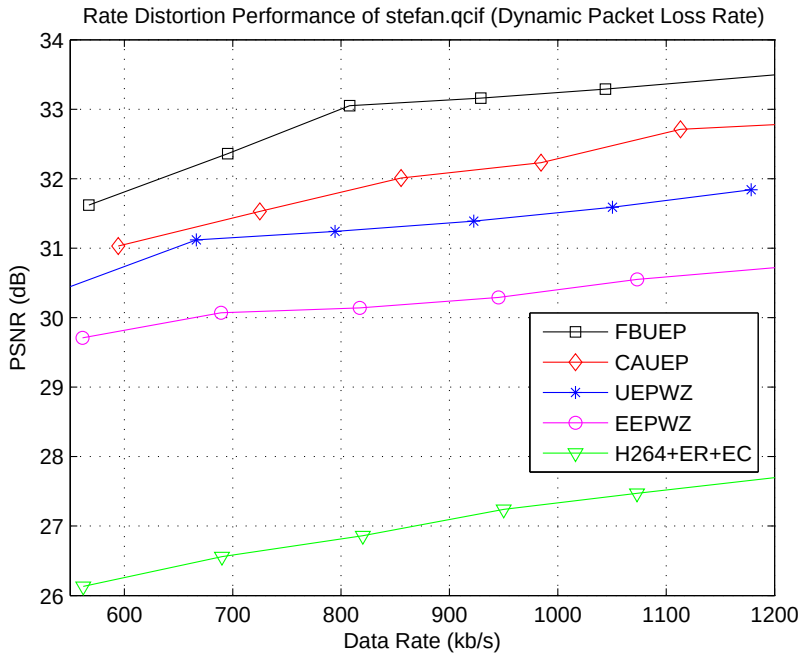

Fig. 4.7. Rate Distortion Performance (stefan-qcif) (Dynamic Packet Loss Case)

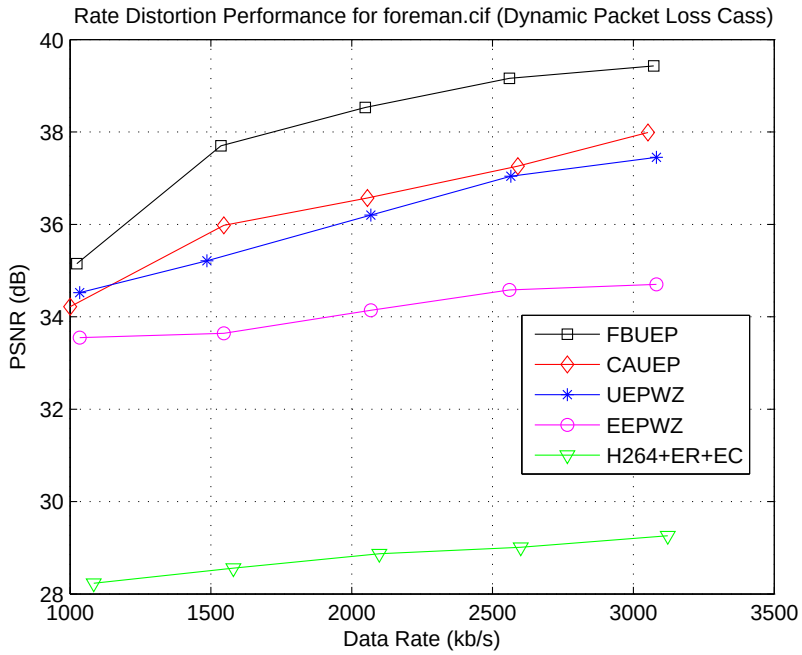

Fig. 4.8. Rate Distortion Performance (foreman.cif) (Dynamic Packet Loss Case)

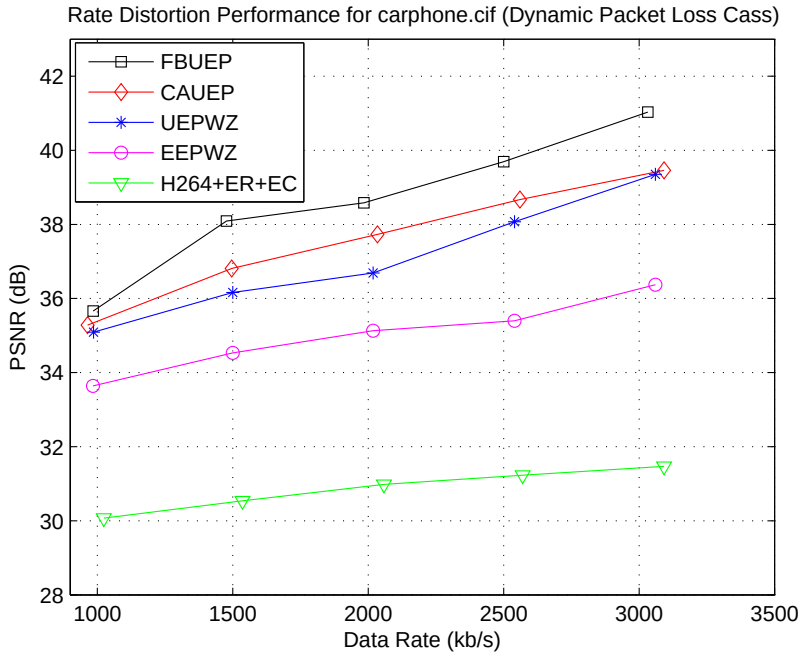

Fig. 4.9. Rate Distortion Performance (carphone.cif) (Dynamic Packet Loss Case)

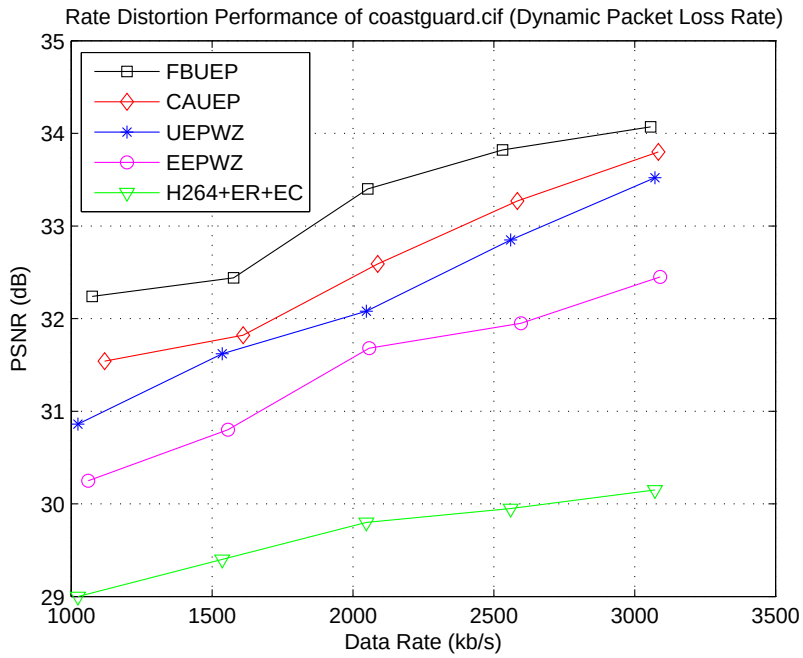

Fig. 4.10. Rate Distortion Performance (coastguard.cif) (Dynamic Packet Loss Case)

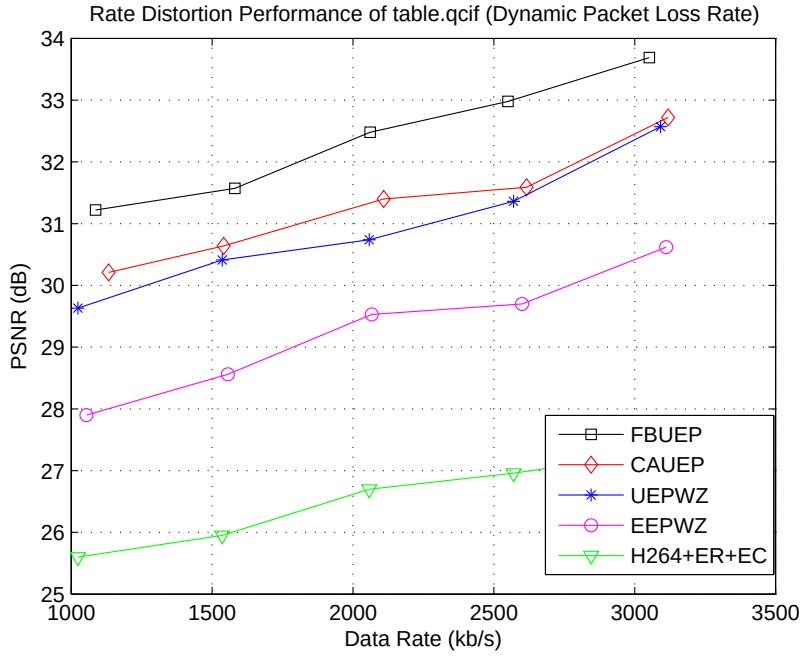

Fig. 4.11. Rate Distortion Performance (table.cif) (Dynamic Packet Loss Case)

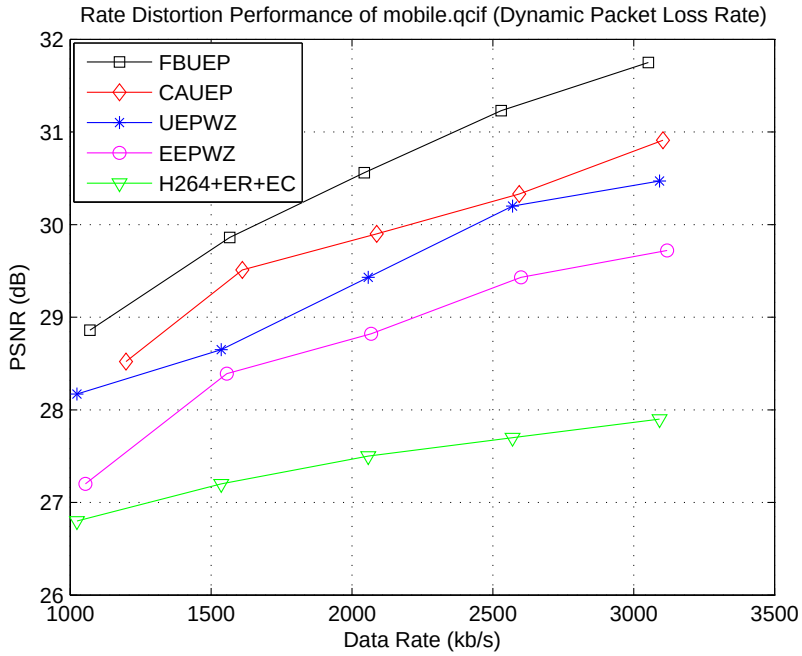

Fig. 4.12. Rate Distortion Performance (mobile.cif) (Dynamic Packet Loss Case)

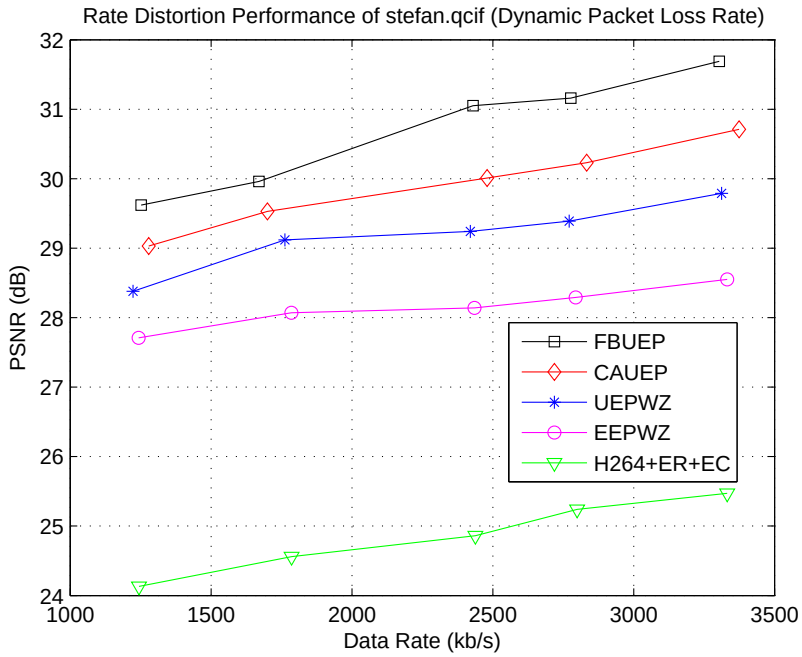

Fig. 4.13. Rate Distortion Performance (stefan.cif) (Dynamic Packet Loss Case)

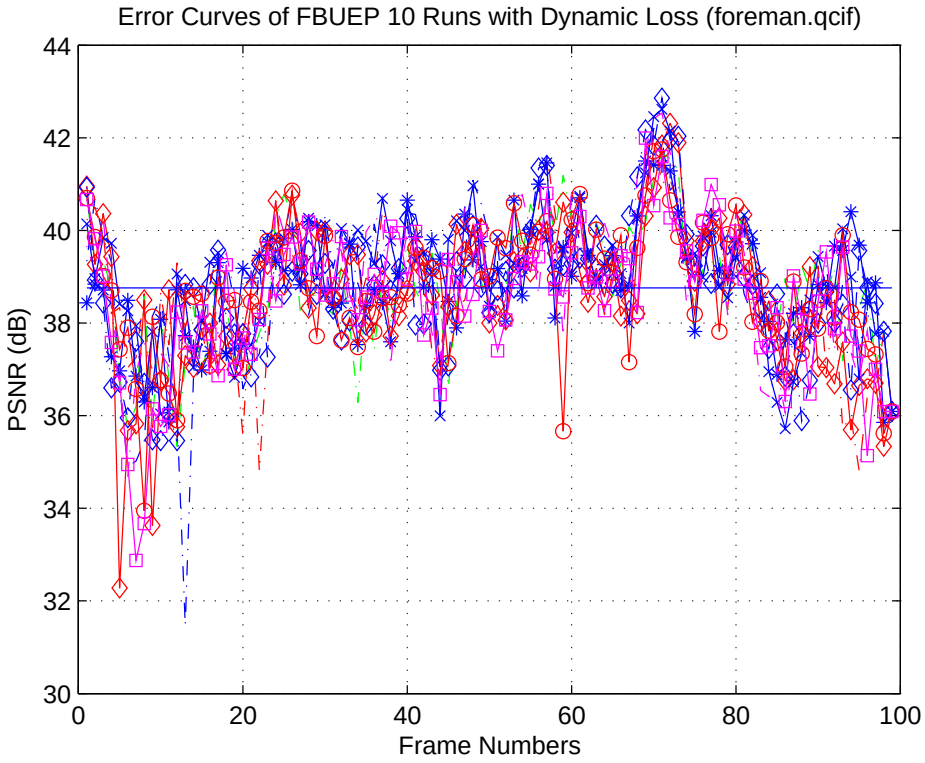

Fig. 4.14. Error Curves of FBUEP 10 Runs with Dynamic Loss (foreman.qcif)

Fig. 4.15. Visual comparison between the original 85th frame (left) and that produced by CAUEP(right: PSNR=38.42dB).

Fig. 4.16. Visual comparison between the 85th frame produced by FBUEP(left: k = 5, PSNR = 39.75dB) and UEPWZ(right: PSNR=37.15dB).

Fig. 4.17. Visual comparison between the 85th frame produced by EEP(left: PSNR=34.64dB) and H264+EC(right: PSNR=29.12dB).

Figure 4.2-4.13 depict the results for the dynamic packet loss case. As can be seen in the dynamic packet loss case, the CAUEP and UEPWZ schemes achieved lower PSNRs at the same data rates compared to PSNRs in the fixed loss cases. One of the reasons is that the CAUEP and the UEPWZ schemes were not able to allocate enough parity bits for protecting the important video information when the channel packet loss rates became higher. Furthermore, distortion is accumulated over a sequence of successive P frames due to motion compensation, until a new I frame is inserted. unlike the other schemes, FBUEP attempts to be aware of the varying packet loss rates and is therefore able to adjust the parity data rates accordingly.

For visual comparison, the 85th frame from *foreman*, which was protected via the various schemes described above, has been decoded and depicted along with the original frame in Figure 4.15, (4.16), and (4.17). The results presented are for dynamic packet losses. It can be seen that both UEPWZ and CAUEP produce block artifacts on the left and right cheeks of the person in the figure, with CAUEP generating less artifacts than UEPWZ. It is also observed that the use of feedback, as in FBUEP, produced the most visually pleasing image. It also has higher PSNR values than the others.

Figure 4.17 compares the results from using EEP and the H.264/AVC with error concealment applied to the decoded frames. Both the visual quality and the PSNRs are much worse than those of UEPWZ, CAUEP and FBUEP.

4.3 Conclusion

In this chapter, we presented a feedback aided unequal error protection technique that is based on our previous unequal error protection method. Both used Wyner–Ziv coding for error resilience. In this technique the parity data rates allocated to motion information and the transform coefficients were adjusted adaptively depending upon feedback received from the decoder regarding packet loss rates. The proposed method resulted in an efficient allocation of the data rate by the Wyner–Ziv codec for the

purpose of error resilience and consequently provided good quality decoded video when data had been corrupted by transmission errors.

5. FEEDBACK AIDED CONTENT ADAPTIVE UNEQUAL ERROR PROTECTION

The system structure of the feedback aided content adaptive unequal error protection (FBCAUEP) is shown in Figure 5.1. FBUEP and CAUEP have been described in the previous chapters. The advantages of using FBUEP are that the encoder is aware of the latest channel packet loss rate which helps the Turbo encoder adjust the error protection strength accordingly. For the CAUEP, the error protection strength is adjusted by an analysis on the content of each frame while no channel packet loss information is known to the encoder. In FBCAUEP, both the content of the video sequence and the latest channel packet loss rate are known by the encoder, therefore, the Turbo encoder can do a more precise parity data rate allocation decision on transform coefficients and the motion information. The data rate ratio between H.264 primary data transmission and Wyner-Ziv parity bits transmission can reach 10:1 to 11:1 due to the efficient parity data rate allocation with assistance from both the video content and the channel packet loss rate.

5.0.1 Pseudo Wyner-Ziv Encoder

The primary codec of the system is an H.264/AVC codec and the Wyner-Ziv codec utilizes coarse quantization and two pairs of Turbo codecs. Instead of protecting everything associated with the coarsely reconstructed frames, we separately protect the motion information and the transform coefficients produced by the primary H.264 encoder. The idea being that since the loss of motion information impacts the quality of decoded video differently from the loss of transform coefficients, both should receive unequal levels of protection that are commensurate with their respective contributions to the quality of the video reconstructed by the decoder [21].

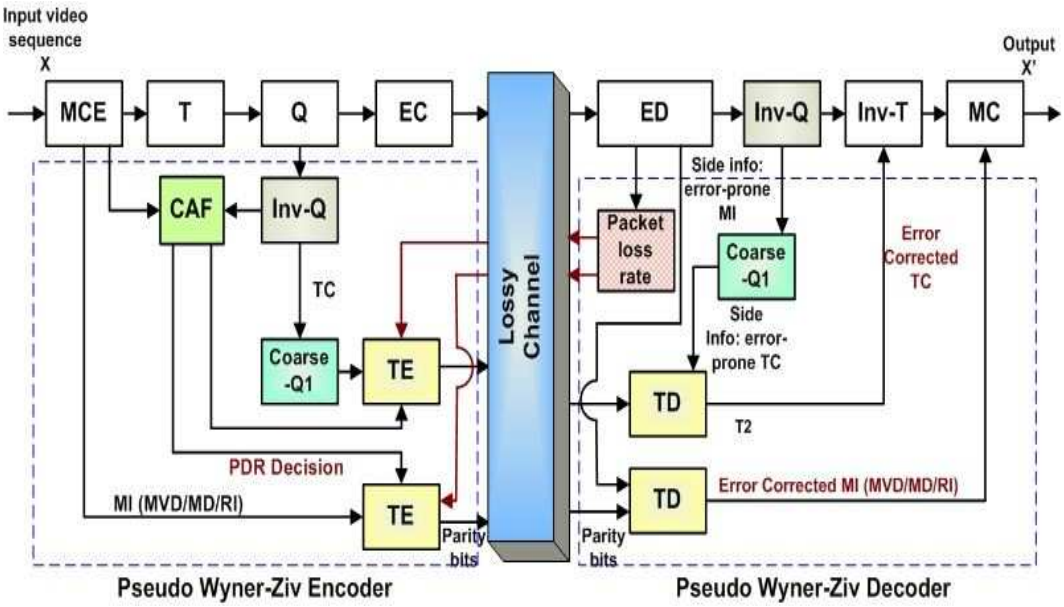

Fig. 5.1. Feedback Aided Content Adaptive Unequal Error Protection Based on Wyner-Ziv Coding

In H.264/AVC, there are 9 modes used for predicting a 4×4 block in an I frame and 4 modes for predicting a 16×16 block from its neighbors [28], [27]. The mode index and the transform coefficients are critical for proper frame reconstruction at the decoder. In the case of P and B frames, the H.264/AVC standard allows the encoder the flexibility to choose among different reference frames and block sizes for motion prediction. In particular, the standard permits block sizes of 4×4, 4×8, 8×4, 8×8, 8×16, 16×8, and 16×16. Since motion vectors belonging to neighboring blocks are highly correlated, motion vector differences (MVD) are encoded and transmitted to the decoder side, together with the reference frame index, mode information and the residual transform coefficients. In the case of I frames, mode information (MI) as well as the transform coefficients are protected whereas motion vector differences, mode information and reference frame index (RI) are protected for P and B frames. These are scanned and used to create long symbol blocks that are sent to the Turbo encoder.

In order to mitigate the mismatch between the transform coefficients input to the Wyner-Ziv encoder and the corresponding side information at the Wyner-Ziv decoder, an inverse quantizer, identical to the one used in the H.264/AVC decoder, is initially used to de-quantize the coefficients. These are then coarsely quantized by a uniform scalar quantizer with 2^N levels ($N \leq 8$), and used to form a block of symbols that is passed onto the Turbo encoder. The quantization step size for processing the transform coefficients is therefore $2^{(8-N)}$.

Turbo Encoding

Due to the importance of maintaining its accuracy the motion information is not quantized. Instead, the Turbo encoder takes in the motion information directly and outputs the selected parity bits. It can be noticed that without using quantization, the processing of Turbo encoding motion information itself is not strictly speaking Wyner-Ziv coding. Therefore, we name the whole secondary encoder as Pseudo Wyner-Ziv encoder instead of Wyner-Ziv encoder. However, the application of Turbo coding in

our schemes is different from straight forward error control coding. In our application, only the parity bits p produced by the Turbo encoder are transmitted to the decoder. The output data stream u from the first branch is not transmitted to the decoder side. This is illustrated in Figure 2.2. The corresponding decoded error prone primary video data from the H.264 decoder will be used as it to co-decode the parity bits received by the Turbo decoders. Because of the independent processing of the motion data and the transform coefficients in the Pseudo Wyner–Ziv encoder, the parity data rates in the corresponding Turbo encoder can be assigned separately.

The Turbo encoder we used consists of two identical recursive systematic encoders (see Figure 2.2) [38], each having the generator function: $H(D) = \frac{1+D^2+D^3+D^4}{1+D+D^4}$. The input symbol blocks sent to the second recursive encoder are interleaved first in a permuter before being passed to it. The puncture mechanism is used to delete some of the parity bits output from the two recursive encoders, in order to meet a target parity data rate. Only parity bits are transmitted to the decoder side. The first branch of data, symboled by the dashed line in Figure 2.2, is not transmitted. The error correction capability of the Turbo coder also depends on the length of the symbol blocks. In our scheme, the symbol block length is in the unit of a frame instead of a slice. Each packet contains the video data corresponding to one slice. Therefore, if a packet got lost, the video information for this slice is dropped. For the transform coefficients, the symbol block length is 25344 for a QCIF sequence. In the proposed scheme the motion vectors are obtained for each 4x4 blocks, which makes the symbol block length of 3168. The experiment results also show that the Turbo encoder still maintains strong error correction ability for such a symbol block length.

Content Adaptive Function

At the encoder side, a content adaptive function (CAF) is implemented in order to help making decisions on the parity data rate allocation for protecting different video data elements. As shown in the system structure diagram (Figure 5.1), the motion information and the residual frame information are passed to the CAF block.

An estimation on the average SAD of the residuals in each frame and the average SAD of the motion vector lengths in each P(B) frame is conducted. The equation description of the two estimation is given as follows:

$$\overline{SAD_n^{Res}} = \frac{1}{N \times M} \sum_{i=0,j=0}^{i=N,j=M} \left| X_{i,j} - X_{p(i,j)} \right| \tag{5.1}$$

where $X_{i,j}$ denotes the reconstructed pixel value at position (i, j), $X_{p(i,j)}$ is the value of the predicted pixel at position (i, j), and $\overline{SAD_n^{Res}}$ represents the average value of SAD of the n-th frame in the sequence.

$$\overline{SAD_n^{MV}} = \frac{1}{Q \times P} \sum_{i=0,j=0}^{i=Q,j=P} \sqrt{(MV_{x(i,j)})^2 + (MV_{y(i,j)})^2} \tag{5.2}$$

where $MV_{x(i,j)}$ and $MV_{y(i,j)}$ denote the motion vector's x-axis and y-axis values at block position (i, j).

After this estimation, the suggested parity data rates assignment are sent to the two Turbo encoders. Together with the received feedback message of the current channel packet loss, the final decision on the parity data rates allocation can be made to Turbo encode the protected motion information and the coarse version of the transform coefficients in each frame.

5.0.2 Pseudo Wyner-Ziv Decoder

The Turbo decoder utilizes the received parity bits and the side information from the H.264/AVC decoder, to perform the iterative decoding using two BCJR-MAP decoders [38]. The error corrected information is then sent back to the H.264/AVC decoder to replace the error corrupted data. In this process, the decoded error-prone transform coefficients are first sent to a coarse quantizer, which is the same as the one used at the Pseudo Wyner–Ziv encoder side. The reason is that at the encoder side, in order to save data rate usage of the Wyner-Ziv coding, a coarse version of the transform coefficients is Turbo encoded. The coarsely quantized transform coefficients then served as the side information to the Turbo decoder, together with the received

parity bits of the Turbo encoded coarse-version transform coefficients, to decode the error corrected coarse version of the transform coefficients.

When using the real-time transport protocol (RTP), packet loss can be inferred at the decoder easily by checking the sequence number field in the RTP headers. Wyner-Ziv decoding only performs when the decoder detects packet losses. When no packet loss happens, the H.264 decoded transform coefficients are used for decoding the residual frames. However, when packet loss happens, the coarser version of the transform coefficients decoded by the Turbo decoder is used to limit the maximum degradation that can occur.

In the parallel process, the error corrupted motion information received by the H.264/AVC decoder was sent directly to the corresponding Turbo decoder, together with the received corresponding parity bits, to decode the error corrected motion information. It is then sent back to the H.264/AVC decoder to replace the error-corrupted motion information. The reconstructed frames can be further used as the reference frames in the following decoding process. Therefore, the final decoded video sequence are obtained based on the error corrected motion information and the transform coefficients, which resulted in good quality decoded frames as shown in Section (5.2). However, in the case of serious channel loss and/or limited available data rate for error protection, the Pseudo Wyner–Ziv coder might not have enough strength to recover all the lost video information. On this point, the unequal error protection takes the advantage of allocating different protection level on different protected video data elements depending on their overall impact on the decoded video sequence. The experiments showed that by assigning unequal data rate for protecting motion information and the transform coefficients, the rate distortion performance can be improved compared to the equal parity data rate allocation case.

Feedback Message Modes and Transmission

It is to be noted that data networks suffer from two types of transmission errors, namely random bit errors due to noise in the channels and packet losses due to

network congestion. When transmitting a data packet, a single uncorrected bit error in the packet header or body may result in the whole packet being discarded [31]-[42]. In the current work, we only consider packet losses, whether due to network congestion or uncorrected bit errors. When using the real-time transport protocol (RTP), determining which packets have been lost can be easily achieved by monitoring the sequence number field in the RTP headers [24], [32]. Therefore, the packet loss rate of each frame can be easily obtained at the decoder.

At the H.264/AVC encoder, each frame is divided into several slices. Both the motion information and the transform coefficients of each slice are sent to the Pseudo Wyner–Ziv encoder to be encoded independently by the two Turbo encoders.

At the decoder, the packet loss rate of each frame is evaluated based on the received video information. It is then sent back to the two Turbo encoders via the RTCP feedback packets. If the available bandwidth for transmitting the feedback packets is above a certain threshold then an immediate mode RTCP feedback message is sent, otherwise the early feedback RTCP mode is used [43]. The two Turbo encoders update the parity data rates for encoding the motion information and the transform coefficients based upon the received RTCP feedback conveying the packet loss rates. This way the Pseudo Wyner–Ziv encoder attempts to adapt to the decoder's needs, while avoiding blindly sending a large number of parity bits that may not be needed when the packet loss is low or zero. In the case of high channel packet loss rate, the Pseudo Wyner–Ziv encoder enhances the protection by allocating more data rates to the Turbo encoded data, especially the motion information, while decreasing relatively the data rate used for encoding the main data stream by the H.264/AVC encoder. In this way, the total data rate is kept as a constant so that it won't exacerbate the possible congestion over the network transmission.

According to the RTCP feedback profile that is detailed in [43], when there is sufficient bandwidth, each loss event can be reported by means of a virtually immediate RTCP feedback packet. In the RTCP immediate mode, feedback message can be sent for each frame to the encoder. In our scheme an initial parity data rates is set at

the beginning of transmitting a video. When the channel loss condition changes, the immediate mode RTCP feedback packet sends the latest channel packet loss rate to the Turbo encoders to adjust the parity data rate assignment. If we let N_L denote the average number of loss events to be reported every interval T by a decoder, B the RTCP bandwidth fraction for our decoder, and R the average RTCP packet size, then feedback can be sent via the immediate feedback mode when $N_L \leq \frac{B*T}{R}$.

In the RTCP protocol profile [43], it was assumed that 2.5 percent of the the RTP session bandwidth is available for RTCP feedback from the decoder to the encoder. For example, for a 512 kbits/s stream, 12.8 kbits are available for transmitting the RTCP feedback. If we assume an average of 96 bytes (768 bits) per RTCP packet and a frame rate of 15 frames/second, we can conclude that $N_L \leq \frac{12800*\frac{1}{15}}{768} = 1.11$. In this case, the RTCP immediate mode can be used to send one feedback message per frame to the encoder.

When $N_L > \frac{B*T}{R}$, the available bandwidth is not sufficient for transmitting a feedback message via the immediate mode. In this case, the early RTCP mode is turned on. In this mode, the feedback message is scheduled for transmission to the encoder at the earliest possible time, although it can not necessarily react to each packet loss event.

When the Pseudo Wyner–Ziv encoder does not receive feedback regarding the current packet loss rates (the feedback packet got lost during transmitting back to the Turbo encoders or the available bandwidth is not sufficient for immediate mode feedback), the Turbo encoders keep using the last received channel packet loss rate to decide the parity data rates for encoding the motion information and the transform coefficients of the current encoded frame.

5.0.3 Parity Data Rate Allocation for FBCAUEP

In the FBCAUEP scheme, the parity data rate allocation is decided by both the current channel packet loss situation and the content of each frame. A systematic experiments have been conducted for four types of the video sequences: slow motion,

mid-slow motion, mid-motion, fast motion, under different amount of packet loss cases. From the statistical results of these experiments, the parity data rate allocation is assigned as in Table(5.1).

In the FBCAUEP, the total data rate allocated for transmitting the parity data is calculated as in Equation5.3

$$WZ - DR_{fbcauep} = \sum_{i=1}^{N} TC_i + \sum_{j=1}^{N} MVD_j \tag{5.3}$$

$$= (\frac{1}{N}\sum_{i=1}^{N} PDR_{TC_i}) \times H_{TC} \times W_{TC} \times PN_{symbol} \times FR \tag{5.4}$$

$$+ (\frac{1}{N}\sum_{j=1}^{N} PDR_{MVD_j}) \times H_{MV} \times W_{MV} \times PN_{symbol} \times FR \tag{5.5}$$

Where H_{TC}, H_{MVD} is the height of the TC and MVD data packet. The W_{TC}, W_{MVD} is the width of the TC and MVD data packet. PN_{symbol} is the number of the symbol plane and the FR is the current frame rate for transmitting the video packets. N is the total number of the frames in the video sequence. In FBUEP technique, parity data rate assigned for protecting TC and the MVD are different for each frame. Therefore the parity data rate assigned for TC and MVD are summed up for all frames.

5.1 Analysis and Modeling of the Unequal Error Protection

In this section, the proposed unequal error protection technique based on Wyner-Ziv coding is analyzed theoretically. An end to end video quality distortion model is derived for the system. An optimization method of the data rate allocation between the primary H.264 encoding and the Wyner-Ziv encoding is proposed.

There are several factors result in the decoded video quality distortion: the primary layer video reconstruction distortion; the quantization mismatch resulted from the coarse quantizer used in the Wyner-Zic encoder; the channel packet losses; the error propagation in the frames that are predicted from the previous decoded frames.

The distortion analysis for the P frames is described as follows and the derivation for I and B frames can be obtained similarly. In a P frame, if we describe the original pixel value in the k-th frame as $Y^k_{org(i,j)}$ and the corresponding reconstructed pixel value in the H.264 encoder as $Y^k_{rec(i,j)}$, the encoder's reconstruction of the original pixel can be given by [7]:

$$Y^k_{rec(i,j)} = Y^{k-z}_{rec(f,g)} + Res^k_{(i,j)} \tag{5.6}$$

Where $Res^k_{(i,j)}$ represents the residual component, and $Y^{(k-z)}_{rec(f,g)}$ is the prediction of the current pixel value from the previously reconstructed $(k-z)$-th frame at position (f,g). Since H.264 allows the reference frame range from the previous 16 reconstructed frames for P frame prediction, we have $z <= 16$.

Similarly we model the Wyner-Zic encoder's reconstruction of $Y^k_{org(i,j)}$ as:

$$Y^{wz(k)}_{rec(i,j)} = Y^{(k-z)}_{rec(f,g)} + Res^{wz(k)}_{(i,j)} \tag{5.7}$$

Where $Res^{wz(k)}_{(i,j)}$ is a coarser version residual value as a coarse quantizer is implemented in the Wyner-Ziv encoder.

To model the H264 decoder's reconstruction, we have:

$$\widehat{Y^k_{rec(i,j)}} = \widehat{Y^{k-z}_{rec(f,g)}} + Res^k_{(i,j)} \tag{5.8}$$

Similarly, the model for the Wyner-Ziv decoder's reconstruction is:

$$\widehat{Y^{wz(k)}_{rec(i,j)}} = \widehat{Y^{(k-z)}_{rec(f,g)}} + Res^{wz(k)}_{(i,j)} \tag{5.9}$$

If we assume the channel packet loss probability is $P^k(l)$ for transmitting the k the frame, $(1-P^k(l))$ represent the loss free case. In our study, we focus on the video streaming over wireless network in which case, $P^k(l)$ mainly depends on the packet size and the channel loss characteristics. We represent the percentage video data that the Wyner-Ziv coding can successfully recovered as $P^k_w z(s)$, which depends on the amount of channel packet losses and the allocated data rate R_{wz} for transmitting the parity bits.

In our current system, no error concealment is applied on the decoded video data. Therefore, there are two end to end video distortion cases.

When no packet loss happens ($(1-P^k(l))$), Wyner-Ziv coding is not necessary, the overall distortion is contributed only from the primary H.264 encoding and decoding. In the case of the inter frame prediction, the error propagation is the main source of the distortion. If we represents the distortion in this case for the k the frame as D^k_{EP}) :

$$D^k_{EP} = E[(Y^k_{org(i,j)} - \widehat{Y^k_{rec(i,j)}})^2] \tag{5.10}$$
$$= E[(Y^k_{org(i,j)} - Y^k_{rec(i,j)} + Y^k_{rec(i,j)} - \widehat{Y^k_{rec(i,j)}})^2] \tag{5.11}$$
$$= E[(Y^k_{org(i,j)} - Y^k_{rec(i,j)})^2] + E[(Y^k_{rec(i,j)} - \widehat{Y^k_{rec(i,j)}})^2)] \tag{5.12}$$
$$+ 2E[(Y^k_{org(i,j)} - Y^k_{rec(i,j)})(Y^k_{rec(i,j)} - \widehat{Y^k_{rec(i,j)}})] \tag{5.13}$$
$$= D^k_p + E[(Y^k_{rec(i,j)} - \widehat{Y^k_{rec(i,j)}})^2)] \tag{5.14}$$
$$= D^k_p + E[(Y^{(k-z)}_{rec(f,g)} - \widehat{Y^{(k-z)}_{rec(f,g)}})^2)] \tag{5.15}$$
$$= D^k_p + E[(Y^{(k-z)}_{org(f,g)} - \widehat{Y^{(k-z)}_{rec(f,g)}} + Y^{(k-z)}_{rec(f,g)} - Y^{(k-z)}_{org(f,g)})^2] \tag{5.16}$$
$$= D^k(H264) + E[(Y^{(k-z)}_{org(f,g)} - \widehat{Y^{(k-z)}_{rec(f,g)}})^2] + E[(Y^{(k-z)}_{rec(f,g)} - Y^{(k-z)}_{org(f,g)})^2] \tag{5.17}$$
$$- 2E[(Y^{(k-z)}_{org(f,g)} - \widehat{Y^{(k-z)}_{rec(f,g)}})(Y^{(k-z)}_{rec(f,g)} - Y^{(k-z)}_{org(f,g)})] \tag{5.18}$$
$$= D^k_p + D^{(k-z)}_{EE} + D^{(k-z)}_p - 2E[(Y^{(k-z)}_{org(f,g)} - Y^{(k-z)}_{rec(f,g)})^2] \tag{5.19}$$
$$- 2E[(Y^{(k-z)}_{rec(f,g)} - \widehat{Y^{(k-z)}_{rec(f,g)}})(Y^{(k-z)}_{org(f,g)} - Y^{(k-z)}_{rec(f,g)})] \tag{5.20}$$
$$= D^k_p + D^{(k-z)}_{EE} + D^{(k-z)}_p - 2D^{(k-z)}_p \tag{5.21}$$
$$= D^k_p + D^{(k-z)}_{EE} - D^{(k-z)}_p \tag{5.22}$$

Where D^k_p represents the distortion in the k-th frame from the primary H.264 encoding process and $D^{(k-z)}_{EE}$ denotes the end to end distortion in the $(k-z)$-th frame. It is assumed that the quantization mismatch between the original pixels and the reconstructed pixels $(Y^{(k)}_{org(f,g)} - Y^{(k)}_{rec(f,g)})$ has zero mean and is independent of the distortion introduced by the channel $(Y^{(k-z)}_{rec(f,g)} - \widehat{Y^{(k-z)}})$. With these assumptions,

the step (10) and (17) in the above derivation become zero. If we assume that the quantization parameters are fixed in the primary layer H.264 encoder and the resulted quantization mismatch distortion in each reconstructed frames is almost the same ($D_p^k = D_p^{(k-z)}$). The overall distortion resulted from the error propagation can be simplified as:

$$D_{EP}^k = D_{EE}^{(k-z)} \tag{5.23}$$

When channel packet loss happens ($P^k(l)$), the Wyner-Ziv coder protect the video information by using the proposed algorithm. In this case, the distortion is contributed from the coarse quantization mismatch, the percentage lost information that can be recovered by the Wyner-Ziv coding and the error propagation from the inter frame prediction.

$$\begin{aligned}
D_{WZ}^k &= E[(Y_{org(i,j)}^k - \widehat{Y_{rec(i,j)}^{wz(k)}})^2] && (5.24)\\
&= E[(Y_{org(i,j)}^k - Y_{rec(i,j)}^{wz(k)} + Y_{rec(i,j)}^{wz(k)} - \widehat{Y_{rec(i,j)}^{wz(k)}})^2] && (5.25)\\
&= E[(Y_{org(i,j)}^k - Y_{rec(i,j)}^{wz(k)})^2] + E[(Y_{rec(i,j)}^{wz(k)} - \widehat{Y_{rec(i,j)}^{wz(k)}})^2] && (5.26)\\
&+ 2E[(Y_{org(i,j)}^k - Y_{rec(i,j)}^{wz(k)})(Y_{rec(i,j)}^{wz(k)} - \widehat{Y_{rec(i,j)}^{wz(k)}})] && (5.27)\\
&= D_{wz(e)}^k + E[(Y_{rec(i,j)}^{wz(k)} - \widehat{Y_{rec(i,j)}^{wz(k)}})^2] && (5.28)\\
&= D_{wz(e)}^k + E[(Y_{rec(f,g)}^{(k-z)} - \widehat{Y_{rec(f,g)}^{(k-z)}})^2] && (5.29)\\
&= D_{wz(e)}^k + D_{EE}^{(k-z)} + D_p^{(k-z)} - 2D_p^{(k-z)} && (5.30)\\
&= D_{wz(e)}^k + D_{EE}^{(k-z)} - D_p^{(k-z)} && (5.31)
\end{aligned}$$

Where $D_{wz(e)}^k$ represents the distortion resulted from the Wyner-Zic encoding process and the same assumptions while deriving D_{EP}^k hold.

To summarize, the overall distortion can be modeled by the following function:

$$\begin{aligned}
D_{EE}^k &= (1 - P^k(l))D_{EP}^k + P^k(l)D_{WZ}^k && (5.32)\\
&= (1 - P^k(l))D_p^k + D_{EE}^{(k-z)} - D_p^{(k-z)} + P^k(l)(D_{wz(e)}^k + D_{EE}^{(k-z)} - D_p^{(k-z)}) && (5.33)
\end{aligned}$$

Where, we assume that Wyner-Ziv coding can always successfully recover the packet losses.

From the analysis in [40], the rate distortion performance of the primary H.264 encoder and the Wyner-Ziv encoder can be modeled by the following parameter equations:

$$D_p^k = D_m + \frac{\theta}{R_p - R_m} \tag{5.34}$$

$$D_{wz(e)}^k = D_m + \frac{\theta}{R_{wz(e)} - R_m} \tag{5.35}$$

It is noted that the transmission data rate for the Wyner-Ziv encoded parity bits is:

$$R_{wz} = PDR \times R_{wz(e)} \tag{5.36}$$

where $PDR < 1$.

The overall transmission data rate is therefore:

$$R = R_p + R_{wz} \tag{5.37}$$

We represent the average end to end distortion for a GOP of frames as:

$$D = \frac{1}{N} \sum_{k=1}^{N} D_{EE}^k \tag{5.38}$$

where N is the number of frames in a GOP

Based on the above equations, an optimization on the data rate allocation between R_p and R_{wz} can be derived which results in the minimum end to end distortion (D) at a given channel packet loss rate.

5.2 Experiment Results

To evaluate the proposed techniques, experiments were carried out using the $JM10.2$ H.264/AVC reference software. The frame rate for each sequence was set at 15 frames per second with an *I-P-P-P...* GOP structure. In our experiment, each QCIF frame is divided into 9 slices. The primary encoded video data output from the

H.264 encoder are packetized into 9 packets per frame, each containing the video information of one slice. The Turbo encoded parity bits of the motion information and the transform coefficients corresponding to each slice are also sent in separate packets. All three types of the packets are subjected to random losses over the transmission channel. Since the packets containing the parity bits of the motion information or the transform coefficients are much smaller in size comparing to the H.264 packets, the possibility of getting lost over a wireless network transmission is therefore much smaller. All the experiments results were averaged over 30 lossy channel transmission realizations.

Data networks suffer from two types of transmission errors: random bit error and packet drop. In our experiments, we only consider the case of packet erasures, whether due to network congestion or uncorrected bit errors. Lower the total data rate to reduce the network congestion is a realistic solution when packet loss is very high. However, since our main application is for video streaming over wireless networks in which case the packet loss situation is more complicated, we didn't consider it in our current experiments. It is to be noted that simply increasing the parity bits when the packet loss rate increases is not applicable, since it will exacerbate network congestion. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases, the primary data transmission rate should be lowered in order to spare more bits for parity bits transmission.

In our experiments, channel packet loss is simulated by using uniform random number generators. Our algorithm focuses on wireless network application in which case severe packet loss could happen. In the case of wireless network transmission, the probability that the packet arrives in error is approximately proportional to its length [44].

The results of the FBCAUEP is compared to those from the unequal error protection (UEPWZ), the content adaptive unequal error protection (CAUEP), the feedback aided unequal error protection (FBUEP), the equal error protection (EEP) and the H.264 with error concealment for the decoded frames.

For the UEPWZ technique, the protection level provided for the motion information and the transform coefficients was fixed for the entire video sequence, and was not changed to cater for different video content or varied channel packet losses. For CAUEP, the protection level of the motion information and the transform coefficients is always adapted to the content of each frame, that is the parity data rates allocated to the protected video information in the Pseudo Wyner–Ziv coder were adjusted according to the content of each frame. The rate, however, was not adjusted to deal with variable channel losses. In the case of FBUEP, the packet loss rates were evaluated by the H.264/AVC decoder and sent back to the Pseudo Wyner–Ziv encoder via RTCP feedback packets.In the event that the RTCP feedback packets got lost during transmission, the Turbo encoders will keep using the last received channel loss rate to decide the parity data rate allocation for the next frame. It is noted that the protection level in the Pseudo Wyner–Ziv encoder is not adjusted according to the frame content in the case of FBUEP. In all three proposed schemes, the error concealment is not applied on the H.264 decoded data before being passed to the Wyner-Ziv decoder.

The results of the dynamic packet losses for transmitting video sequences *foreman*, *carphone*, *mobile*, *coastguard*, *table*, *stefan* are shown from Figure 5.2 to Figure 5.7. It can be seen that for *foreman*, FBCAUEP outperforms FBUEP by around 0.2-0.6 dB and it outperforms the CAUEP by around 1.2-2.1 dB. For *carphone*, FBCAUEP outperforms FBUEP by around 0.1-0.4 dB and it outperforms the CAUEP by around 0.4-2 dB. For *mobile*, FBCAUEP outperforms FBUEP by around 0.1-0.3 dB and it outperforms the CAUEP by around 0.5-1 dB. For *coastguard*, FBCAUEP outperforms FBUEP by around 0.2 dB and it outperforms the CAUEP by around 1 dB. For *table*, FBCAUEP outperforms FBUEP by around 0.1-0.2 dB and it outperforms the CAUEP by around 1-1.3 dB. *stefan*, FBCAUEP outperforms FBUEP by around 0.1-0.3 dB and it outperforms the CAUEP by around 0.8-1.2 dB.

The visual quality of the decoded frames are shown from Figure 5.14 to Figure 5.19 for video sequence *table.qcif*, *stefan.qcif* and *foreman.qcif* at a total data rate of

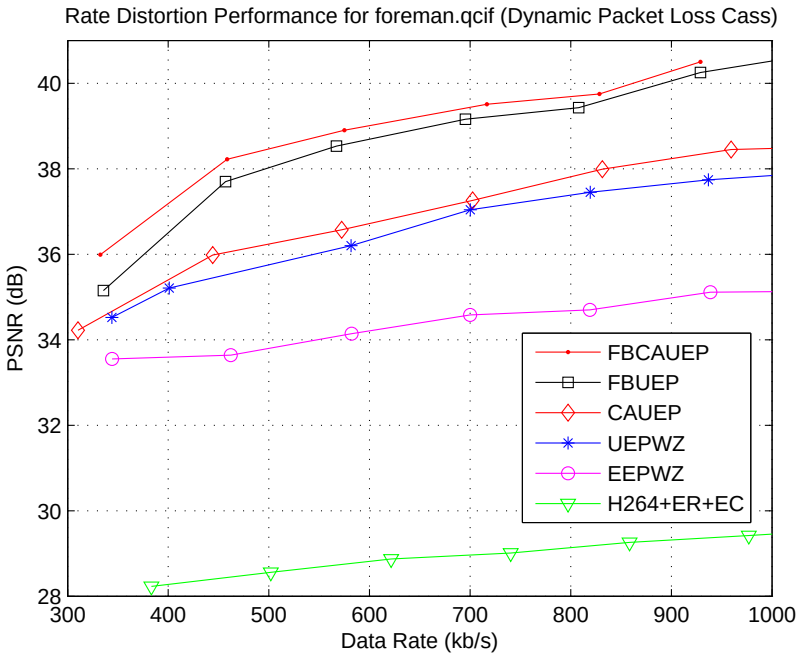

Fig. 5.2. Rate Distortion Performance of foreman.qcif

512 kbps with dynamic packet losses. It can be seen that FBCAUEP can reconstruct a better visual quality compared to that of FBUEP.

Figure 5.13 shows the ten runs of the PSNR curves of the *foreman.qcif* sequence for the first 100 frames. The channel packet loss is at a dynamic packet loss pattern in which case, the packet loss is randomly varied from 0 to 5 packets per frame.

5.3 Conclusion

In this chapter we presented an error resilient scheme that combines the content adaptive unequal error protection and the feedback aided unequal error protection. By adapting the parity data rates to both the content of each frame and the current channel packet losses, it improves the rate distortion performance as well as the visual quality of the decoded frames.

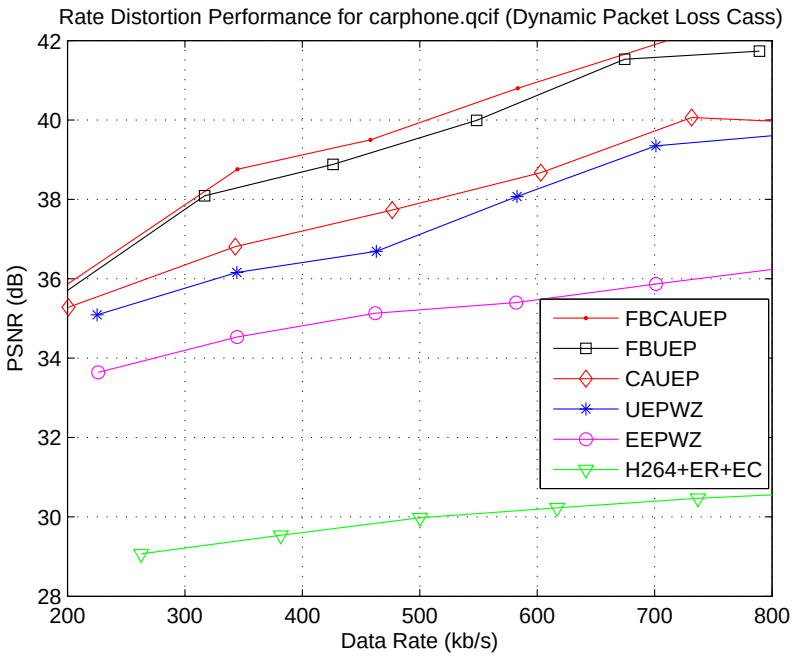

Fig. 5.3. Rate Distortion Performance of carphone.qcif

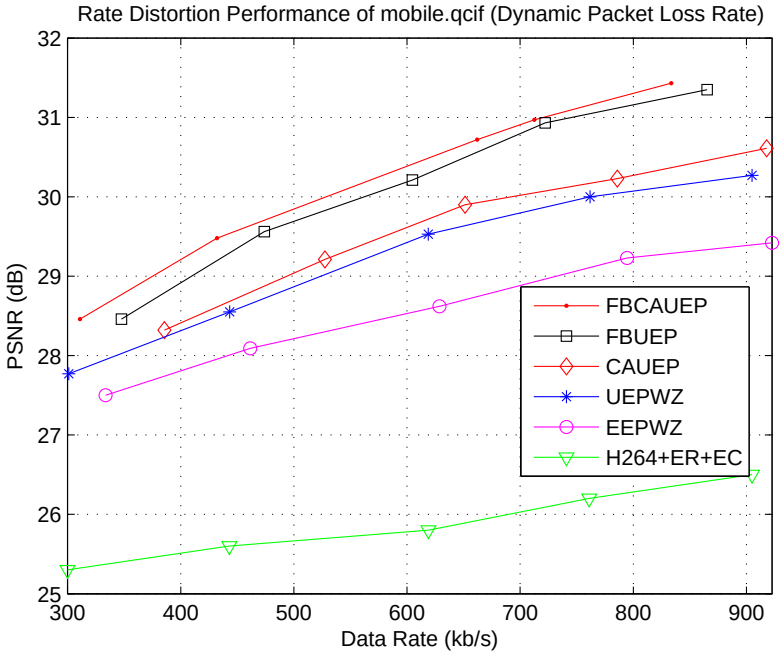

Fig. 5.4. Rate Distortion Performance of mobile.qcif

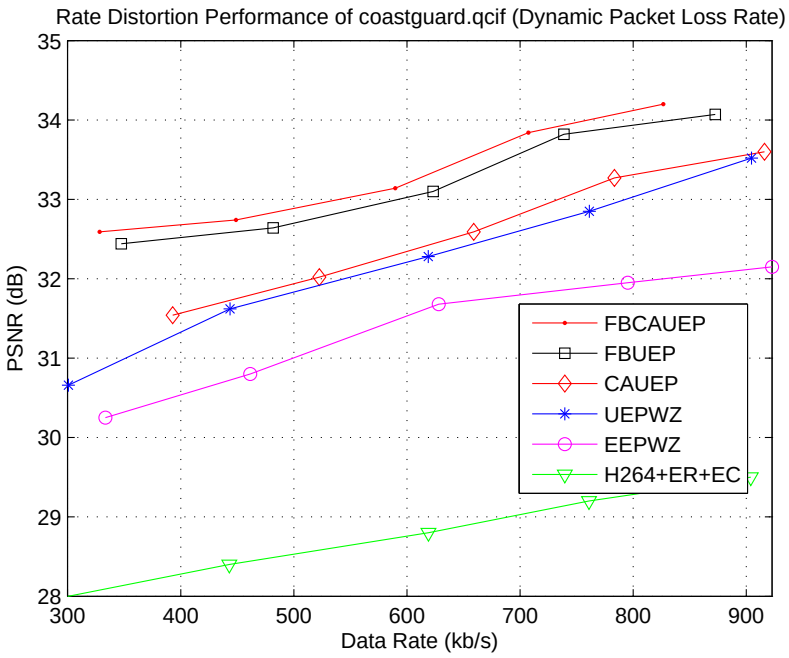

Fig. 5.5. Rate Distortion Performance of coastguard.qcif

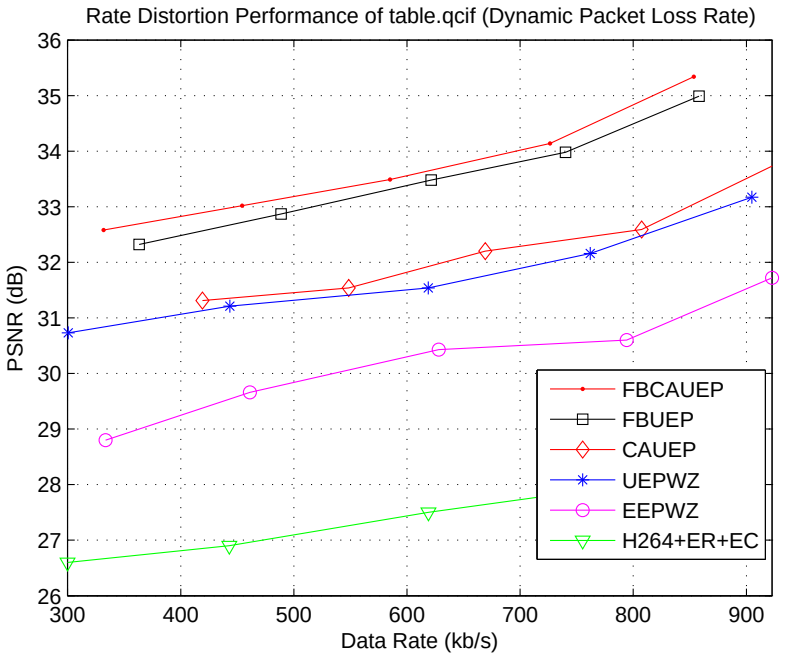

Fig. 5.6. Rate Distortion Performance of table.qcif

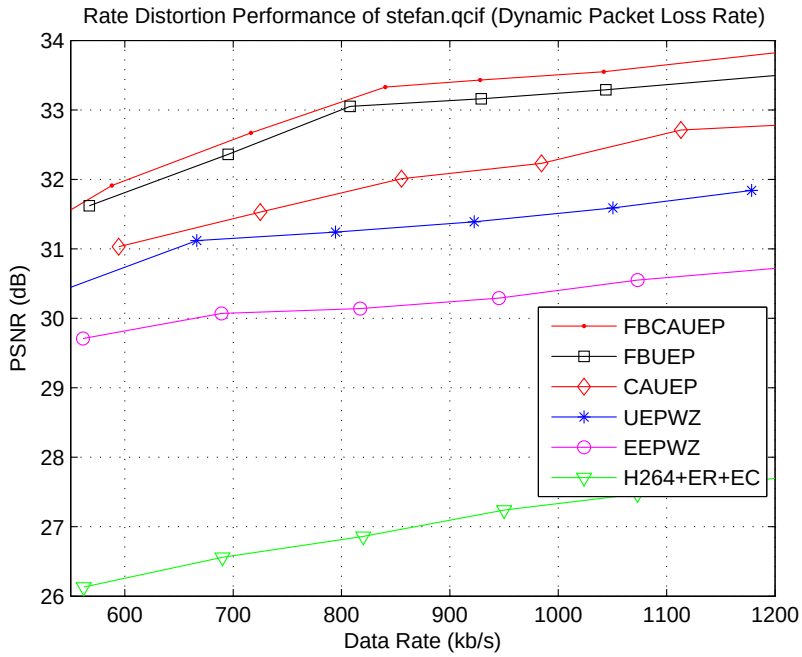

Fig. 5.7. Rate Distortion Performance of stefan.qcif

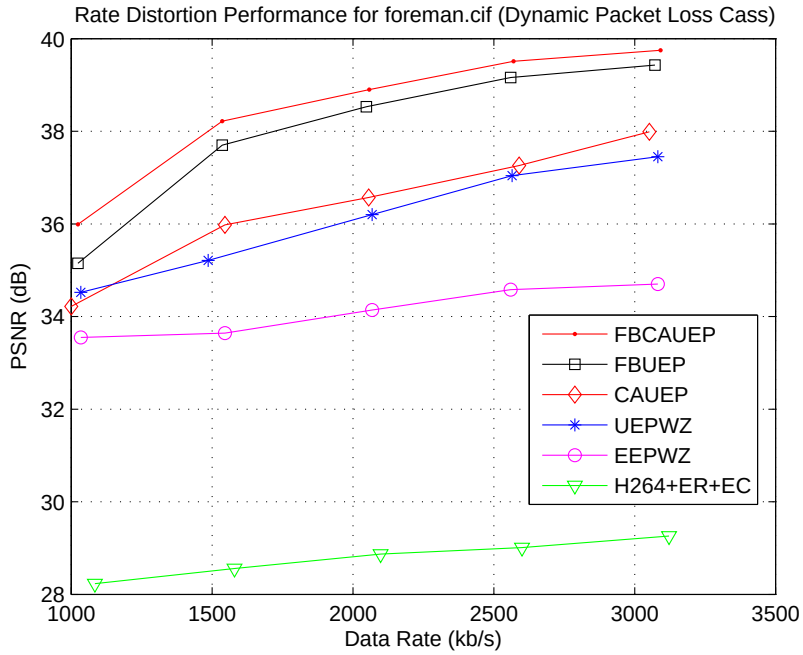

Fig. 5.8. Rate Distortion Performance (foreman.cif) (Dynamic Packet Loss Case)

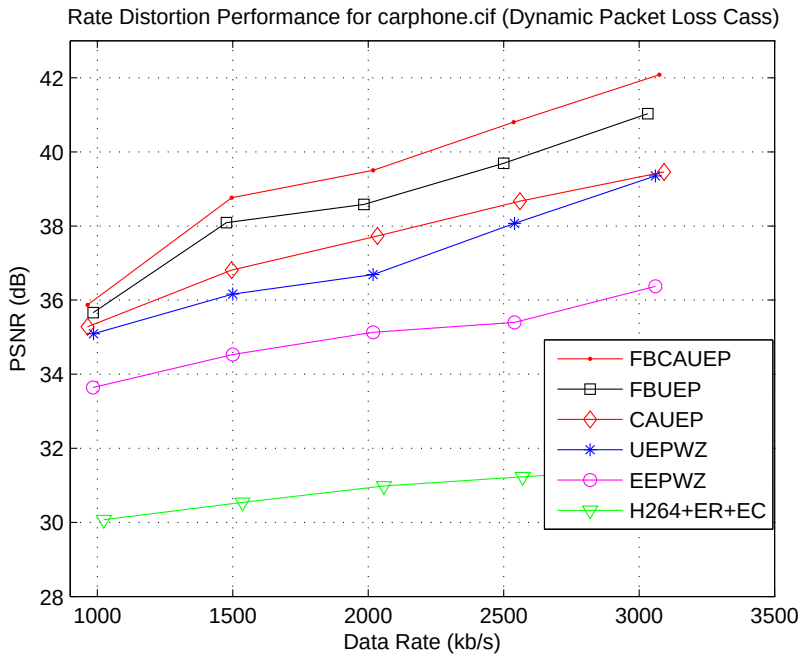

Fig. 5.9. Rate Distortion Performance (carphone.cif) (Dynamic Packet Loss Case)

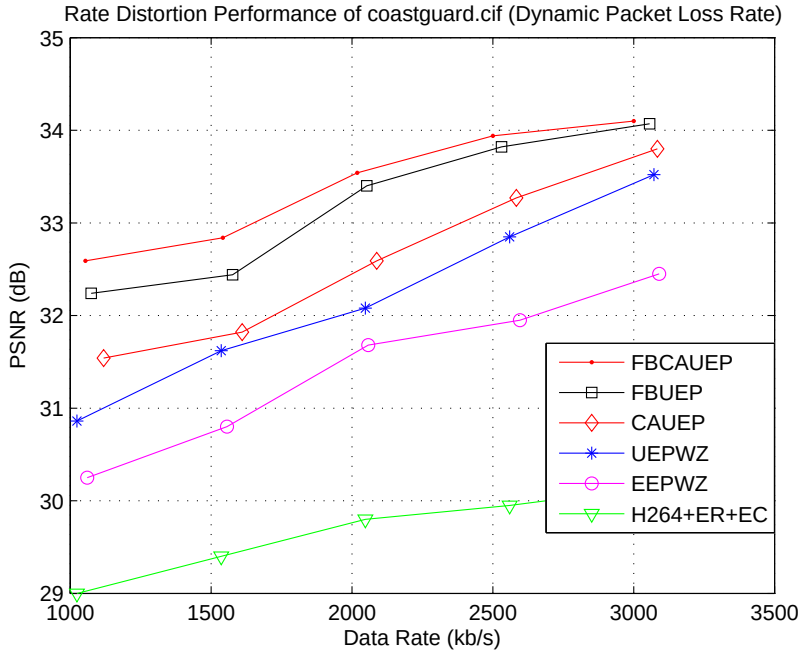

Fig. 5.10. Rate Distortion Performance (coastguard.cif) (Dynamic Packet Loss Case)

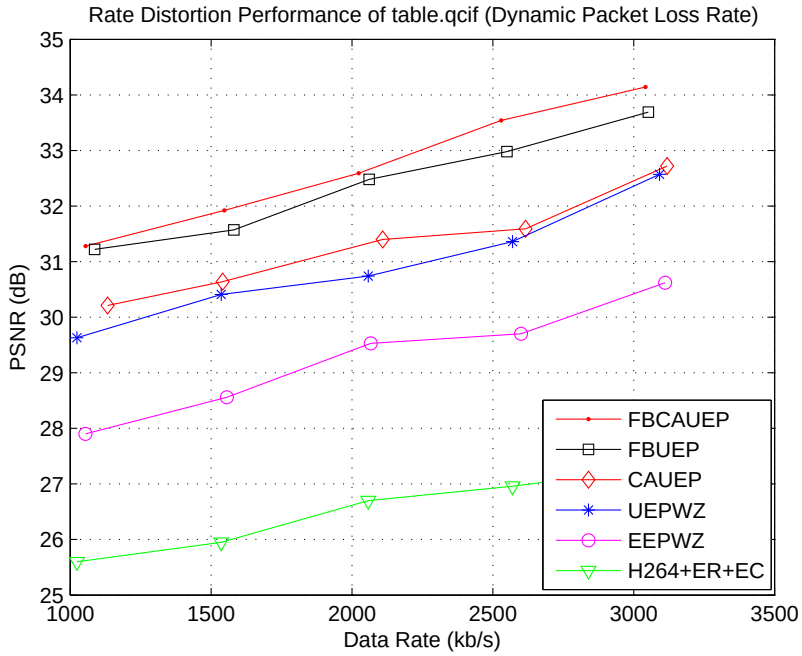

Fig. 5.11. Rate Distortion Performance (table.cif) (Dynamic Packet Loss Case)

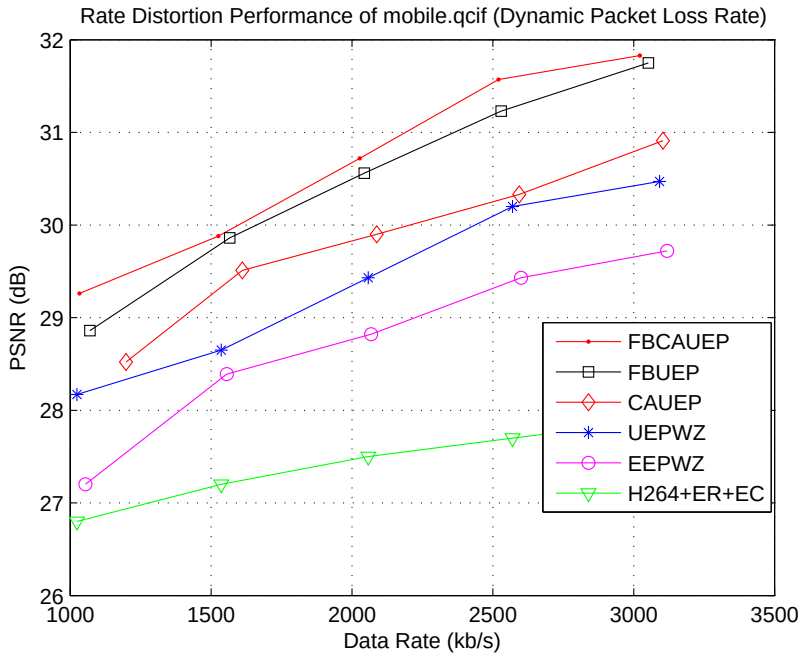

Fig. 5.12. Rate Distortion Performance (mobile.cif) (Dynamic Packet Loss Case)

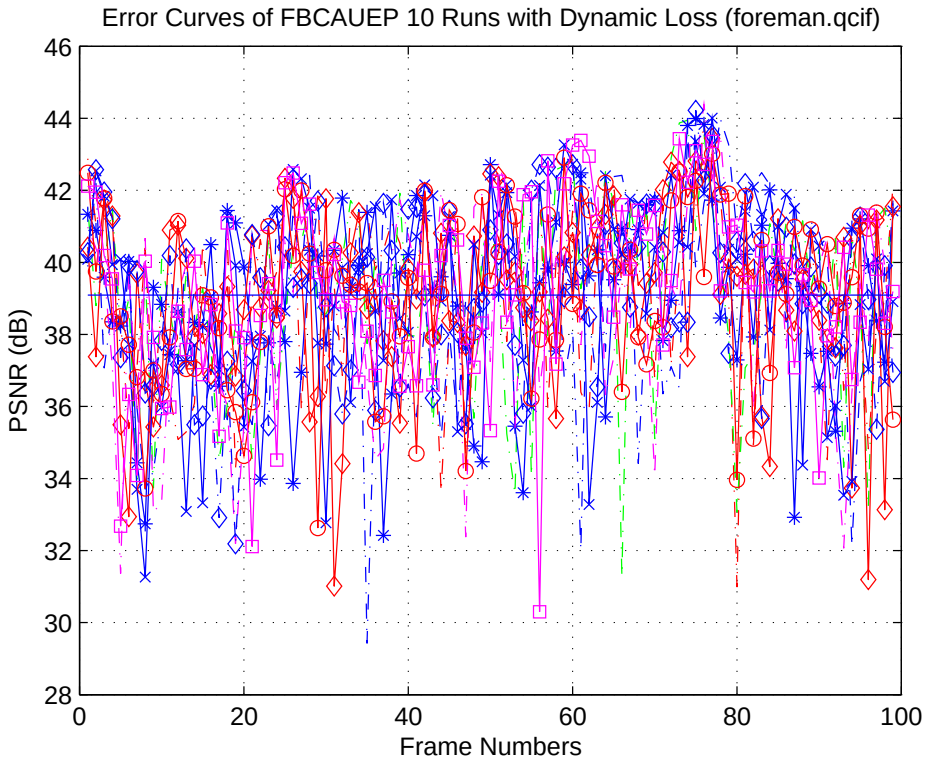

Fig. 5.13. Error Curves of FBCAUEP 10 Runs with Dynamic Loss (foreman.qcif)

Fig. 5.14. FBCAUEP: Visual Quality of the 97th frame in *table* sequence at Dynamic Packet Loss

Fig. 5.15. FBUEP: Visual Quality of the 97th frame in *table* sequence at Dynamic Packet Loss

Fig. 5.16. FBCAUEP: Visual Quality of the 33th frame in *stefan* sequence at Dynamic Packet Loss

Fig. 5.17. FBUEP: Visual Quality of the 33th frame in *stefan* sequence at Dynamic Packet Loss

Table 5.1
Parity Data Rate Allocation for FBCAUEP

Feedback Messages	Frame Content	Parity Data Rate (PDR)
$0 < N \leq 11\%$	$SAD^{Res} \leq T_1^{Res}$ $SAD^{MV} \leq T_1^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{2}{16}$
$0 < N \leq 11\%$	$T_1^{Res} < SAD^{Res} \leq T_2^{Res}$ $T_1^{MV} < SAD^{MV} \leq T_2^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{3}{16}$
$0 < N \leq 11\%$	$T_2^{Res} < SAD^{Res} \leq T_3^{Res}$ $T_2^{MV} < SAD^{MV} \leq T_3^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{4}{16}$
$0 < N \leq 11\%$	$SAD^{Res} > T_3^{Res}$ $SAD^{MV} > T_3^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{5}{16}$
$11\% < N \leq 22\%$	$SAD^{Res} \leq T_1^{Res}$ $SAD^{MV} \leq T_1^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{3}{16}$
$11\% < N \leq 22\%$	$T_1^{Res} < SAD^{Res} \leq T_2^{Res}$ $T_1^{MV} < SAD^{MV} \leq T_2^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{4}{16}$
$11\% < N \leq 22\%$	$T_2^{Res} < SAD^{Res} \leq T_3^{Res}$ $T_2^{MV} < SAD^{MV} \leq T_3^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{5}{16}$
$11\% < N \leq 22\%$	$SAD^{Res} > T_3^{Res}$ $SAD^{MV} > T_3^{MV}$	$PDR_{TC} = 0; PDR_{MV} = \frac{6}{16}$
$22\% < N \leq 33\%$	$SAD^{Res} \leq T_1^{Res}$ $SAD^{MV} \leq T_1^{MV}$	$PDR_{TC} = \frac{2}{16}; PDR_{MV} = \frac{4}{16}$
$22\% < N \leq 33\%$	$T_1^{Res} < SAD^{Res} \leq T_2^{Res}$ $T_1^{MV} < SAD^{MV} \leq T_2^{MV}$	$PDR_{TC} = \frac{2}{16}; PDR_{MV} = \frac{7}{16}$
$22\% < N \leq 33\%$	$T_2^{Res} < SAD^{Res} \leq T_3^{Res}$ $T_2^{MV} < SAD^{MV} \leq T_3^{MV}$	$PDR_{TC} = \frac{3}{16}; PDR_{MV} = \frac{6}{16}$
$22\% < N \leq 33\%$	$SAD^{Res} > T_3^{Res}$ $SAD^{MV} > T_2^{MV}$	$PDR_{TC} = \frac{3}{16}; PDR_{MV} = \frac{8}{16}$
$33\% < N \leq 44\%$	$SAD^{Res} \leq T_1^{Res}$ $SAD^{MV} \leq T_1^{MV}$	$PDR_{TC} = \frac{2}{16}; PDR_{MV} = \frac{7}{16}$
$33\% < N \leq 44\%$	$T_1^{Res} < SAD^{Res} \leq T_2^{Res}$ $T_1^{MV} < SAD^{MV} \leq T_2^{MV}$	$PDR_{TC} = \frac{3}{16}; PDR_{MV} = \frac{8}{16}$
$33\% < N \leq 44\%$	$T_2^{Res} < SAD^{Res} \leq T_3^{Res}$ $T_2^{MV} < SAD^{MV} \leq T_3^{MV}$	$PDR_{TC} = \frac{4}{16}; PDR_{MV} = \frac{7}{16}$
$33\% < N \leq 44\%$	$SAD^{Res} > T_3^{Res}$ $SAD^{MV} > T_2^{MV}$	$PDR_{TC} = \frac{4}{16}; PDR_{MV} = \frac{8}{16}$
$44\% < N$	$SAD^{Res} \leq T_1^{Res}$ $SAD^{MV} \leq T_1^{MV}$	$PDR_{TC} = \frac{3}{16}; PDR_{MV} = \frac{8}{16}$
$44\% < N$	$T_1^{Res} < SAD^{Res} \leq T_2^{Res}$ $T_1^{MV} < SAD^{MV} \leq T_2^{MV}$	$PDR_{TC} = \frac{4}{16}; PDR_{MV} = \frac{8}{16}$
$44\% < N$	$T_2^{Res} < SAD^{Res} \leq T_3^{Res}$ $T_2^{MV} < SAD^{MV} \leq T_3^{MV}$	$PDR_{TC} = \frac{4}{16}; PDR_{MV} = \frac{10}{16}$
$44\% < N$	$SAD^{Res} > T_3^{Res}$ $SAD^{MV} > T_2^{MV}$	$PDR_{TC} = \frac{8}{16}; PDR_{MV} = \frac{10}{16}$

Fig. 5.18. FBCAUEP: Visual Quality of the 93th frame in *foreman* sequence at Dynamic Packet Loss

Fig. 5.19. FBUEP: Visual Quality of the 93th frame in *foreman* sequence at Dynamic Packet Loss

6. CONCLUSIONS

6.1 Contribution of The Thesis

In this thesis, we developed error resilience techniques based on Wyner-Ziv coding [21], [22], [23], [25], [26]. The main contribution of the thesis are:

- Unequal Error Protection Based on Wyner–Ziv Coding

  Our work on unequal error protection [21] using Wyner–Ziv coding is motivated by the SLEP system. However, there are two major differences from SLEP. First, instead of protecting everything associated with coarsely reconstructed frames, we separately input motion information and the transform coefficients from a primary H.264 encoder to our Wyner–Ziv encoder and protect them independently. Second, since motion information and the transform coefficients are processed independently, different parity data rates can be assigned according to their impact on the quality of the decoded frames. This results in an efficient way of protecting the video elements based on their importance level. Experimental results show that proposed scheme significantly improved the visual quality of corrupted frames at the expense of a small increase in data rate.

- Content Adaptive Unequal Error Protection Based on Wyner–Ziv Coding

  In this work, we improved the performance of our unequal error protection technique by adapting the parity data rates of the protected video information to the content of each frame [22]. A content adaptive function was used to evaluate the sum of the absolute difference (SAD) between the reconstructed frame and the predicted frame. Depending on pre-selected thresholds, the parity data rates assigned to the motion information and the transform coefficients were varied for each frame. This resulted in a more effective and flexible er-

ror resilience technique that had an improved performance compared to our original work.Experimental results show that the proposed scheme significantly improved the quality of corrupted frames and efficiently utilized the available data rate.

- Feedback Aided Unequal Error Protection

  In this work, we developed a feedback aided error resilience technique based on the previously proposed unequal error protection method. At the decoder, the current packet loss rates are estimated based on the received data and sent back to the Wyner–Ziv encoder via the real-time transport control protocol (RTCP) feedback mechanism. This is utilized by the Turbo encoder, as a Slepian–Wolf lossless coder inside the Wyner–Ziv codec, to update the parity data rates of the motion information and the transform coefficients, which are still protected independently. At the Wyner–Ziv decoder, the received parity bits together with the side information from the primary decoder are used to decode the corrupted slices. These in turn are sent back to the primary decoder to replace their corrupted counterparts. It is to be noted that simply increasing the parity slices when the packet loss rate increases is not applicable, since it will exacerbate network congestion [24], [34], [35]. Instead, the total transmission data rate should be kept constant, which means that when the packet loss rate increases the primary data transmission rate should be lowered in order to spare more bits for parity bit transmission. Also, considering the advantage of unequal error protection, when the packet loss rate is high, more data rate should be allocated to motion information parity data rate since motion vectors mostly have higher importance than the transform coefficients. Whenever feedback is used in a communication system, the amount of delay it incurs needs to be taken in consideration. In our feedback system, an RTCP packet is sent back to the Wyner–Ziv encoder by using the immediate/early RTCP feedback mode, whenever possible. The immediate RTCP feedback mode is the mode that is

capable of reporting every loss event immediately back to the receiver. However, the success of sending the feedback message in such a timely manner also depends on the available bandwidth [36], [37]. The proposed method resulted in an efficient allocation of the data rate by the Wyner–Ziv codec for the purpose of error resilience and consequently provided good quality decoded video when data had been corrupted by transmission errors.

- Feedback Aided Content Adaptive Unequal Error Protection

  In this work, we studied the case of combining the content adaptive function in the encoder side and the channel loss feedback aided from the decoder side. In this method, a new content adaptive function directly analyzing the statistics of the motion information and the transform coefficients is implemented in the Wyner-Ziv encoder. The improved rate distortion performance is shown from the experiment results. Further more, the analysis and the modeling of the whole system is studied in order to find an optimal data rate allocation. The experiment results showed the improved performance on both the rate distortion and the visual quality of the decode video frames.

6.2 Future Work

In this work, we aimed to develop efficient and effective error resilience techniques consuming as less data rate as possible for protecting the important video data via Wyner-Ziv coding. The unequal error protection, content adaptive unequal error protection, feedback aided unequal error protection and feedback aided content adaptive unequal error protection have been proposed in our work and the experimental results showed the significantly improved visual quality in the decoded video frames when the packet loss happened during video stream transmission.

In order to further reduce the data rates consumed by encoding the motion information and the transform coefficients, an error resilient entropy coding method

should be applied on the protected video data before they are processed by the turbo coders. This will be studied in our future work.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Y.Wang and Q.F.Zhu, "Error control and concealment for video communication: A review," *Proceedings of the IEEE*, vol. 86, pp. 974–997, May 1998.

[2] D.Wyner and J.Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, January 1976.

[3] A.Aaron, S.Rane, R.Zhang, and B.Girod, "Wyner-ziv coding for video - applications to compression and error resilience," in *Proceedings of IEEE Data Compression Conference*, (Snowbird Utah), pp. 93–102, March 2003.

[4] A.Aaron, S.Rane, D.Rebollo, and B.Girod, "Systematic lossy forward error protection for video waveforms," in *Proceedings of International Conference on Image Processing*, (Spain), September 2003.

[5] S.Rane, A.Aaron, and B.Girod, "Systematic lossy forward error protection for error resilient digital video broadcasting," in *Proceedings of SPIE Visual Communications and Image Processing*, (San Jose, CA.), January 2004.

[6] S.Rane and B.Girod, "Systematic lossy error protection versus layered coding with unequal error protection," in *Proceedings of SPIE Visual Communications and Image Processing*, (San Jose, CA), January 2005.

[7] S. Rane and B. Girod, "Analysis of error-resilient video transmission based on systematic source-channel coding," in *Picture Coding Symposium*, (San Francisco, CA), December 2004.

[8] S. Rane, A. Aaron, and B. Girod, "Systematic lossy forward error protection for error resilience digital video broadcasting - a wyner-ziv coding approach," in *Proceedings of IEEE International Conference on Image Processing*, (Singapore), October 2004.

[9] S.Rane, A.Aaron, and B.Girod, "Error resilient video transimission using multiple embedded wyner-ziv descriptions," in *Proceedings of IEEE International Conference on Image Processing*, (Italy), September 2005.

[10] S. Rane and B. Girod, "Systematic lossy error protection of video based on h.264/avc redundant slices," in *Proceedings of Visual Communication and Image Processing*, (San Jose, CA), January 2006.

[11] S.Rane, P.Baccichet, and B.Girod, "Modeling and optimization of a systematic lossy error protection system," in *Proceedings of International Picture Coding Symposium*, (Beijing, China), April 2006.

[12] P. Baccichet, S. Rane, and B. Girod, "Systematic lossy error protection using h.264/avc redundant slices and flexible macroblock ordering," *Journal of Zheijang University*, vol. 7, pp. 727–736, May 2006.

[13] B. Girod, A. M. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proceedings of the IEEE*, vol. 93, January 2005.

[14] R. Puri, A. Majumdar, and K. Ramchandran, "Prism: A video coding paradigm with motion estimation at the decoder," *IEEE Transactions On Image Processing*, vol. 16, pp. 2436–2448, October 2007.

[15] A. Seghal, A. Jagmohan, and N. Ahuja, "Wyner-ziv coding of video: an error resilient compression framework," *IEEE Transactions On Multimedia*, vol. 6, pp. 249–258, April 2004.

[16] J. Wang, A. Majumdar, and K. Ramchandran, "Robust video transmission over a lossy network using a distributed source coded auxiliary channel," in *Picture Coding Symposium*, 2004.

[17] J. Wang, V. Prabhakaran, and K. Ramchandran, "Syndrome-based robust video transmission over networks with bursty losses," in *IEEE International Conference on Image Processing*, 2006.

[18] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *International Conference on Communications*, (Geneva, Switzerland), pp. 1064–1070, May 23-26 1993.

[19] C. Berrou and A. Glavieux, "Near optimium error correcting coding and decoding: Turbo codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261 – 1271, October 1996.

[20] Q. Xu, V. Stankovic, and Z. Xiong, "Layered wyner-ziv video coding for transmission over unreliable channels," *Signal Processing: Special Section on Distributed Source Coding*, vol. 86, pp. 3212–3225, November 2006.

[21] L. Liang, P. Salama, and E. J. Delp, "Unequal error protection using wyner-ziv coding," in *Proceedings of SPIE Visual Communications and Image Processing*, (CA), January 2007.

[22] L. Liang, P. Salama, and E. J. Delp, "Adaptive unequal error protection based on wyner–ziv coding," in *Picture Coding Symposium 2007*, (Lisbon, Portugal), November 2007.

[23] L. Liang, P. Salama, and E. J. Delp, "Feedback aided error resilience technique based on wyner-ziv coding," in *Proceedings of SPIE Visual Communications and Image Processing*, (CA), January 2008.

[24] M. Johanson, "Adaptive forward error correction for real-time internet video," in *Proceedings of PV2003*, (Nantes, France), April 2003.

[25] L. Liang, P. Salama, and E. J. Delp, "Feedback aided content adaptive unequal error protection based on wyner-ziv coding," in *Picture Coding Symposium 2009*, (Chicago, United States), May 2009.

[26] L. Liang, P. Salama, and E. J. Delp, "Unequal error protection techniques based on wyner–ziv coding," *EURASIP Journal on Image and Video Processing – Special Issue on Distributed Video Coding*, 2009.

[27] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, July 2003.

[28] I.E.G.Richardson, *H.264 and MPEG-4 Video Compression*. England: Wiley, 2003.

[29] E. Setton and B. Girod, "Rate-distortion analysis and streaming of sp and si frames," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, pp. 733–743, June 2006.

[30] M. Karczewicz and R. Kurceren, "The sp and si frames design for h.264/avc," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 637–644, July 2003.

[31] S. Wenger, "H.264/avc over ip," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 645–656, July 2003.

[32] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/avc in wireless environments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 657–673, July 2003.

[33] A.Aaron and B.Girod, "Compression with side information using turbo codes," in *Proceedings of IEEE Data Compression Conference*, (Snowbird, Utah), April 2002.

[34] B.Girod and N. Farber, "Feedback-based error control for mobile video transmission," *Proceedings of the IEEE*, vol. 87, pp. 1707–1723, October 1999.

[35] X. Zhu and B. Girod, "Video streaming over wireless networks," in *Proc. European Signal Processing Conference*, (EUSIPCO-07, Poznan, Poland), September 2007.

[36] J. Ott, S. Wenger, N. Sato, C. Burmeister, and J. Ray, "Extended rtp profile for real-time transport control protocol (rtcp) - based feedback (rtp/avpf)," July 2006.

[37] C. Burmeister, R. Hakenberg, A. Miyazaki, J. Ott, N. Sato, and S. Fukunaga, "Extended rtp profile for real-time transport control protocol (rtcp) - based feedback (rtp/avpf): Results of the timing rule simulations," July 2006.

[38] W. E. Ryan, "Concatenated codes and iterative decoding," in *Wiley Encyclopedia of Telecommunications*, (New York), Wiley and Sons, 2003.

[39] J. G. A. Y. J. Liang and B. Girod, "Analysis of packet loss for compressed video: Does burst loss matter?," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, (ICASSP-03, Hongkong, China), April 2003.

[40] K. Stuhlmuller, N. Farber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1012–1032, June 2000.

[41] D. Wu, Y. T. Hou, W. Zhu, Y. Zhang, and H. J. Chao, "Mpeg-4 compressed video over the internet," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS'99)*, (Orlando, USA), May 1999.

[42] Y. T. Hou, D. Wu, W. Zhu, and H. Lee, "An end-to-end architecture for mpeg-4 video streaming over the internet," in *IEEE Int. Conference on Image Processing*, October 1999.

[43] J. Ott, S. Wenger, N. Sato, C. Burmeister, and J. Ray, "Extended rtp profile for real-time transport control protocol (rtcp) - based feedback (rtp/avpf)," July 2006.

[44] P.Baccichet, S. Rane, and B.Girod, "Systematic lossy error protection using h.264/avc redundant slices and flexible macroblock ordering," *Journal of Zhejiang University*, vol. 7, pp. 727–736, May 2006.

VITA

VITA

Liang Liang received her Master's degree in Aeronautics and Astronautics from Purdue University, West Lafayette, USA, in 2004, and Bachelor's degree in Electrical Engineering from Civil Aviation University of China, Tianjin, China, in 2002. She is receiving her Ph.D degree on Video Processing from School of Electrical and Computer Engineering at Purdue University, West Lafayette, USA, in 2009.