INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Combined Forward Error Correction and Error Concealment for Digital Video Transmission

Yan Mei

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montréal, Québec, Canada

June 2001

ABSTRACT

Combined Forward Error Correction and Error Concealment for Digital Video Transmission

Yan Mei

Forward Error Correction (FEC) coding has been widely used in digital communications to improve the transmission performance over noisy channels. Based on the introduced redundancy and encoding structure, a maximum likelihood (ML) decoding scheme can be used to correct significant part of transmission errors. For digital video transmission, further enhancement of the reconstructed video signal can be done by error concealment based on the residual redundancy of the video signal. FEC and error concealment are usually treated as independent processes.

This thesis presents an error resilience scheme, which combines Forward Error Correction (FEC) and Error Concealment (EC) in order to further enhance the quality of the video signal transmitted over a noisy channel. Of particular, for a given FEC scheme in use, a Multiple Candidate Likelihood (MCL) channel decoding strategy is proposed. Unlike the traditional ML decoder that provides only one corrected sequence, the proposed MCL decoding strategy offers a number of candidates with their reliability information. These candidates are further examined in terms of syntax/semantic validity and their discontinuity measure, which are related to the video source coding.

Performance of video signals in terms of the peak signal-to-noise ratio (PSNR) as well as subjective measure by visual inspection of video frames and sequence is investigated for different video sequences and various FEC coding schemes. Both simulation and analysis are used to evaluate performance of the proposed schemes and to assess their complexity. Research results indicate that the proposed strategy outperforms the traditional ML decoding technique for the same FEC code in use, especially at the high channel bit error rates (BER). Simulation results are in a good agreement with the analytical prediction. Selection of key parameters and characteristics of the proposed combined FEC and EC are discussed.

Keywords: Video transmission, error concealment, forward error correction, multiple candidate likelihood decoder, syntax, discontinuity

Dedicated to my father and mother .....

ACKNOWLEDGEMENTS

First I would like to express my deepest appreciation to my advisors Dr. Le-Ngoc and Dr. Lynch for proposing the research topic, providing a continuous support and directing this work. I would also like to thank all the professors with whom I have interacted during my studies at Concordia University.

I also would like to thank my family and my boyfriend Xu. Their love, trust and support throughout all these years made the impossible possible. I am grateful to you.

Last but not least, thanks to all my friends in Concordia for their constant moral support and for sharing with me a challenging experience.

Thanks also for financial supports from the Natural Sciences and Engineering Research Council of Canada.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

ATM	Asynchronous Transmission Mode
BE	Bandwidth Expansion
BER	Bit Error Rate
BSC	Binary Symmetric Channel
DC	Director Current
DCT	Discrete Cosine Transform
EC	Error Concealment
FEC	Forward Error Correction
FLC	Fixed Length Code
GF	Galois Field
GOP	Group Of Pictures
HDTV	High Definition TV
IDCT	Inverse Discrete Cosine Transform
ISO	International Standard Organization
MAP	Maximum A Posteriori
MAD	Mean Absolute Difference
MB	Macroblock
MCL	Multiple Candidate Likelihood
ML	Maximum Likelihood
MPEG	Moving Picture Experts Group
MSE	Mean Squared Error
PSNR	Peak Signal to Noise Ratio
SBEC	Syntax Based Error Concealment
SDBEC	Syntax and Discontinuity Based Error Concealment
SNR	Signal to Noise Ratio
SSG0	Search Space Group G0
SSG1	Search Space Group G1
SSG2	Search Space Group G2
VLC	Variable Length Code
VLSI	Very Large Scale Integration
VQ	Vector Quantization

Chapter 1

Introduction

1.1 Transmission of compressed video

With the rapid deployment and growth of communication systems, image and video transmission is a particularly important and challenging application that deserves attention. Video signals require a huge amount of resources to transmit when compared to audio or text information. For example, for a one-second sequence at 30 frames/sec and 704*480 pixels for the luminance component [1], which is closely related to the perception of brightness, and subsampled in both spatial directions for the color components (4:2:0 format), which are related to the perception of color hue and saturation, requires about 15 Mbytes or 120 Mb/s. Despite the development of broadband networks, compression techniques are needed to reduce the required transmission bandwidth. Even assuming the compression ratio is only 15:1, the transmission speed for the video above would be reduced to 8 Mb/s. If the compression ratio is higher, i.e., 25:1 or 40:1, the required transmission bandwidth will be lower.

Compression is likewise important when storing image or video signals despite the availability of larger storage devices. Modern image and video compression techniques offer the possibility to store and transmit the vast amount of data necessary to represent digital images and video in an efficient way. The video compression algorithms developed by the Moving Pictures Expert Group(MPEG) [2] have developed into important and successful video coding standards worldwide. The group has produced the MPEG-1, MPEG-2 and MPEG-4 standards. An increasing number of MPEG VLSI chip-sets and products are becoming available on the market [3]. This has led to a wide range of applications [4], such as video on demand, digital TV/HDTV, terrestrial and satellite broadcasting, multimedia, image/video database services.

While storage devices are generally reliable, communication networks often introduce errors. Two types of errors that may occur are cell loss and data corruption [5]. In packet switched networks, cell loss due to network congestion is an important source of transmission errors. When several sources transmit at their peak rates

simultaneously, the buffer space at some switches may be inadequate. The congestion at those switches will lead to cell loss due to buffer overflow. Thus in cell loss many bits often from the same spatial-temporal area of the video are lost. The large number of bits lost often results in large areas of the image being corrupted. The impact of cell loss may become more significant as the packet or cell size become larger.

Data corruption occurs when noise and/or interference in the communication channel introduces bit errors. Variable length codes (VLC's) like the Huffman code are often used in image/video compression. If an error causes the decompressor to lose track of where the current VLC begins and ends then not only that VLC but many or all subsequent VLC's may be decoded incorrectly. Thus a single bit error in a coded bitstream can cause a propagating error that can lead to objectionable degradations over a large portion of the video. Generally standards insert resynchronization points to stop indefinite propagation. Since these resynchronization points are somewhat costly in bits they occur relatively infrequently. See Section 2.1.3 for the placement of resynchronization points in MPEG2. In addition to propagation due to loss of VLC synchronization, errors may also propagate in the temporal direction because of the motion compensated predictive interframe coding.

In this thesis techniques to alleviate bit errors are studied and developed. Cell or packet loss is not considered.

Figure 1.1 shows one frame from the sequence "Table Tennis" decompressed, with a channel Bit Error Rate (BER) of 10^{-5} and 10^{-4}, respectively. A single bit error causes the lose of pixels and thus the black stripe in half a line in the first frame. The second frame contains a greater degradation due to the higher BER. Figure 1.2 shows the impact of bit errors in the temporal direction. The first figure shows frame 0, where bit errors occurred, while the second figure shows the impact of those bit errors after 12 frames.

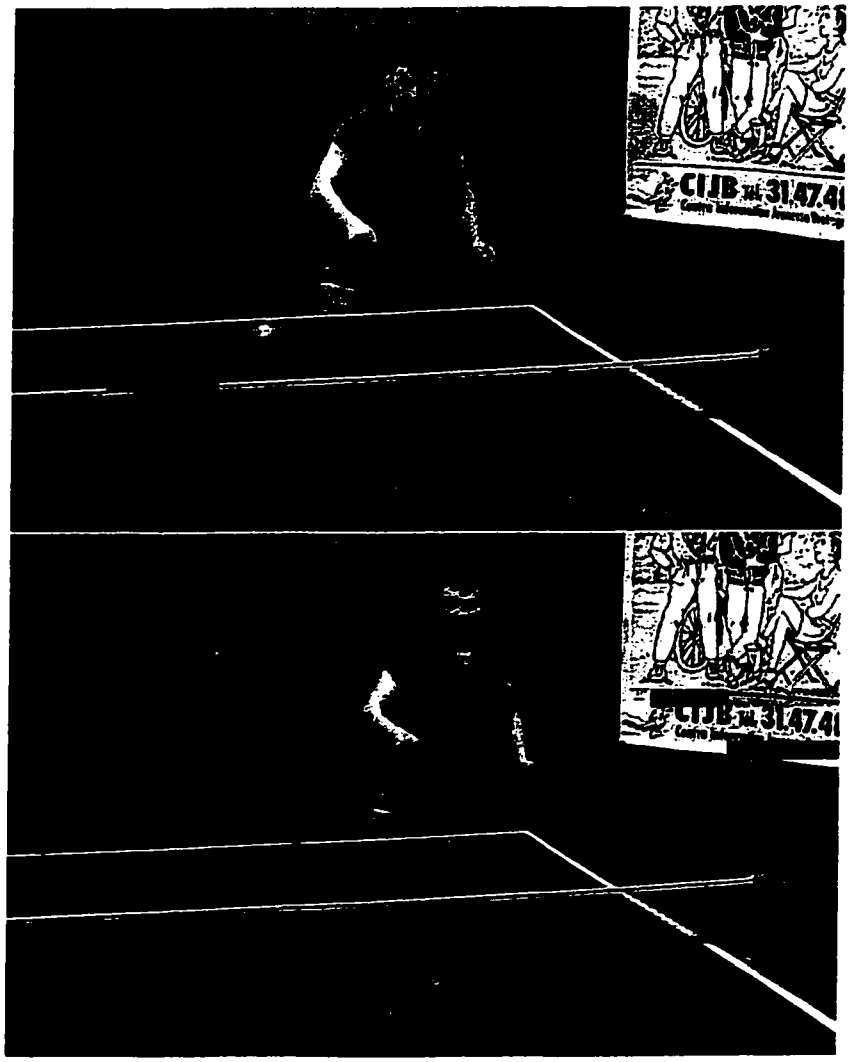

Figure 1.1: Frames with bit errors

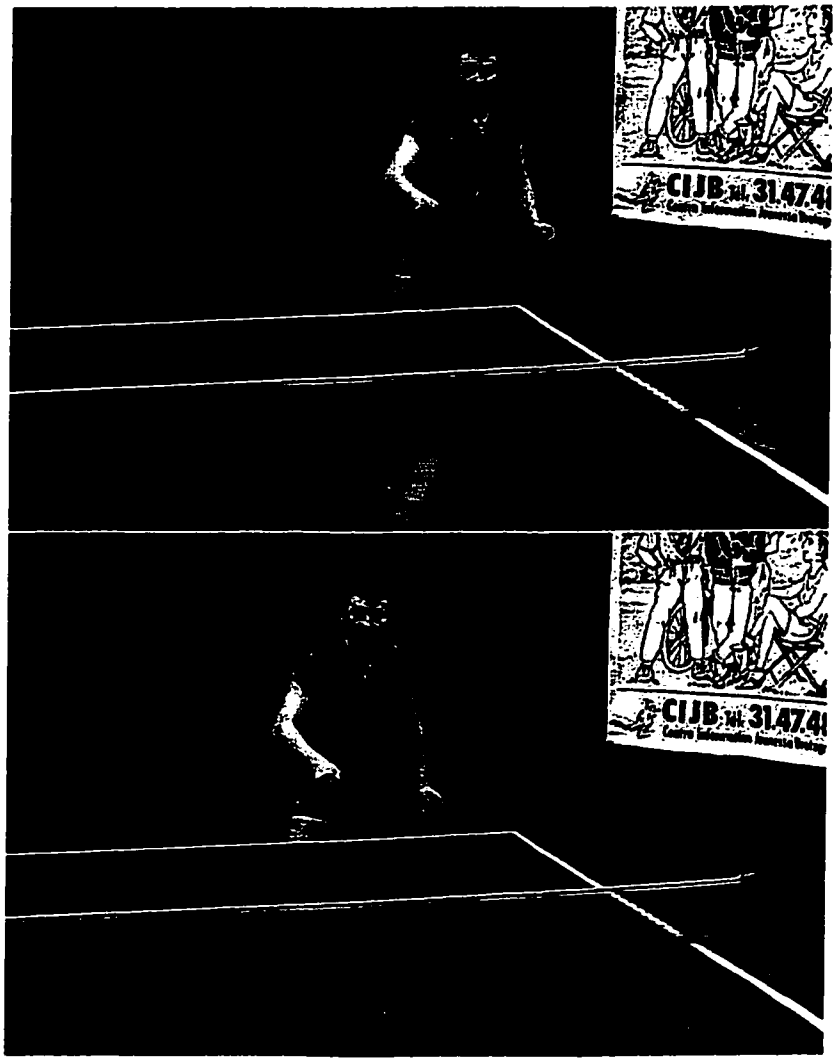

Figure 1.2: Temporal effect of bit errors

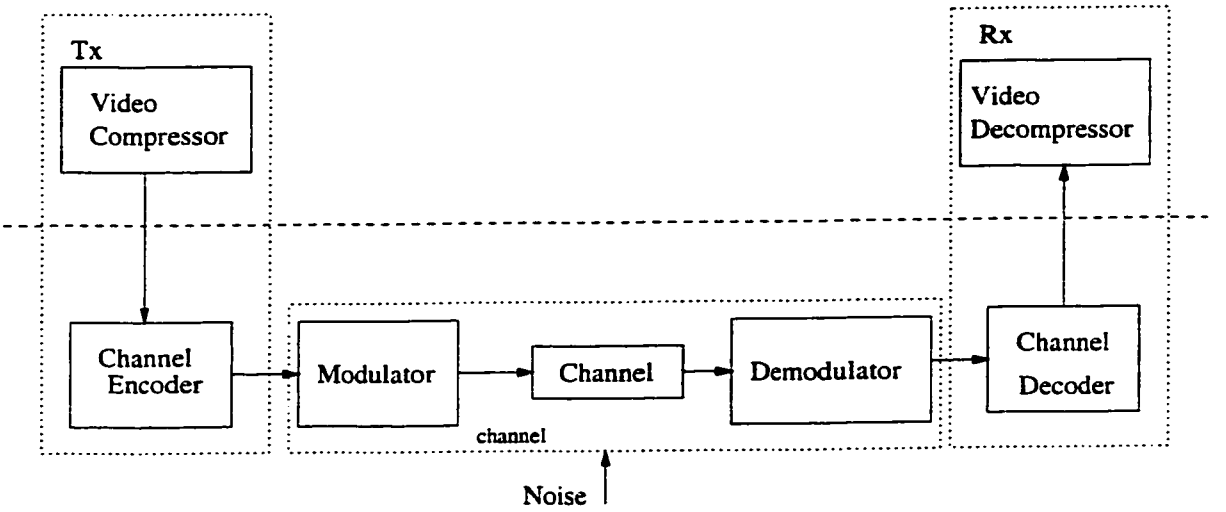

Figure 1.3: Problem statement

1.2 Problem statement

In this thesis, error resilience techniques to alleviate the effects of bit errors are studied and developed. Figure 1.3 shows the block diagram of a typical video communication systems.

Traditionally, channel encoder/decoder is used to enhance the transmission performance, independently of the video decompressor. The channel encoder adds redundancy to the transmitted data in a systematic manner so that the channel decoder can base on the encoding structure to correct errors occurring in the channel. Compressed video signals also contain residual correlation, which can be used to conceal errors. Various error concealment techniques have been proposed [6]. The channel coding/decoding and error concealment are generally treated in an independent manner.

In this work, an error resilience scheme is proposed. It combines the channel decoding and error concealment in order to further enhance the quality of the video signal transmitted over a noisy channel. Specifically, a multiple-candidate likelihood (MCL) channel decoding scheme is considered. It provides a number of candidates with their corresponding reliability information rather than only one solution. These candidates are further examined by the error concealment scheme based on the

residual redundancy of the video signal to select the final solution.

The proposed error resilience scheme aims to make use of the existing channel coding and to exploit a joint error decoding/concealment strategy, which maximizes the capability of the channel code and residual redundancy of the video signal.

1.3 Figures of Merit

Video quality or fidelity can be evaluated subjectively or objectively. At this time a rigorous subjective quality assessment requires a set of impartial human viewers (twenty five or more), either expert or non expert to rate the degraded video either alone or in relation to an "original" sequence [7, 8, 9]. Since human observers are the ultimate consumers of most video this is considered the "best" way to measure video quality. However, this can be prohibitively costly and time consuming to assess the subjective quality of video in this way, in particular at the algorithmic development and the parameter tweaking stage. There are efforts [9] to automate the subjective evaluation of video quality but they have not been successful up to now. In this thesis non-rigorous methods of subjective assessment are used. Specifically two expediencies are employed: individual frames are shown in the thesis for the readers' own judgment and the author's comments on subjective quality are inserted in the text.

Objective measures of video quality do not need to use human subjects and are generally automated (which is their main advantage). Mean square of error (MSE), peak signal to noise ratio (PSNR), mean absolute difference (MAD) are among the common objective measures. PSNR[10], which is a variation of the MSE is used as an objective measure of goodness in this work. PSNR for the luminance component[1] is defined as

$$PSNR_Y \triangleq 10 \log_{10} \frac{255^2}{\frac{1}{N_Y} \sum_{i=0}^{N_Y} (x_{oY}[i] - x_{rY}[i])^2} (dB) \tag{1.1}$$

where $x_{oY}[i]$ denotes luminance component of original image, $x_{rY}[i]$ denotes the

luminance of reconstructed image, and N_Y denotes the total number of luminance pixels. Similarly, the PSNR for the chrominance components can be defined as

$$PSNR_U \triangleq 10\log_{10}\frac{255^2}{\frac{1}{N_U}\sum_{i=0}^{N_U}(x_{oU}[i]-x_{rU}[i])^2}(dB) \tag{1.2}$$

$$PSNR_V \triangleq 10\log_{10}\frac{255^2}{\frac{1}{N_V}\sum_{i=0}^{N_V}(x_{oV}[i]-x_{rV}[i])^2}(dB) \tag{1.3}$$

where $x_{oU}[i]$, $x_{oV}[i]$ denote the blue and red chrominance components of the original image, $x_{rU}[i]$, $x_{rV}[i]$ denote those of reconstructed image, and N_U, N_V denote the number of chrominance pixels. Note that in PSNR, the peak signal value of 255 is used for the signal value. That is because traditional SNR is unfortunate in that a very bright image might mask the error since the signal part would be much larger. Thus PSNR that is the MSE put on a logarithmic scale with a suitable common value (255) put into to compare against is used.

Due to the compression distortion, the PSNR of the sequence before transmission is not infinite. Generally speaking, video with a PSNR of 40 dB is close to perfect. A picture with a PSNR 30 dB is acceptable. But when the PSNR drops below mid 25 dB, the quality of picture is not good. All of these figures can vary with the content of the video and the nature of the degradation.

In this thesis, for every simulation sequence, a PSNR is calculated and is used to draw PSNR versus Bit Error Rate (BER) curves. It should be noted that for the same sorts of impairments the PSNR and the subjective assessment of human observers will tend to track each other; however, there is no guarantee of this.

The Bandwidth Expansion(BE), defined as

$$BE = \frac{codewordsize}{datablocksize} \tag{1.4}$$

represents the increase in required bandwidth to cover extra parity added to the information prior to the transmission. It is the multiplicative inverse of the code

rate of the FEC code in use.

The computational complexity of an algorithm is also a figure of merit. Ideally the complexity can be measured in terms of number of operations. In this work it is often difficult to count the number of operations. Instead, two proxies named "number of slice candidates" and "number of bits extracted" are used for complexity. See section 3.4.

The figures of merit above are affected by the channel BER. They need to be evaluated over a range of BER's. In communications, there are close relationships between BER and transmitted power, speed, and complexity etc [11]. As BER increases, both the subjective and objective quality of the transmitted pictures would degrade. More transmitted power is needed to minimize the effects of noise. To compensate the errors, the complexity of the channel decoder/encoder would increase and the resulting delay would decrease the transmitted speed. Thus it is important to evaluate the performance over a range of BER's.

1.4 Outline of the thesis

This thesis is organized as follows:

Chapter 2 covers background materials related to the work. Important aspects of MPEG2 are introduced. Also, this chapter introduces the classic FEC approach and the general Error Concealment (EC) techniques that make use of the residual redundancy which still exists in the compressed bitstream.

Chapter 3 introduces the techniques that make use of the syntax and discontinuity information and presents the proposed combined FEC/EC scheme. The components of the proposed scheme are presented in detail.

Chapter 4 evaluates the performance of the technique by simulation results, including subjective and objective evaluations for a range of BER's. Experiments to fix parameters of the developed technique are presented. Computational complexity is also discussed.

Chapter 5 concludes the thesis by summarizing the proposed schemes and discussing their performance. Future work is also included in this chapter.

Chapter 2

Error Resilience of Compressed Video over Noisy Channel: An Overview

This chapter briefly covers the necessary background in video compression, transmission and error concealment. Section 2.1 gives an overview of the MPEG2 video compression standard. Classical FEC is introduced in Section 2.2. Section 2.3 covers some error concealment schemes that use the residual redundancy in the video. Section 2.4 gives a survey of joint FEC/EC schemes that combines the error concealment along with the error control coding.

2.1 MPEG2 video standard

This section gives an overview of the MPEG2 video standard. Section 2.1.1 describes the context of the MPEG standards. The MPEG2 video encoder and decoder are introduced in Section 2.1.2 and 2.1.3 respectively. In Section 2.1.4, important information about the structure of MPEG2 bitstream is presented.

2.1.1 Context of the MPEG standard

In 1988 the International Standard Organization (ISO) formed the Moving Picture Experts Group (MPEG) to develop a standard for video and associated audio on digital storage media[12]. The group has produced MPEG-1, MPEG-2 and MPEG-4. The current thrust is MPEG-7 "Multimedia Content Description Interface", scheduled for completion in July 2001. The MPEG activities cover more than video compression since the compression of the associated audio and the issue of the audio visual synchronization can not work independently of the video compression [8, 1, 13].

MPEG-1 is the standard on which such products as Video CD and MP3 are based. It was initially intended for video coding at bit rates of about 1.2 Mb/s with stereo audio coding at bit rates of around 250 Kb/s. It assumes low errors rate and ignores interlaced TV. The standard consists of the following part: system, video, audio, conformance and software [13].

MPEG-2 is the follow-on to MPEG-1 and is the standard on which such products as Digital Television set top boxes and DVD are based. It builds upon the

MPEG-1 capabilities. Five different profiles have been defined for MPEG-2 video , providing particular sets of capabilities ranging from function close to MPEG-1 to the very advanced video compression techniques needed for HDTV. The features offered by MPEG-2 includes: random access, trick modes, multicast to many terminal types, multiple audio and video, and compatible stereoscopic 3D [1, 13].

MPEG-2 systems includes all of the features of MPEG-1 with enhancements that allow it to operate in more error-prone environment. In addition, more video and audio streams can be handled and they are no longer required to have a common time base. MPEG-2 systems still use the pack and packet concepts, but combines them in both backwards compatible and noncompatible ways. Some of the noncompatible ways support asynchronous transmission mode (ATM) applications [1].

MPEG-2 audio is forward and backward compatible with MPEG-1. The requirement to backwards compatible necessitated some coding efficiency compromises. The nonbackwards compatible audio is also being developed [1].

Originally MPEG-2 video was primarily intended for coding of interlaced video of standard TV resolution with good quality in the bit rate range of 4 to 9 Mb/s. However, the scope of MPEG-2 video was considerably revised to include video of higher resolutions such as HDTV at higher bit rate as well as hierarchical video coding for a range of applications. Furthermore, it also supports coding for interlaced video of a variety of resolutions. MPEG-2 video maintains all of the MPEG-1 video syntax, but uses extensions to add additional flexibility and functions. "Scalable" extensions are added to provide video data streams with multiple resolutions for potential normal TV and HDTV coexistence. Other scalable extensions allow the data stream to be partioned into two pieces. They also offer some temporal flexibility so that not all frames have to be reconstructed. The MPEG-2 video is a syntactic superset of the MPEG-1 video and is thus able to meet the requirement of forward compatibility. The requirement of backward compatibility is achieved via the use of scalability and supported in specific profiles [1, 13].

MPEG-1 and MPEG-2 are very similar in the viewpoint of compression. The

main difference lies in the systems portion of the MPEG-2 standard. MPEG-3 was aimed at higher bit rates (10-20 Mb/s or higher) but it was abandoned when it was clear that MPEG-2 could handle these higher bitrates.

MPEG-4, was originally targeted at very low bitrates (tens to hundreds of kb/s, or video telephone rates). MPEG-1 and MPEG-2 are both block-based compression, i.e. the pictures are partitioned into blocks and processed. MPEG-4 is instead an object-based compression scheme. Its syntax allows the enoder to segment the video into real world objects and then each of these can be compressed separately. The compressor may choose to make its real world objects blocks in which case MPEG-4 is much like MPEG-2. An important advantage of MPEG-4 is that it represents audiovisual data as a composition of multimedia objects, so that they can be transported over network channels providing a QoS appropriate for the nature of the specific media objects; and interact with the audiovisual scene generated at the receiver's end. MPEG-4 potentially offers better compression than MPEG-2. However, currentlys its main advantage is that it makes it easier to process different media objects separately and create compound media objects. The work in the future can focus on MPEG-4 bitstream [1, 12].

MPEG-1, MPEG-2 and MPEG-4 are all compression standards. However, MPEG-7 does not focus on compression. It offers a comprehensive set of audiovisual description tools so that large quantities of multimedia objects can be searched/filtered automatically in a database application. This is a challenging task given the broad spectrum of requirements and targeted multimedia applications, and the broad number of audiovisual features of importance in such context. The question of identifying and managing content is not just restricted to database retrieval applications such as digital libraries, but extends to areas like broadcast channel selection, multimedia editing, and multimedia directory services [12].

For further explanation of MPEG, see [13, 3, 1, 8, 12].

2.1.2 Video encoder

Video compression relies on the reduction of the spatial or temporal redundancies in the data. Compression techniques that reduce the spatial correlation are referred to as intraframe coding, while those that reduce the temporal correlation are called interframe techniques. The intraframe coding only uses information from a single picture. It takes advantage of the similarity of a given region of a picture to immediately adjacent areas in the same picture. The interframe coding uses the similarity or predictability from one picture to the neighbors in a sequence. In a first stage the displacement of objects between successive frames is estimated (motion estimation). Then the resulting information is exploited in an efficient interframe predictive coding (motion compensation). The motion compensated predictive interframe coding may cause the error propagation [1].

The MPEG standards do not define the encoding process. They only specify the syntax of the coded bitstream and the decoding process. Based on these requirements, the functions needed by a typical MPEG2 encoder are shown in Figure 2.1 [?, 8].

The uncompressed video is first made some preprocessing. This includes color conversion to YCbCr (luminance and chrominance), format translation (int erlace to progressive), prefiltering, and subsampling.

The input video is fed to a motion estimator that feeds a motion vector to the motion compensation, which feeds a prediction to the minus of the substractor. The prediction error is then passed through to the Discrete Cosine Transform (DCT). DCT based implementations are used because of their high decorrelation performance and the fast DCT algorithms suitable for real-time implementations are available.

The quantization stage comes after the DCT transformation stage. Quantization reduces the possible values for the DCT coefficients and the required number of bits.

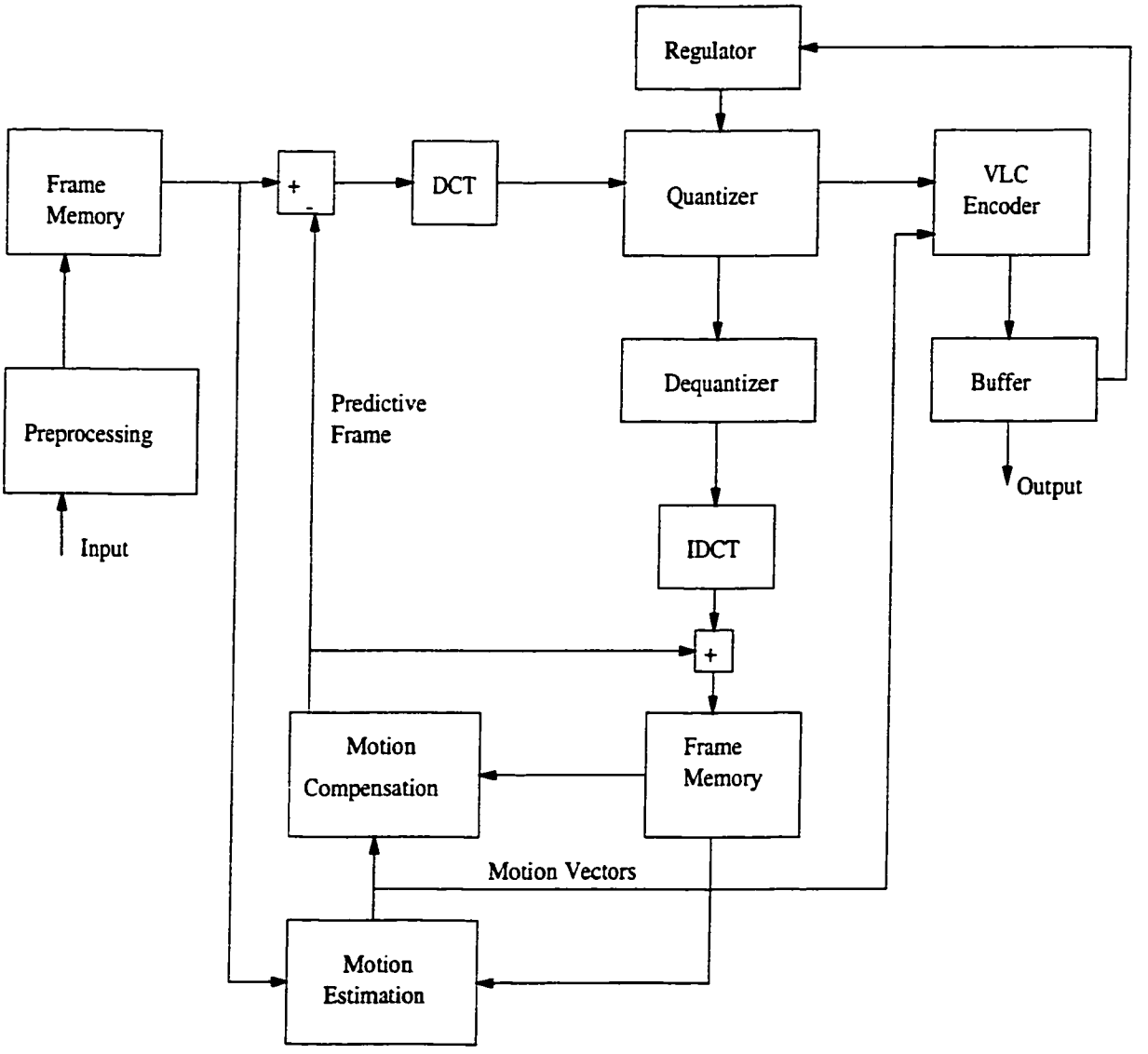

Figure 2.1: Block diagram of an MPEG encoder

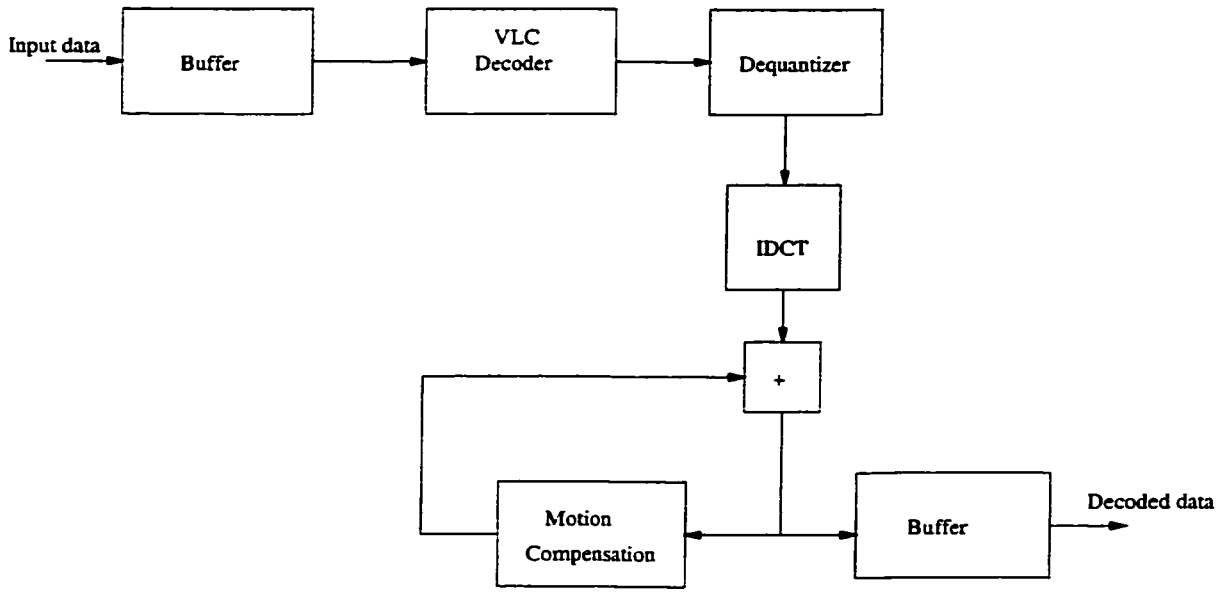

Figure 2.2: Block diagram of an MPEG2 decoder

The final compression stage starts with the serialization of the quantized coefficients. The sequence of coefficients is then coded using Variable Length Code (VLC). The way the VLC allocates code lengths depends on their expected probability of occurance. Note that the use of VLC, although it provides coding efficiency, makes the compressed bitstream fragile to errors.

Quantized levels are converted to reconstructed DCT coefficients by the dequantizer and Inverse DCT (IDCT) to produce a coded prediction error. The IDCT output is merged with the prediction to form the reconstruction, which is used by the motion estimator to estimate the motion vector. Then the motion vector is sent to the motion compensation to get the prediction.

2.1.3 Video decoder

There are many ways to implement a decoder and the standard does not recommend a particular way. The block diagram of a typical MPEG2 decoder is shown in Figure 2.2 [1, 8].

The bitstream is demultiplexed into overhead information such as motion information, quantization step size, macroblock type and quantized DCT coefficients.

The motion vectors describing the direction and amount of motion of the macroblocks are transmitted to the decoder as part of the bitstream. The decoder then knows which area of the reference picture is used for each prediction. The quantized DCT coefficients are dequantized and are input to the Inverse DCT(IDCT). The IDCT result is added to the result of the prediction to obtained the reconstructed frame. The reconstructed frame is used by the motion compensation to get the prediction.

MPEG2 is an asymmetric system. Decoders, since they merely follow the directions encoded in the bitstream, are relatively simple. Encoders, however, are much more complex than decoders and must have more intelligence.

2.1.4 Structure of the MPEG2 bitstream

In this thesis, the information about MPEG2 bitstream structure is used to do the error concealment. So it is important to clarify the structure of the MPEG2 bitstream.

A video sequence is a series of pictures taken at closely spaced intervals in time. The MPEG2 syntax for the coded video bitstream is given in Figure 2.3. It has an hierarchical representation with six layer[1]:

1. Sequence layer
2. Group of pictures layer
3. Picture layer
4. Slice layer
5. Macroblock layer
6. Block layer

The sequence layer is the top coding layer. It includes a sequence header and is followed by one or more groups of pictures. It ends with a sequence end code. The sequence header includes information like the vertical and horizontal picture size,

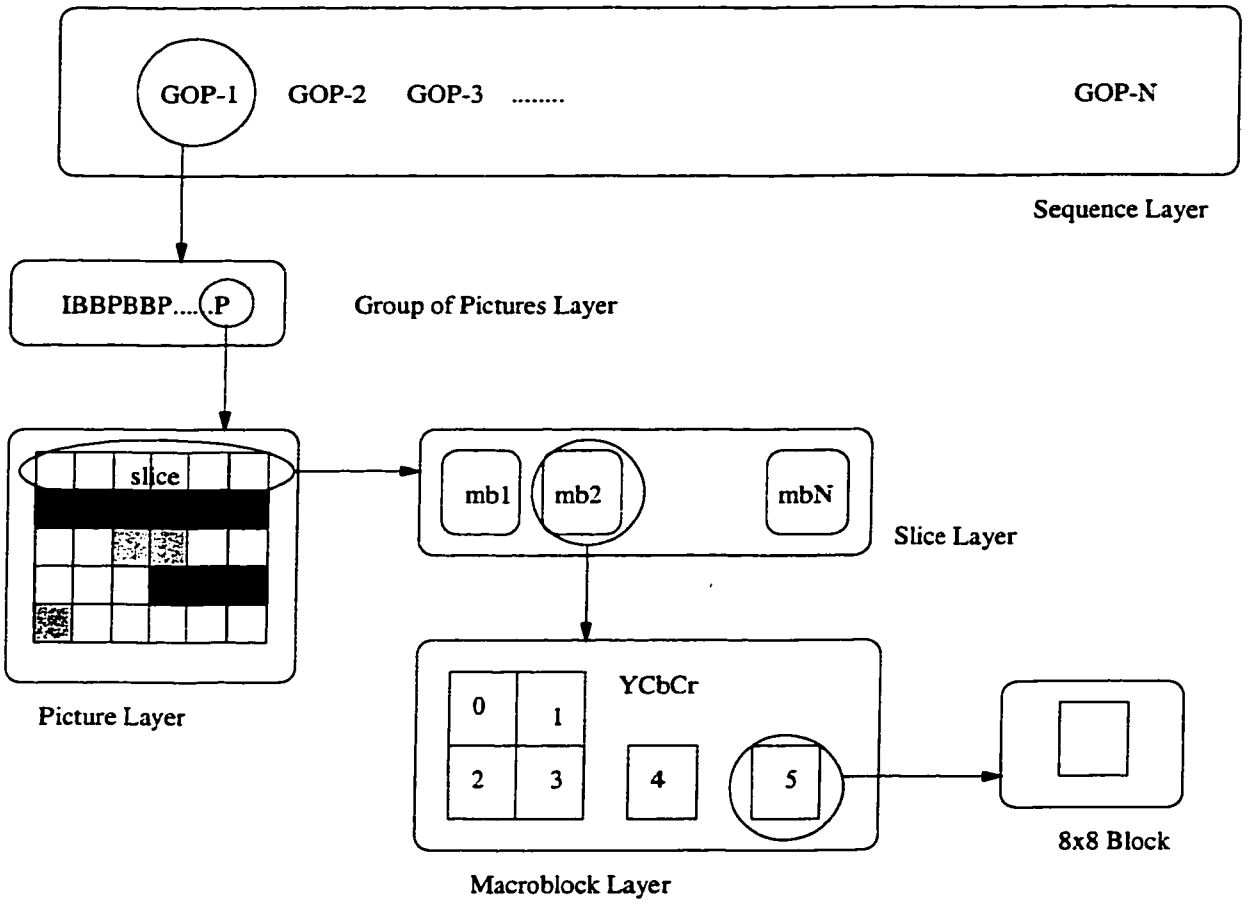

Figure 2.3: Layered structure of MPEG2 bitstream

the frame rate, the bit rate, and the minimum buffer size needed by the source decoder, etc.

A Group of Pictures (GOP) is a set of pictures in contiguous display order. MPEG2 supports three types of pictures: I-frames, P-frames and B-frames. Intra-frames (I-frames) are compressed using intraframe coding, without the need to reference to another picture. Predicted frames (P-frames) are coded with reference to the nearest previously coded picture (either I or P frame), using a motion-compensated prediction mechanism. Bidirectional predicted frames (B-frames) use both previous and future I or P frames as reference for motion estimation and compensation. A GOP must contain one I frame. The pictures in a GOP are independent of past GOP's and thus errors in those past pictures will not propagate temporally between GOP's. The header of a GOP includes timing information and user data and provides support for random access, fast search, and editing.

A picture in MPEG terminology is the basic unit of display and corresponds to a single frame in the sequence. The header of a picture provides information about the picture type, synchronization, and the resolution and range of the motion vectors.

Each picture is divided into slices. A slice is a horizontal strip of macroblocks within a frame. As shown in Figure 2.3, an MPEG2 slice can be as big as one line of macroblocks across the picture and as small as a single macroblock. It is the basic processing unit in MPEG2, and provides the last VLC resynchronization points within the bitstream. Thus error propagation due to loss of VLC synchronization should not propagate between slices. A slice header contains information for its position within a picture and a quantizer scale factor that can be used by the decompressor to dequantize the coded DCT coefficients. In case of data corruption, the information in the slice header allows for a smoother recovery by the source decoder. The reason for the slice structure is error recovery purpose.

A slice is divided into macroblocks. A macroblock is the basic coding unit with MPEG2. For the 4:2:0 format it consists of a 16×16 pixel array of luminance samples

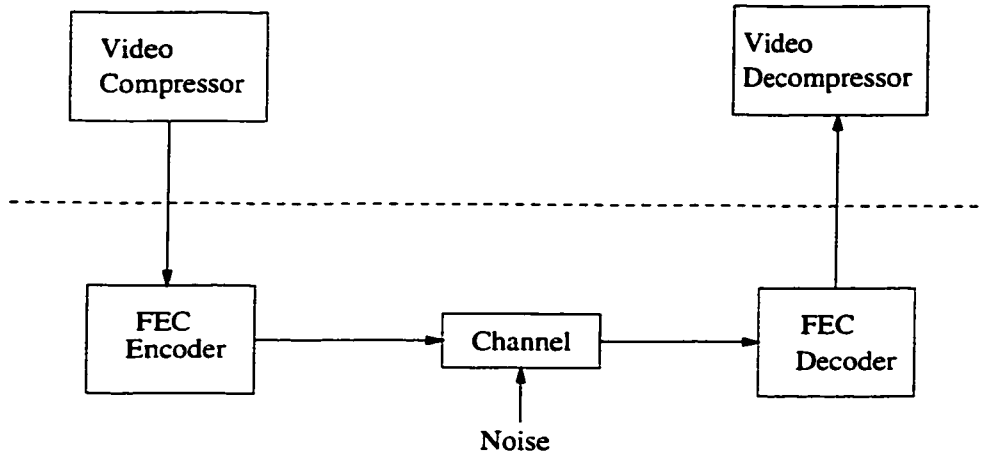

Figure 2.4: Protecting MPEG2 using a FEC code

together with one 8×8 sample array for each of two chrominance components. Motion estimation and compensation are done on a macroblock by macroblock basis.

The block is the smallest coding unit in the MPEG2 algorithm, and is the basic element for the DCT. It is made up of 8×8 pixels and can be one of the three types: luminance (Y), blue chrominance (Cb), and red chrominance (Cr).

2.2 Error control coding

The simplified diagram of a video transmission system using error correction is given in Figure 2.4.

After compression, the bitstream is encoded for error correction, and then transmitted through a channel. At the receiver, data passes through the FEC decoder. The decoded sequence is then sent to the video decompressor.

The FEC decoder uses the maximum likelihood (ML) decoding algorithm based on a defined likelihood metric. The FEC decoder select the candidate that gives the largest metric [14]. For a binary symmetric channel (BSC), the largest likelihood metric is equivalent to the shortest Hamming distance.

For example, for an (n, k) binary code, there are 2^k distinct messages. Corresponding to the 2^k possible message blocks, there are 2^k n-bit codewords. The

weight of a word is defined as the number of "1"s in the word. Let r be the received n-bit word. Let c be any valid codeword. The Hamming distance between r and c is defined as:

$$d_H(r,c) \triangleq w(r \oplus c) \tag{2.1}$$

where $w(\cdot)$ is the to get the weight of a word, and $\oplus$ is the modulo-2 adder.

Let $\mathcal{C} : \{c_0, c_1, ...c_{m-1}\}$ be the group of $m = 2^k$ valid codewords. The FEC decoder computes the Hamming distances between r and c_j for $j = 0, 1, ..., (m-1)$ and selects the valid codeword c_i that satisfies

$$w(r \oplus c_i) \leq w(r \oplus c_j) \tag{2.2}$$

where $j = 1, 2, ...m-1$, $j \neq i$, the Hamming distance between c_i and r, i.e., $d_H(c_i, r)$, is the shortest d^s for r.

In the case of a tie, (i.e., more than one $c_i's$ with the same d^s), only one of them is selected randomly.

The minimum distance of an (n, k) binary code is defined as the shortest Hamming distance between any pair of valid codewords, i.e.,

$$d_{min} = min\{d_H(c_i, c_j) : c_i, c_j \in \mathcal{C},\ c_i \neq c_j\} \tag{2.3}$$

It can be shown that the guaranteed number of correctable bit errors of this (n, k) code is [14]:

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor \tag{2.4}$$

where $\lfloor \frac{d_{min}-1}{2} \rfloor$ denotes the integer part of $(d_{min} - 1)/2$.

It follows that, for a BSC with a channel bit error rate of p, the upper bound on the post-decoding error rate of an n-bit word is [11]:

$$P_M \leq 1 - \sum_{i=0}^{t} \binom{n}{i} p^i (1-p)^{n-i} \tag{2.5}$$

For $p \ll 1$, it can have a rough approximation

$$P_M \approx \binom{n}{t+1} p^{t+1}$$

which indicates a significant improvement as t increases.

2.3 Error concealment techniques

EC techniques make use of residual redundancy in the frequency, temporal and spatial domain after compression to conceal transmission errors. Typically the concealment problem is broken into two parts: first, detection of the area (usually a block) that is corrupted; second, concealment of that damaged area. Often error concealment schemes [4] concentrate on the second of these problems and little mention is made of the first. Generally no effort is made to ensure that the resulting image/video is consistent with the bitstream received.

Error concealment schemes can be divided into four categories. Simple Concealment, Frequency Concealment, Spatial Concealment and Temporal Concealment.

Simple concealment algorithms include simple replacement spatially or temporally adjacent blocks from the previous transmitted blocks. These schemes work best when the BER is not high and when there is not much scene change and motion from frame to frame [4].

Frequency concealment makes use of the fact that the 8*8 block DCT does not result in fully uncorrelated coefficients, especially low frequency coefficients of adjacent blocks. A typical frequency concealment algorithm [4] does linear or polynomial interpolation of adjacent DCT coefficients.

For an example of frequency concealment see Park, Kim and Lee [15]. They use the information in the compressed domain (syntax of the bitstream) to detect an error occurance. Then a discontinuity measure at the boundary of each macroblock is calculated to determine the exact location of the corrupted block in the slice. When a corrupt block is detected, all information about that block is discarded and a concealment scheme is used that adjusts the lost DCT coefficients to minimize a boundary smoothness objective function. The resulting video is not guaranteed to be close to the transmitted video in terms of number of bits that are different between concealed bitstream and transmitted bitstream. PSNR values drop quickly for moderate BER. This scheme works in the compressed domain. Video is compressed by MPEG2 and bit errors are considered.

Chu and Leou [16] do a similar work to Park[15]. The syntax of the bitstream is used to detect transmission errors in the compressed domain. It will indicate that an error has occurred. The information in the uncompressed domain, namely spatial discontinuity, is used to determine the exact location of the corrupted block in the slice. Then the corrupted blocks are concealed using a combination of spatial and temporal concealment scheme, which switches according to a fitness function also based on spatial and temporal continuity. Video is compressed with H.261 and burst errors are considered.

Spatial concealment [17] is to interpolate directly into the spatial domain using the neighboring blocks within the same picture. Because interpolation is implemented in the spatial domain, finer details of the image can be reconstructed and consistency of the edges can be maintained to a certain degree.

Hung and Chang [18] present a spatial concealment scheme that uses the Vector Quantization (VQ) idea. A VQ codebook is generated using fairly small blocks. This is not used for transmission but only for concealment. Once a lost block is identified blocks are chosen that contain some lost pixels and some uncorrupted pixels. The closest codeword in the VQ code book (i.e. closest to the uncorrupted pixels) is selected and the lost pixels are filled in accordingly. This scheme works in the

uncompressed domain.

Temporal concealment is possible for the compression algorithms where the video sequence is further compressed in the temporal domain. So a lost motion vector can be reconstructed and then the referenced block can be put into the same position. It works well with low motion picture but there is a trade-off between the performance and real-time implementation [4].

The simple, frequency, spatial and temporal concealment techniques conceal the errors by postprocessing at the source decoder. An alternative approach to combat transmission errors from the source side is by using multiple description coding (MDC). With MDC, several coded bitstreams (referred to as "descriptions") of the same source signals are generated. The MDC encoder and decoder are designed such that the quality of the reconstructed signal is acceptable with any one description and that incremental improvement is achievable with more descriptions.

Wang, Orchard and Reibman [19] use MDC for noisy channels by pairing transform coefficients. The proposed MDC algorithm codes a pair of variables simultaneously. Instead of coding the two variables independently, they are transformed into two other variables, which are quantized and transmitted separately. In the receiver, if both transformed variables are available, they are inversely transformed to recover the original two variables. If only one transformed variable is received, it is used to estimate the original two variables. The proposed scheme can guarantee an acceptable image quality in the presence of any one description with only a small overhead over a standard JPEG coder. Their emphasis is on long burst errors.

2.4 Survey on joint FEC/EC techniques

Forward Error Correction techniques correct transmission errors based on a designed channel code structure. Error Concealment compensates the errors using source information. Generally speaking, these schemes will attempt to make output video as consistent as possible with the bits received, while factoring in information from the source video. Recently, several jointly source/channel decoding schemes using

the residual redundancy in the video together with classic error control coding have been presented. Wang and Wenger[20] gives a view of the error resilience schemes for real-time video transport over unreliable networks. They categorized these techniques into three groups: those introduced at the source and channel encoder to make the bitstream more resilient to potential errors; those invoked at the decoder upon detection of errors to conceal the effect of errors; and those which require interactions between the source encoder, decoder and channels.

The joint FEC/EC scheme can be classified by their compression scheme. Some schemes that work on VQ compressed video are briefly discussed in the following paragraphs.

In [21], a joint source/channel decoding for vector quantized compressed video using Turbo codes is presented. It aims at bit errors. When the channel decoder cannot correct all bit errors in the transmitted codevector indices, the corresponding blocks in the reconstructed image are detected by exploiting spatial contiguity in an image and the reliability information of the channel decoder. For those erroneous blocks, the reliability information of each bit is examined. The bit with lowest reliability is then changed. The source information is fed back to the Turbo decoder to improve the error correction capability.

In [22], error detection and correction of bit errors for pyramid coding are studied. One of the schemes presented here inserts one bit of parity check in the "data block" of pyramid entries to detect the errors. When errors are detected, they are corrected by bit toggling inside the corrupted portions of the bitstream. All corrections are in the compressed domain and the syntax is used to find a "feasible" solution.

Burlina and Alajaji [23] propose a joint source/channel decoding scheme which exploits statistical redundancy in the image data. They model the image data and use that along with a channel model with a maximum a posterior (MAP) detector. For the compressed video, the residual redundancy is exploited via unequal error protection and MAP detection. This scheme is for still images.

There are joint FEC/EC techniques proposed for MPEG, JPEG, H.261[24], and H.263[25] methods. These schemes can be divided into two classes: with and without feedback channel.

With feedback channel, the source encoder and decoder cooperate in the process of error concealment. When the receiver detects an error, it will notify the transmitter, over the feedback channel, which parts of the bitstream are received error-free and/or which parts could not be decompressed and have to be concealed.

Girod and Farber [26] give a review of the feedback error control techniques. H.263 is the target codec. They examine strategies that use feedback from the receiver to the transmitter along with error tracking, error confinement and reference picture selection to alleviate interframe error propagation. A more detailed error tracking scheme is given in [27]. Their scheme assumes that the video compressor is partially controlled by the channel. While this may be feasible in a point to point, interactive application, it presents difficulties in a broadcast or video server scenario. For the video server application they offer the expedient of the server storing a P frame and I frame version of each frame to be switched to as needed by the channel. This in turn introduces a mismatch in the "residue" that is DCT compressed. The mismatch error is less objectionable than channel errors. Broadcast applications or point to multipoint appear not to have been considered.

Other techniques use the forward error concealment and there is no feedback by the channel. There is little interaction between the receiver and transmitter.

Kim [28] uses syntax information to detect errors. Then extra data, namely parity-check DC coefficients, are calculated and inserted into the bitstream to locate the errors. Then a temporal concealment technique is done to retrieve the motion vector of the erroneous block. An error measure, which measures the difference between the parity-check DC coeffients in corrupted and original sequence, is calculated to evaluate which motion vector is the best. This measure is similar to the one used later in this thesis, which is stated in equation 3.1-3.4 in that it matches the boundary value of pixels. However it measures the mean absolute difference, where

this work measure the difference in DC value. Video is compressed with H.263 and burst errors are considered.

In [29], the locations of the bit errors within a bitstream are provided by a simple detection code – parity check. Then temporal/spatial concealment is used to conceal the errors. That is part of the previous work. In the other part of the previous work, the same error detection scheme is used and then the syntax of the bitstream and spatial discontinuity is used to correct/conceal these errors. These schemes will be outlined in Section 3.2.

Chapter 3

A Combined FEC and Error Concealment Technique

This chapter presents a proposed method for recovering transmission errors in MPEG2 compressed bitstreams based on information given by both FEC decoder and video decompressor. Unlike the general FEC scheme that gives only one solution, the proposed FEC decoder can provide a number of candidate solutions along with their reliability. These solutions are further screened by examining the corresponding syntax and discontinuity in an error concealment scheme.

The two kinds of source information used in this thesis, namely syntax and discontinuity are introduced in 3.1. Section 3.2 presents some previous work which is the basis of the work in this thesis. In Section 3.3, the combined FEC/EC scheme is presented and the description of the system is given along with the details concerning the receiver. At last, the evaluation of the complexity is introduced in Section 3.4.

3.1 Source information applicable to error concealment

As reviewed in Section 2.3, different kinds of source information such as spatial, temporal, etc. can be used to conceal errors. In this thesis, two kinds of source information are used to conceal errors: syntax/semantic correctness and the discontinuity of the uncompressed video. Section 3.1.1 outlines the syntax/semantic information while section 3.1.2 outlines the discontinuity information which is quantified by a discontinuity measure.

3.1.1 Syntax

MPEG2 standards are generic in the sense that the values of important parameters are specified in the syntax. Assuming that the valid MPEG2 bitstreams conform to the syntax/semantics in the standard so that the violation of the syntax/semantic is regarded as an obvious error.

The bitstream is claimed to have a syntax/semantic violation, if the incoming bits do not match any MPEG2 codeword in the VLC table, or the decompressed parameters violate the specification imposed by the coding standard [30]. Some example errors are given as follows:

1. An VLC decoding error

The MPEG2 bitstream is a concatenation of VLC and Fixed Length codes (FLC). When the MPEG2 bitstream is decompressed, both FLC and VLC should be parsed and decoded. As stated in Section 1.1, VLC is vulnerable to errors. When an VLC codeword is decoded, if it cannot be found in the corresponding codebook, there is an VLC decoding error, and the bitstream is claimed to have an syntax/semantic error.

2. Motion displacement range violates the semantics

It is specified in MPEG2 standard that for both luminance and chrominance components [1], all motion displacement pixel values should lie strictly within the picture, the size of which is defined in the sequence header. Therefore, the bitstream is claimed to have a syntax/semantic error if any of the motion displacement pixels is out of the picture boundary.

3. Slice_ Start_ Code is in the middle of blocks

The MPEG2 syntax reserves several start codes for synchronization purpose. As stated in Section 2.1.3, a slice is divided into macroblocks, which are composed of blocks. The Slice_Start_Code should be at the beginning of the block. If a Slice_Start_Code is detected in the middle of the block, then there is a syntax/semantic violation in this slice.

4. The number of DCT coefficients in a block exceeds 64

Since the DCT size is 8×8, the maximum number of the DCT coefficients in each block should not be greater than 64. When the end_of_block has not been encountered but the coefficient count has already reached 64, a syntax/semantic error is detected.

Because many syntax errors are not recognizable as such until the end of a slice, the decision is made on a slice-by-slice basis. All slice candidates are decompressed and checked for syntax errors and the ones that result in an MPEG syntax/semantic error are rejected. These slice candidates will result in the most egregious degradation of the video.

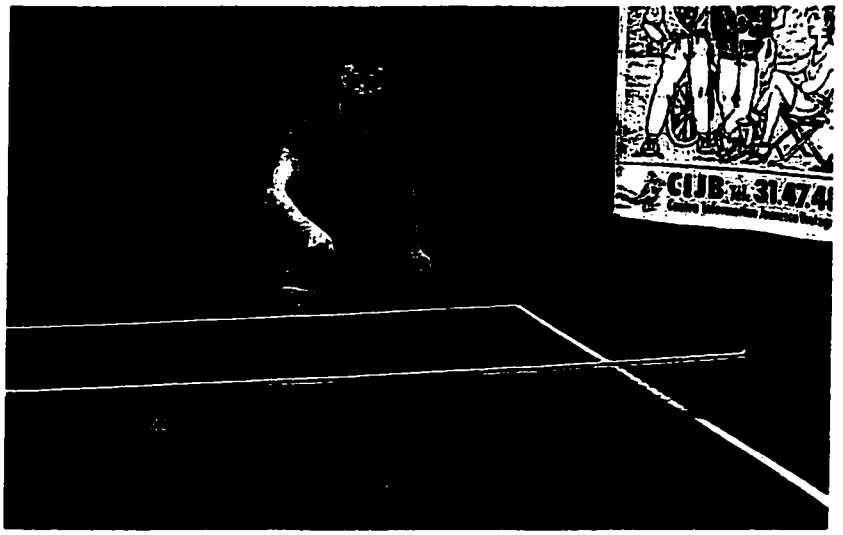

Figure 3.1: Frame 15 from "Table Tennis", with a edge between the right leg and the background

If no such errors exist then the probability of correctness of the slice candidate can be judged by the discontinuity of the uncompressed slice. A measure of this discontinuity is presented in the next section.

3.1.2 Discontinuity measure

The discontinuity of the pixel values occurs when there is an edge in the picture. But abrupt discontinuity is very unlikely in a natural picture. There would be some transition pixel values at the edge. For example, in Figure 3.1 there is a edge between the right leg of the person and the background.

But when the pixel values are examined in detail as shown in Figure 3.2, it is found that, instead of an abrupt discontinuity, there are some transition points.

If the differentiation of the pixel values is calculated, it can be seen that instead of an impulse (i.e., very narrow pulse), which is the differentiation of an abrupt discontinuity, there is also a transition area. That is shown in Figure 3.3.

Since the abrupt discontinuity would not occur in natural pictures, it can be detected as an error to be concealed. For this, the following discontinuity measure is introduced.

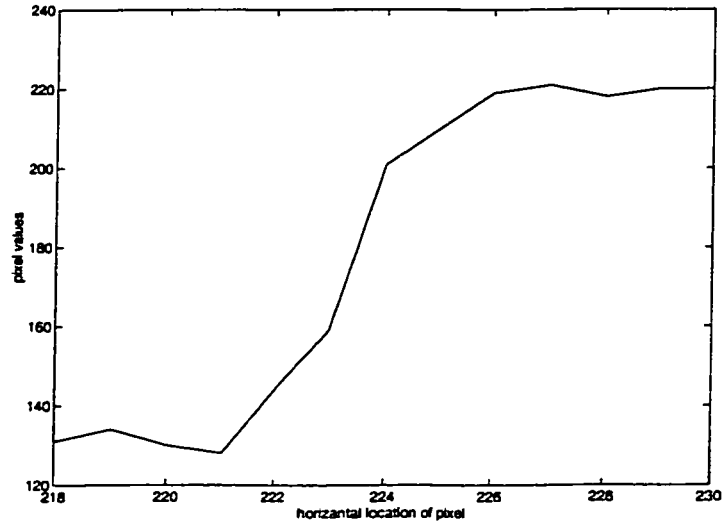

Figure 3.2: The pixel values at the edge

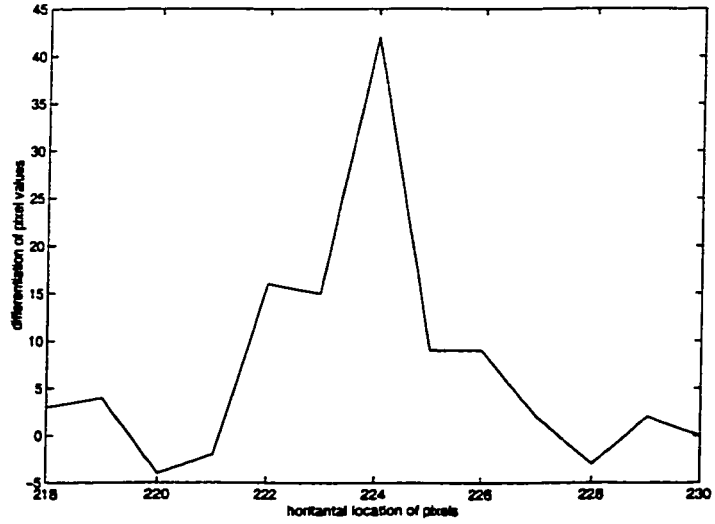

Figure 3.3: Differentiation of the pixel values

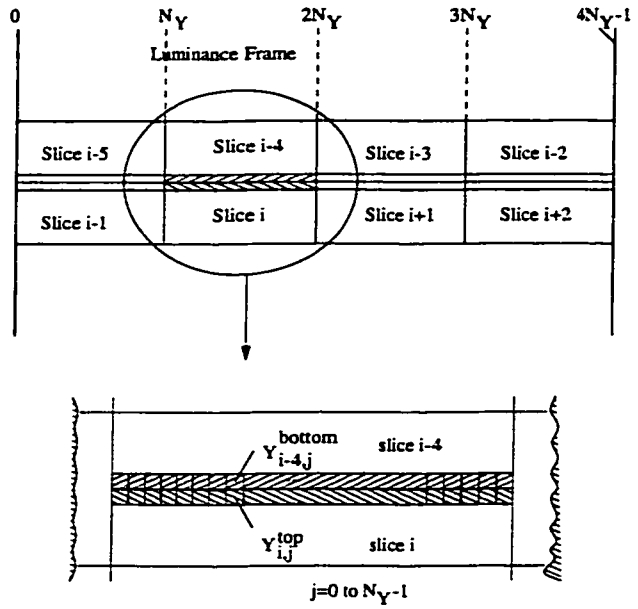

Figure 3.4: Location of pixels to be used in discontinuity measure

Among the slice candidates without syntax/semantic errors the worst degradations result from either errors in the DC values or left/right shift in the macroblocks [31]. Both these egregious errors may propagate along the slice, and then infect the subsequent frames via motion compensation. One feature of these two error types are that they cause abrupt discontinuities at slice boundaries. The discontinuity measure is used to weed out these errors.

The order of decompressing is from top to down and from left to right, and since the slice is one macroblock high and several macroblocks across, the longest slice boundary available when decompressing a slice, is the one between the current slice and the one above. Note that for the sequence used in the simulation, there are 4 slices per line. To test for the two types of degradations outlined above, let S_i denote the slice i being decoded, define $Y^{top}_{i,j}$ as the luminance values of the top row of pixels of S_i , and $Y^{bottom}_{i-4,j}$ as the luminance values of the bottom row of pixels in the above slice. Here $j = 1, 2, ...N_Y$, and N_Y is the width of a slice in luminance pixels. See Figure 3.4 for the illustration of these values in a luminance picture. The two shaded rows of pixels are the ones of interest.

The spatial discontinuity measure is defined for the luminance component as[31]:

$$M_Y \triangleq \frac{1}{N_Y} \sum_{j=0}^{N_Y-1} (Y_{i-4,j}^{bottom} - Y_{i,j}^{top}) \tag{3.1}$$

The difference between each pixel in the top line of current slice and the bottom line of slice above is averaged. In Section 2.3 and 2.4, some error concealment schemes also make use of the discontinuity measure. But they are different to the discontinuity measure used in this thesis in that they add the absolute value of the difference instead of average. In that case, the noise will pile up and there is always a bias for the discontinuity measure, no matter there is error or not. But if the difference is averaged, the noise can be cancelled. Only when there are abrupt discontinuities at the slice boundaries, there would be a bias for the discontinuity measure. An adverse effect of this discontinuity measure is that the noise cancellation may also cancel the checker board blocks. Thus, the checkerboard blocks cannot be corrected. This will be suggested as future work in Chapter 5.

Similar measures can be defined for the chrominance components M_U and M_V [31].

$$M_U \triangleq \frac{1}{N_U} \sum_{j=0}^{N_U-1} (U_{i-4,j}^{botttom} - U_{i,j}^{top}) \tag{3.2}$$

$$M_V \triangleq \frac{1}{N_V} \sum_{j=0}^{N_V-1} (V_{i-4,j}^{bottom} - V_{i,j}^{top}) \tag{3.3}$$

where $U_{i,j}^{top}$, $V_{i,j}^{top}$ are the chrominance values of the top row of pixels in S_i, $U_{i-4,j}^{bottom}$, $V_{i-4,j}^{bottom}$ are the chrominance values of the bottom row of pixel in the slice above S_i, and N_U, N_V are the width of the slice in chrominance pixels.

What is of interest is the value of the bias instead of the direction, so the overall

discontinuity measure is defined as:

$$M \triangleq (\lambda \times M_Y^2 + M_U^2 + M_V^2)^{1/2} \tag{3.4}$$

where λ is a weight coefficient. Since the chrominance component is subsampled by a factor of two horizontally, i.e., $N_Y = 2N_U = 2N_V$, when the discontinuity measure is divided by $\frac{1}{N_Y}$ or $\frac{1}{N_U}$ and then squared, the coefficient divided by M_Y^2 is 4 times over that divided by M_U^2 or M_V^2. Thus $\lambda = 4$ is an appropriate value and is used in the remainder of this thesis.

3.2 Error concealment using syntax information

Classical temporal and spatial concealments (see Section 2.3) work in the uncompressed domain and try to have a "smooth" image [31]. In the previous work, the errors are corrected in the compressed domain, in which the primary criteria is a bitstream without syntax/semantic errors. As a refinement a discontinuity measure calculated from the uncompressed domain is added to select among slice candidates without syntax violation.

3.2.1 Syntax Based Error Concealment (SBEC)

In [32], SBEC is presented and its block diagram is shown in Figure 3.5. (The dashed line of "uncompressed video" is present only in the scheme described in section 3.2.2, not in SBEC). To generate the slice candidates, for every block of 12 compressed data bits, a parity check bit is appended. The combined 13 bit block of data and parity is an Error Concealment (EC) block. At the error detection decoder, the received EC block is checked to determine whether or not the parity is satisfied. If the parity is not satisfied, the EC block will be marked as in error. All odd number of errors can be detected in this way. For each flagged erroneous EC block, the slice which contains it will be partially decompressed up to 13 times. In each partial decompression, a different bit in the EC block is toggled until no syntax

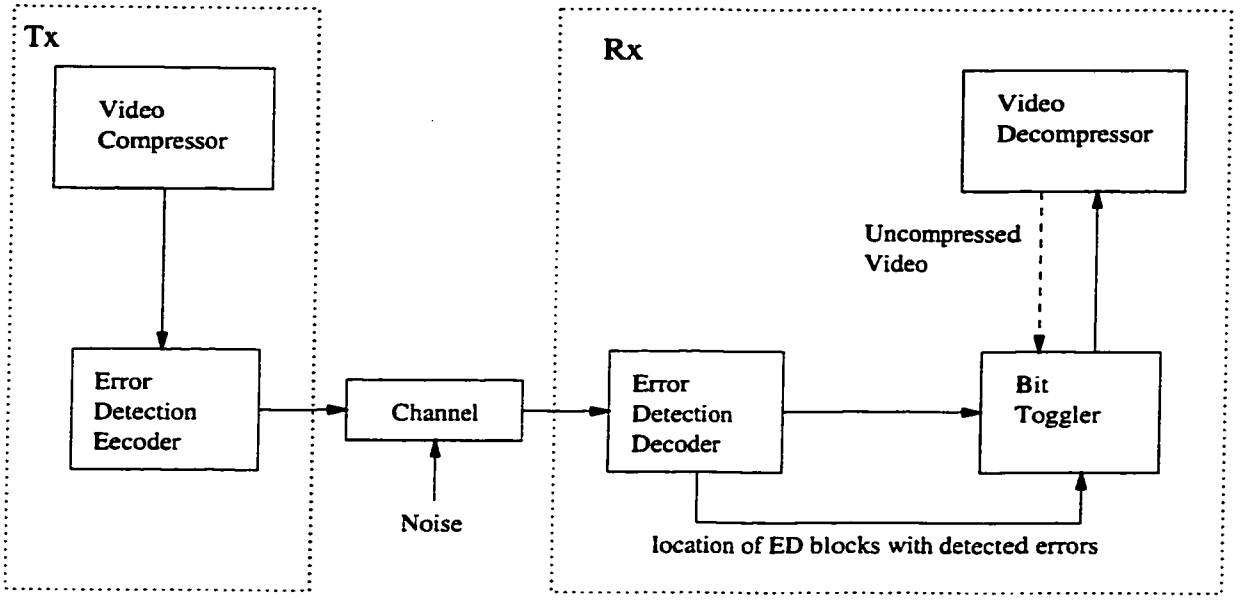

Figure 3.5: The SBEC scheme(no dashed line), and the SDBEC scheme (includes dashed line)

violation is found. Only partial decompressions are needed since the video itself is not constructed.

3.2.2 Syntax and Discontinuity Based Error Concealment (SDBEC)

SDBEC [31] is based on SBEC and is a refinement of SBEC. It works as SBEC but does a full decompression of the bitstream instead. To generate the slice candidates, the same parity check scheme as that of SBEC is needed. For those slice candidates that do not have a syntax violation, the discontinuity measure is calculated to measure their goodness. All togglings are examined and the one with no syntax violation and the best measure is taken as correct. SDBEC is shown in Figure 3.5 and the dashed line is present.

The scheme proposed in this thesis builds on the work presented above. It is presented in the next section.

3.3 Proposed scheme

SDBEC [31] mentioned in section 3.2.2 makes use of the syntax information and

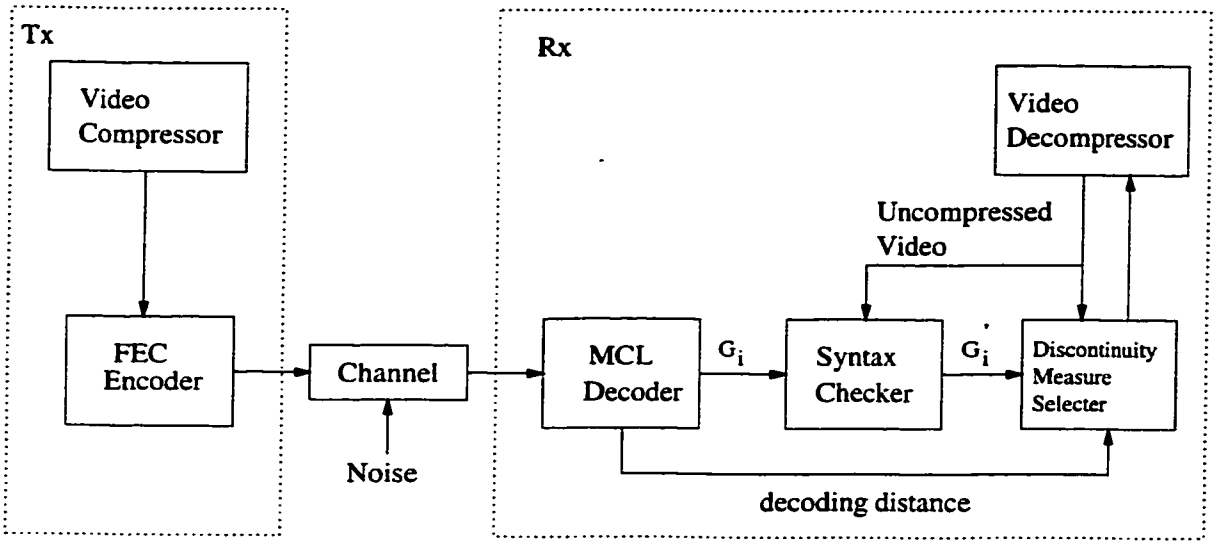

Figure 3.6: Structure of proposed combined FEC/EC scheme

discontinuity measure to correct errors. The proposed scheme does not introduce a new EC algorithm, but uses existing methods [31, 32]. The close coupling of FEC and EC is the main contribution of the proposed scheme. The expansion of the number of slice candidates and the use of FEC reliability information is the main difference between the proposed scheme and SDBEC.

The block diagram of the proposed scheme is illustrated in Figure 3.6. After compression, the bitstream is encoded for error correction, and then transmitted through a channel. At the receiver, data pass through the FEC decoder. Unlike the classic FEC approach, the FEC decoder gives more than one slice candidates along with their reliability information to the syntax checker where syntax/semantic errors are checked for. One decompression for each slice candidate takes place. The slice candidates with syntax/semantic errors are rejected and only those without syntax/semantic violations are sent to the discontinuity measure selector. The discontinuity measure of each slice candidate is calculated and then combined with the Hamming distance to get a modified discontinuity measure. The slice candidate with the best modified discontinuity measure is taken as the correct one.

In the following sub-sections, each part of the receiver is described in detail.

3.3.1 Multiple-Candidate Likelihood (MCL) decoder

Prior to the descriptions of the proposed MCL decoder, it is worth reviewing the principle of a conventional Maximum Likelihood (ML) decoder.

Consider a designed code C containing m valid codewords, c_j for $j = 0, 1, ..., (m-1)$. For a received word r, the likelihood metric between r and c_j, $l(r, c_j)$ is calculated. An ML decoder will give an output valid codeword c_i that satisfies

$$l(r, c_i) \geq l(r, c_j) \tag{3.5}$$

for $j = 0, 1, ..., (m-1)\, and\, j \neq i$.

For an additive white Gaussion noise (AWGN) channel, the likelihood metric is:

$$l(r, c_j) = -d_E(r, c_j) \tag{3.6}$$

where $d_E(r, c_j)$ is the Eudidean distance between r and c_j.

For a binary symmetric channel (BSC), the likelihood metric is:

$$l(r, c_j) = -d_H(r, c_j) \tag{3.7}$$

where $d_H(r, c_j)$ is the Hamming distance between r and c_j.

From the above relations, it can be seen that the ML decoder will select an output c_i such that

$$d(r, c_i) \leq d(r, c_j) \tag{3.8}$$

where

$$d(r, c_j) = \begin{cases} d_E(r, c_j) & in\, an\, AWGN\, channel \\ d_H(r, c_j) & in\, a\, BSC \end{cases}$$

for $j = 0, 1, 2, ..., (m-1)\, and\, j \neq i$.

It can also be seen that, for the case of more than one valid codewords that satisfy the ML criterion, the conventional ML decoder will randomly select one solution.

In the proposed Multiple-Candidate Likelihood (MCL) decoder, rather than selecting *one* solution, it considers a set:

$$B_0 = \{c_i \mid d(r, c_i) = d^s \leq d(r, c_j),\ c_j \in \mathcal{C}\} \tag{3.9}$$

i.e., B_0 is the set of candidate codewords $c_i's$ with the shortest distance d^s.

In a general case, an extension to a set of valid codewords is considered:

$$B_k = \{c_k \mid d(r, c_k) = d^s + a,\ c_k \in \mathcal{C}\} \tag{3.10}$$

for a given offset "a". B_a is called the set of "shortest-distance-plus-a" candidates.

The FEC decoding is processed on the unit of a received word with length equal to that of the valid codeword. On the other hand, the decompression (including both syntax/semantic check and discontinuity measure calculation) operates on the slice-by-slice basis. Depending on the selected code length, a received slice, S, is considered as a concatenation of the message portions x_i of L received words r_i,

$$S = x_1|x_2|.....|x_L \tag{3.11}$$

where | denotes concatenation.

For each message portion x_i of received word r_i, there may have more than one candidates produced by the MCL decoder. Let c_i denote a codeword candidate and y_i denote its message portion. The distance between r_i and c_i is:

$$d_i \triangleq d(r_i, c_i) \tag{3.12}$$

As a result, there are more than one slice candidates. Let $\mathcal{S}$ denote a slice candidate:

$$\mathcal{S} = y_1|y_2|.....|y_L \tag{3.13}$$

The distance between the received slice S and the slice candidate $\mathcal{S}$ is defined as:

$$D(S,\mathcal{S}) \triangleq \sum_{i=1}^{L} d_i \tag{3.14}$$

It follows that if all $c_i's$, $i = 1, 2, ...L$ are selected from sets $B_0's$ with the shortest distance d_i^s, then the slice candidates $\mathcal{S}$ belongs to G_0, the group of slice candidates with the shortest distance

$$D^s \triangleq \sum_{i=1}^{L} d_i^s \tag{3.15}$$

In general, G_a, the group of all slice candidates with distance $(D^s + a)$ for the received slice S, is defined as:

$$G_a \triangleq \{\mathcal{S}|d(S,\mathcal{S}) = D^s + a\} \tag{3.16}$$

For convenience, define "search space" as the range of slice candidates to be checked by the syntax checker. It may contain G_0 or G_0 and G_a.

For example, if only one codeword candidate c_i belongs to the set of shortest-distance-plus-a candidates, B_a, and the $(L-1)$ remaining codeword candidates belong to the sets B_0 with the shortest distances, then the corresponding slice candidate $\mathcal{S}$ belongs to G_a.

3.3.2 Benefits of MCL decoding

If only candidate with the shortest distance D^s is selected, then the ML criterion is achieved. From the channel coding point of view, all slice candidates with the

shortest distance are equally good. Therefore, without further consideration, an arbitrary selection of one solution from these candidates yields the same post-decoding error correction performance. This is exactly what is done by the traditional ML decoder[14].

However, if further consideration for error resilience using other measures such as in this application of error concealment based on the residual redundancy of the digital video signal, then the traditional ML decoder prevents the chance to examine other equally good candidates. In this sense, the proposed MCL decoding strategy with all shortest-distance slice candidates will definitely enhance the performance because all equally good slice candidates (from the channel coding viewpoint) are further considered using additional metrics based on the digital video signal properties.

At a low BER the number of slice candidates with the shortest distance is mostly 1. In this case, the MCL decoder with shortest distance is equivalent to the traditional ML decoder with one solution. The performance of the proposed scheme is expected to be the same as that of the traditional FEC scheme. At a high BER, the number of slice candidates with shortest distance increases and the proposed scheme is expected to outperform the traditional FEC scheme.

Figure 3.7 shows the simulated number of slice candidates in G_0 for each slice of the sequence "Table Tennis" at BER of 1.2×10^{-2} using a (16,8) quasi-cyclic code.

Though for most slices, there are only one slice candidate in G_0, there are about 5% of the total 21600 slices that have more than one slice candidates in G_0.

By allowing slice candidates with a shortest distance plus a where $a > 0$ to be considered in further error concealment, more trust is essentially put on the further error concealment. As "a" increases, the number of slice candidates is increased. However, by allowing a larger value of "a", the "trust" on the channel decoder is reduced intentionally. This may degrade the performance of the proposed scheme.

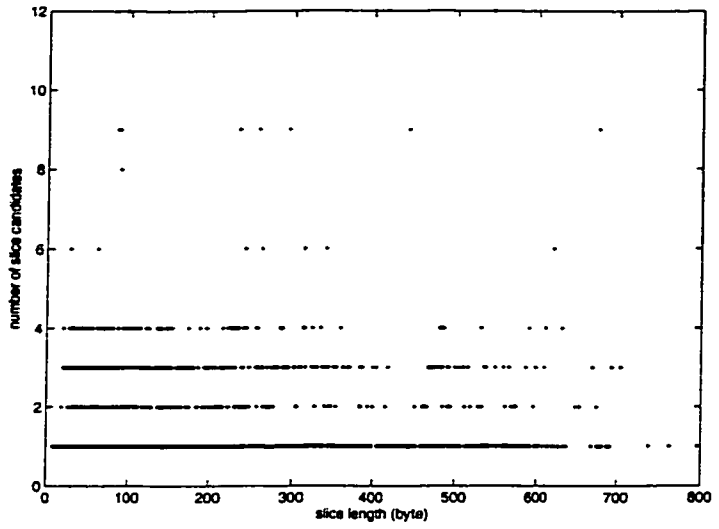

Figure 3.7: Simulated number of slice candidates in G_0 for each slice of the sequence "Table Tennis", at BER of 1.2×10^{-2}

3.3.3 Size of slice candidate groups

Consider all the slice candidates in G_0. Let N_i be the number of possible slice candidates in G_i. For these slice candidates each of their constituent message portions gets a message portion candidate from B_0. Given a received slice $S = x_1|x_2|.....|x_L$, suppose there are $f_0(x_i)$ message portion candidates in B_0 for x_i, then N_0 is

$$N_0 = \prod_{i=1}^{L} f_0(x_i) \tag{3.17}$$

If G_1 is considered, to generate a slice candidate, any one message gets one message portion candidate from B_1. Suppose there are $f_1(x_i)$ candidates in B_1 for x_i, then N_1 is:

$$N_1 = \sum_{j=1}^{L}[f_1(x_j) \prod_{i=1, i\neq j}^{L} f_0(x_i)] \tag{3.18}$$

When G_2 is considered, to generate a slice candidate, there are two possibilities: any two message portions get message portion candidates from B_1 or any one message portion gets the message portion candidate from B_2. Suppose there are $f_2(x_i)$

message portion candidates in B_2 for x_i, then N_2 is:

$$N_2 = \sum_{j=1}^{L}[f_2(x_j) \prod_{i=1, i\neq j}^{L} f_0(x_i)] + \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} [f_1(x_i) f_1(x_j) \prod_{k=1, k\neq i,j}^{L} f_0(x_k)] \quad (3.19)$$

For $(a+1)$ groups, G_0, G_1, ..., G_a, given by the MCL decoder, there are

$$N = \sum_{i=0}^{a} N_i \quad (3.20)$$

slice candidates to the syntax checker.

3.3.4 Syntax checker

As discussed in Section 3.1.1, syntax/semantic based redundant information can be used to conceal bit error occurance in the compressed bitstream. The objective of the syntax checker is to check the slice candidates given by the channel decoder for syntax/semantic errors before they are sent to be evaluated by the discontinuity measure.

For the N slice candidates given by the MCL decoder, the syntax checker fully decompresses each slice candidate and checks for syntax/semantic errors. Because many errors are not recognizable as such until the end of the slice, the decisions are made on a slice by slice basis. The slice candidates that result in an MPEG syntax/semantic error are rejected and only those slice candidates which have no syntax/semantic violation are sent to the discontinuity measure selector to calculate their discontinuity measures. The "no syntax/semantic violation" condition occurs when the decompression arrives at the end of the current slice without finding a syntax/semantic error.

3.3.5 Discontinuity measure selector

After syntax checker, the outright syntax violation has been removed by the syntax checker. The slice candidates passed from the syntax checker are then sent to the discontinuity measure selector.

The discontinuity measure as defined in Section 3.1.2 indicates the change in pixel values as compared to neighbor slices. Since a large change unlikely occurs, a slice candidate with the smallest value of discontinuity measure is considered as the best choice.

When slice candidates in the same group (G_0, G_1 or G_2) are considered, discontinuity measure is used to evaluate their "picture" goodness. But if slice candidates in different groups are evaluated, their discontinuity measure cannot be treated equally because these slice candidates have different performance in view of channel coding. This factor must also be taken into consideration as follows.

The distance of a slice candidate from the received slice is an evaluation of how likely that slice candidate is, in view of channel coding, e.g., a slice candidate $\mathcal{S}_0 \in G_0$ has a better performance in terms of channel coding than a slice candidate $\mathcal{S}_1 \in G_1$. When they are further considered in terms of discontinuity measure, this should be included. For this, a modified discontinuity measure for a slice candidate $\mathcal{S}_i$ belonging to group G_a is introduced:

$$\mathcal{M}_i \triangleq M_i + \alpha_a \tag{3.21}$$

where M_i is the discontinuity measure of slice $\mathcal{S}_i$, α_a is a constant offset. Since a slice candidate $\mathcal{S}_i$ belonging to group G_a has a better performance than a slice candidate $\mathcal{S}_j$ of group G_b where $a < b$, it is logical to select $\alpha_a < \alpha_b$. For $a = 0$ (i.e., slice with the shortest distance), $\alpha_0 = 0$ is selected.

The objective of α_a is therefore to balance between candidates in G_0 and those in G_a so that the PSNR is maximized. The search for α_a will be presented in Section 4.2.

3.3.6 Video Decompressor

The decompressor module have the ability to perform in the presence of errors. Also it can pass the uncompressed video to the syntax checker and discontinuity measure

selector. The MPEG2 decompressor V1.2a is used in this work. The diagram of a video compressor is shown in Figure 2.2.

3.4 Complexity of the proposed scheme

The complexity of the scheme depends on two factors:

- The complexity of the FEC decoder
- The complexity of the syntax checker and discontinuity measure selector

Complexity of the FEC decoder depends on the selected code. Complexity of the syntax checker and discontinuity measure selector is mainly related to two processes:

- Slice-by-slice decompression of the MPEG2 sequence
- Discontinuity measure calculation for each slice

Discontinuity measure calculation is relatively simple. The major contributor to the complexity of the proposed scheme is in the repeated decompression of a number of slice candidates. Therefore, the order of complexity can be represented by:

(i) Number of slice candidates to be decompressed

(ii) Number of bits extracted from the bitstream

3.4.1 Number of slice candidates to be decompressed

The number of slice candidates to be decompressed can be estimated as follows. Consider a (16,8) quasi-cyclic code with $d_{min} = 5$ defined in Appendix A.

The $2^{16} = 65536$ possible received words are split into classes according to how far each is away from the closest valid codeword. Specifically Class i contains all received words that are i away from the closest valid codeword. Thus Class 0 contains the valid codewords since they are 0 away from themselves, and there are $2^8 = 256$ members in Class 0. For the minimum distance of 5, the numbers of members in Classes 1, 2 can also be calculated. For each valid codeword, there

Type τ	Class i	n_τ	Distance Profile $\gamma(\tau, j)$ for j=0,1,...16																	Probability P_τ
			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
0	0	256	1	0	0	0	0	24	44	40	45	40	28	24	10	0	0	0	0	0.82434900147619
1a	1	2048	0	1	0	0	8	16	32	52	48	37	32	20	8	2	0	0	0	0.08009986458629
1b	1	2048	0	1	0	0	7	17	36	48	42	43	36	16	7	3	0	0	0	0.08009972289769
2a	2	2048	0	0	1	3	4	15	38	46	44	46	33	15	8	3	0	0	0	0.00100890786913
2b	2	5120	0	0	1	2	5	18	35	44	46	44	35	18	5	2	1	0	0	0.00249310113476
2c	2	4096	0	0	1	2	6	17	31	48	52	38	31	22	6	1	1	0	0	0.00199476428500
2d	2	5120	0	0	1	2	6	18	30	44	56	44	25	18	10	2	0	0	0	0.00249345965853
2e	2	4096	0	0	1	2	5	19	34	40	50	50	29	14	9	3	0	0	0	0.00199448434964
2f	2	4096	0	0	1	3	5	14	34	50	50	40	29	19	9	2	0	0	0	0.00201509911544
2g	2	4096	0	0	1	1	6	22	31	38	52	48	31	17	6	2	1	0	0	0.00197114951919
2h	2	2048	0	0	1	1	7	21	27	42	58	42	27	21	7	1	1	0	0	9.8571644819e-04
3a	3	4096	0	0	0	2	9	19	29	40	50	50	34	14	5	3	1	0	0	4.9915146030e-05
3b	3	6144	0	0	0	3	9	15	28	46	54	46	28	15	9	3	0	0	0	1.1030003050e-04
3c	3	4096	0	0	0	3	9	14	29	50	50	40	34	19	5	2	1	0	0	7.3529911835e-05
3d	3	5120	0	0	0	2	10	18	25	44	56	44	30	18	6	2	1	0	0	6.2748154025e-05
3e	3	2048	0	0	0	3	7	16	36	43	42	48	36	17	7	0	0	1	0	3.6481557826e-05
3f	3	2048	0	0	0	4	8	11	32	52	48	42	32	16	8	3	0	0	0	4.8432371139e-05
3g	3	2048	0	0	0	3	8	15	33	46	44	46	38	15	4	3	1	0	0	3.6623267323e-05
3h	3	512	0	0	0	4	7	12	36	48	42	48	36	12	7	4	0	0	0	1.2072670636e-05
3i	3	2048	0	0	0	3	8	16	32	42	48	52	32	11	8	4	0	0	0	3.6624988237e-05
3j	3	2048	0	0	0	2	8	20	32	37	48	52	32	16	8	0	0	1	0	2.4815863519e-05
4	4	256	0	0	0	0	10	24	28	40	45	40	44	24	0	0	0	0	1	1.8469887415e-07

Table 3.1: The distance profile $\gamma(\tau, j)$ for the received words, with probability calculated for BER of 1.2×10^{-2}

are $\begin{pmatrix} 16 \\ 1 \end{pmatrix} = 16$ received words that are 1 away from it. So there are total $16 \times 256 = 4096$ members in Class 1. Similarly, the number of received words in Class 2 is $\begin{pmatrix} 16 \\ 2 \end{pmatrix} \times 256 = 30720$. For Class 3 there may be one received word that are 3 away from two different valid codewords, so the number of received words after Class 2 cannot be predicted. For the (16,8) quasi-cyclic code in use, there are no received words in Class 5 or higher, as expected for a well designed code.

What is of interest is not only how far a certain received word is to the closest valid codeword but also how far it is to each of the 256 valid codewords. This is done for each of the 65536 possible received words and 22 distinct types is found as shown in Table 3.1, in which each row represents one type.

Assume the valid codewords are transmitted with equal probability over a BSC with a given BER. The probability of each received word in one type can be calculated as shown in the final column of Table 3.1.

Specifically this probability is calculated as follows. Let $BER = p$, $q = 1 - p$. Let n_τ denote the number of received words in type τ, and $\gamma(\tau, j)$ be the number of valid codewords that are j bits away from the received word in type τ, i.e., the element (τ, j) in Table 3.1. The probability of each received word in type τ is:

$$P_\tau = \frac{n_\tau}{256} \sum_{j=0}^{16} p^j (1-p)^{16-j} \gamma(\tau, j) \tag{3.22}$$

Given a BER of 1.2×10^{-2}, the probability for each type is shown in the last column of Table 3.1. The matlab source code to derive this table is given in Appendix B.

The number of slice candidates to be searched in G_0 is given by Equation 3.17. Assuming independently transmitted codewords and independent errors, the mean value of N_0 is:

$$E\{N_0\} = \prod_{i=1}^{L} E\{f_0(x_i)\} \tag{3.23}$$

where $E\{\}$ is the expectation operator. Since x_i are identically distributed,

$$E\{N_0\} = [E\{f_0(x_i)\}]^L \tag{3.24}$$

Taking the expectation of N_1 and N_2 using their derivatin in Equation 3.18 and 3.19,

$$\begin{aligned} E\{N_1\} &= E\{\sum_{j=1}^{L} f_1(x_j) \prod_{i=1, i\neq j}^{L} f_0(x_i)\} \\ &= LE\{f_1(x_i)\}[E\{f_0(x_i)\}]^{L-1} \end{aligned} \tag{3.25}$$

$$
\begin{aligned}
E\{N_2\} &= \sum_{j=1}^{L} E\{f_2(x_j) \prod_{i=1, i\neq j}^{L} f_0(x_i)\} \\
&\quad + \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} E\{f_1(x_i) f_1(x_j) \prod_{k=1, k\neq i,j}^{L} f_0(x_k)\} \\
&= LE\{f_2(x_i)\}(E\{f_0(x_i)\})^{L-1} \\
&\quad + \frac{L(L-1)}{2}[E\{f_1(x_i)\}]^2 \times [E\{f_0(x_i)\}]^{L-2}
\end{aligned}
\tag{3.26}
$$

The calculation of the variance for N_0, N_1, and N_2 is given in Appendix C. Here only the results are given.

$$
var\{N_0\} = E\{N_0^2\} - E^2\{N_0\} = [E\{f_0^2(x_i)\}]^L - [E\{f_0(x_i)\}]^{2L} \tag{3.27}
$$

$$
var\{N_1\} = E\{N_1^2\} - E^2\{N_1\} \tag{3.28}
$$

where

$$
\begin{aligned}
E\{N_1^2\} &= LE\{f_1^2(x_j)\}[E\{f_0^2(x_i)\}]^{L-1} \\
&+ (L^2 - L)[E\{f_1(x_j) f_0(x_j)\}]^2 [E\{f_0^2(x_i)\}]^{L-2}
\end{aligned}
\tag{3.29}
$$

$$
var\{N_2\} = E\{N_2^2\} - E^2\{N_2\} \tag{3.30}
$$

where

$$
\begin{aligned}
E\{N_2^2\} &= LE\{f_2^2(x_j)\}[E\{f_0^2(x_i)\}]^{L-1} \\
&+ (L^2-L)[E\{f_2(x_j)f_0(x_j)\}]^2[E\{f_0^2(x_i)\}]^{L-2} \\
&+ \frac{(L^2-L)}{2}[E\{f_1^2(x_j)\}]^2[E\{f_0^2(x_i)\}]^{L-2} \\
&+ L(L-1)(L-2)E\{f_1^2(x_i)\}[E\{f_1(x_i)f_0(x_i)\}]^2[E\{f_0^2(x_i)\}]^{L-3} \\
&+ \frac{L(L-1)(L-2)(L-3)}{4}[E\{f_1(x_j)f_0(x_i)\}]^4[E\{f_0^2(x_i)\}]^{L-4} \\
&+ 2(L^2-L)E\{f_1(x_i)f_2(x_i)\}E\{f_1(x_i)f_0(x_i)\}[E\{f_0^2(x_i)\}]^{L-2} \\
&+ L(L-1)(L-2)[E\{f_1(x_i)f_0(x_i)\}]^2E\{f_2(x_i)f_0(x_i)\}[E\{f_0^2(x_i)\}]^{L-3}
\end{aligned}
\tag{3.31}
$$

The values of $E\{f_a(x_i)\}$ are calculated as follows. Consider the distribution of distance profile $\gamma(\tau, j)$. For each type τ, the element $\gamma(\tau, j)$ in row τ indicates the number of valid codewords that are j bits away from a received word of type τ. Recall that a received word of type τ is i bits away from a valid codeword where i is the class number corresponding to type τ. Therefore, the first non-zero element in row τ appearing at column J, $\gamma(\tau, J)$, represents the number of candidates (i.e., valid codewords) with the shortest distance from the received word of type τ.The element in row τ at column $(J+a)$, $\gamma(\tau, J+a)$, indicates the number of candidates with the shortest distance plus a from the received word of type τ (i.e., belonging to set B_a). It follows that

$$
E\{f_a(x_i)\} = \sum_{\textit{all types } \tau} \gamma(\tau, J+a) \cdot P_\tau \tag{3.32}
$$

Using the value in Table 3.1 for the (16,8) quasi-cyclic code and $p = 1.2 \times 10^{-2}$, the calculations for $E\{f_a(x_i)\}$ are summarized in Table 3.2.

To estimate the confidence limits of the number of slice candidates in G_0, change

τ	J	$\gamma(\tau, J)$	$\gamma(\tau, J+1)$	$\gamma(\tau, J+2)$	P_τ
0	0	1	0	0	0.82434900147619
1a	1	1	0	0	0.08009986458629
1b	1	1	0	0	0.08009972289769
2a	2	1	3	4	0.00100890786913
2b	2	1	2	5	0.00249310113476
2c	2	1	2	6	0.00199476428500
2d	2	1	2	6	0.00249345965853
2e	2	1	2	5	0.00199448434964
2f	2	1	3	5	0.00201809911544
2g	2	1	1	6	0.00197114951919
2h	2	1	1	7	9.8571644819e-04
3a	3	2	9	19	4.9915146030e-05
3b	3	3	9	15	1.1030003050e-04
3c	3	3	9	14	7.3529911835e-05
3d	3	2	10	18	6.2748154025e-05
3e	3	3	7	16	3.6481557826e-05
3f	3	4	8	11	4.8432371139e-05
3g	3	3	8	15	3.6623267323e-05
3h	3	4	7	12	1.2072670636e-05
3i	3	3	8	16	3.6624988237e-05
3j	3	2	8	20	2.4815863519e-05
4	4	10	24	28	1.8469887415e-07
		$E\{f_0(x_i)\}$	$E\{f_1(x_i)\}$	$E\{f_2(x_i)\}$	
		1.0009	0.0342	0.0899	

Table 3.2: Values of $E\{f_a(x_i)\}$ for $p = 1.2 \times 10^{-2}$ and (16,8) quasi-cyclic code

N_0 to log scale,

$$\log N_0 = \sum_{i=1}^{L} \log f_0(x_i) \tag{3.33}$$

From the above assumptions, $f_0(x_i)$ are discrete independent random variables with identical distribution, and $\log f_0(x_i)$ defines a one-to-one transformation between the values of $f_0(x_i)$ and $\log f_0(x_i)$, thus $\log f_0(x_i)$ are identical distributed independent random variables with distributions derived from Table 3.2. Consequently, the expectation of $\log N_0$ can be calculated as:

$$E\{\log N_0\} = \sum_{i=1}^{L} E\{\log f_0(x_i)\} = LE\{\log f_0(x_i)\} \tag{3.34}$$

From the central limit theorem[11], the sum of of statistically independent and identically distributed random variables with finite mean and variance approaches a Gaussian cdf as $L \to \infty$. In practice, when $L > 30$, the distribution of the sum can be looked as Gaussian. So $\log N_0$ approaches a Gaussian cdf when $L > 30$.

From Table in [33], the 90% confidence interval of $\log N_0$ is

$$\begin{aligned} LE\{\log f_0(x_i)\} - 1.645\sqrt{L}var\{\log f_0(x_i)\} \quad &< \quad \log N_0 < \\ LE\{\log f_0(x_i)\} + 1.645\sqrt{L}var\{\log f_0(x_i)\} & \end{aligned} \tag{3.35}$$

Use the Chebyshev's theorem [33]

$$P(\mu - k\sigma < X < \mu - k\sigma) \geq 1 - \frac{1}{k^2} \tag{3.36}$$

if $1 - \frac{1}{k^2} = 90\%$, then the range of N_0 is:

$$E\{N_0\} - \sqrt{10}var\{N_0\} < N_0 < E\{N_0\} + \sqrt{10}var\{N_0\} \tag{3.37}$$

These two limits for 90% confidence interval for different length is plotted in Figure

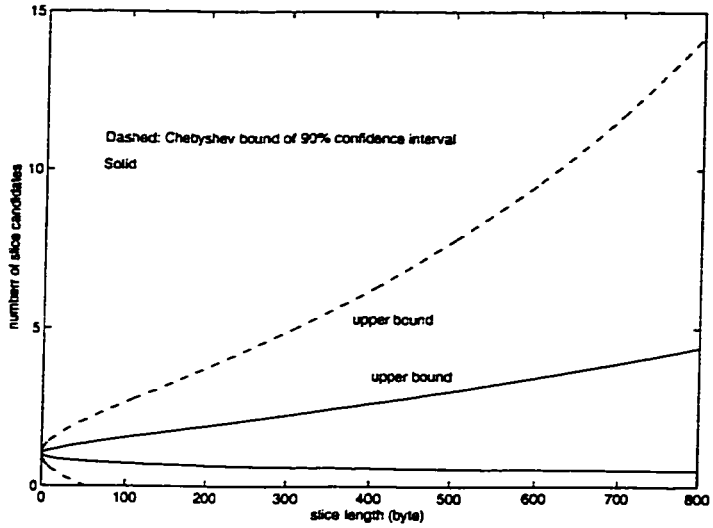

Figure 3.8: Central limit and Chebyshev limits for 90%confidence interval for N_0

3.8. Since N_0, N_1 and N_2 are larger than 0, when the limits are less than zero, they are set to zero.

Figure 3.8 shows that the Chebyshev confidence limit is looser than the central limit confidence limit.

Because of the nature of the equations for N_1 and N_2 only the Chebyshev confidence limit can be calculated. The curves are shown in Figures 3.9, 3.10, respectively:

3.4.2 Number of bits extracted from the bitstream

Since all slices do not have the same length, and the slice candidates that cause syntax violation are rejected and are not completely decompressed, a more exact measure of complexity than the number of slice candidates to be decompressed may be formulated.

Such a measure is a running number of bits extracted from the bitstream. If a slice candidate is fully decompressed, then the total length of the slice in bits is added to the bits extracted. But if a slice which is 1000 bits long has a syntax violation which is caught after 400 bits, this slice would not be decompressed fully and would only add 400 bits to the number of bits extracted. Each slice candidate

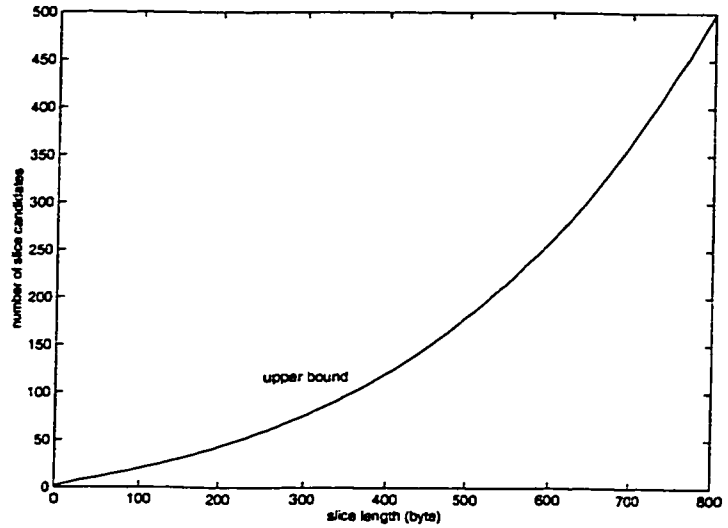

Figure 3.9: Chebyshev's limit for 90%confidence interval for N_1

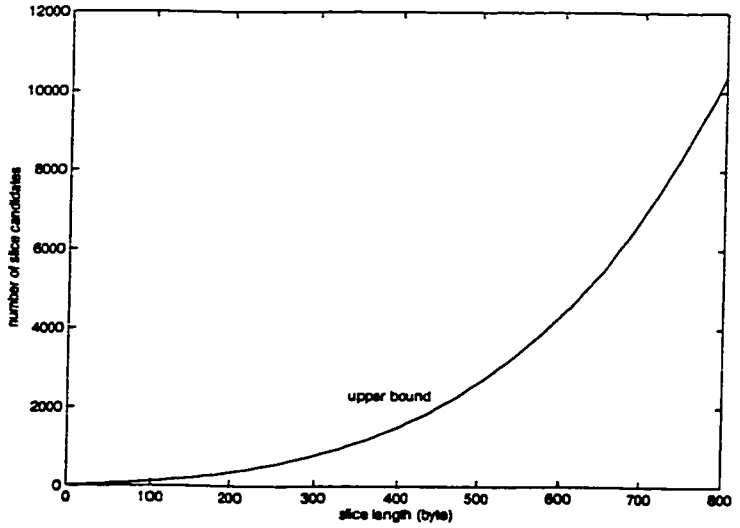

Figure 3.10: Chebyshev's limit of 90%confidence interval for N_2

decompressed adds its number of bits extracted to the total number of bits extracted. The total number of bits extracted can be compared to the size of the bitstream to indicate complexity. This complexity indicator can be measured empirically in the experiments. Unfortunately unlike the number of slice candidates it is harder to predict theoretically.

3.4.3 Effect of FEC code length on complexity

As previously discussed, a slice is divided into L message portions of received words. L decreases as the code length increases. For example, suppose there are 8000 bits in one slice. If a (16,8) binary code is used, then $L = 1000$. For a (26,16) binary code, $L = 500$. As L decreases, the number of slice candidates in G_a decreases for lower complexity.

But longer FEC code increases the complexity of FEC decoder. For example, a table lookup scheme is used for decoding of an (n,k) binary code. A table of 2^n is set up and used to decode the FEC codewords. As n increase, the entries of the table increase exponentially. For example, the (26,16) FEC code has a 2^{26} entries, which is 2^{10} times that of the (16,8) FEC code. The (26,16) FEC code has a larger complexity to set up the table. Thus when FEC codes that have the same error correction capability but different length are selected, there is a trade-off between the complexity of FEC decoder and decompressor.

Chapter 4

Performance Evaluation

In this chapter, the performance of the proposed scheme is objectively and subjectively evaluated. PSNR (introduced in Section 1.3) is used as the objective measure. As an illustrative indication of subjective picture quality, some frames from the sequence and comments on the sequence observed on the computer are given. Performance comparison of the proposed scheme and a classic FEC scheme using the same channel code, as well as a case without FEC is presented.

Two 2Mb/s MPEG2 sequences "Table Tennis", and "Flower Garden" are used in the simulations, each one consisting of 180 frames at 30 frames/sec. For "Table Tennis", the frame size is $704 * 480$ for the luminance part and subsampled in both spatial directions for the color components (4:2:0 format). So the compression rate is 60:1. For "Flower Garden", the frame size of the luminance component is also $704 * 480$, while the chrominance is subsampled only horizontally but not vertically (4:2:2 format). The compression rate is is 80:1. These two sample sequences are obtained from the homepage of MPEG [12].

The simulations are done with software (C language). It runs under the Sun system (unix). The sequences are played under the Sun system to be observed its subjective quality. All experiments are conducted five times and the results presented are the average of these five trials. Complexity of the proposed scheme is also discussed.

4.1 Performance of classical ML receiver using (16,8) quasi-cyclic code

In this experiment, sequence "Table Tennis" is used. The simulation uses the block diagram shown in Figure 2.4 with a (16,8) quasi-cyclic code described in Appendix A and a binary symmetric channel (BSC). A table look-up ML decoder is implemented. For a 16-bit received word r, it produces a valid codeword c_j with the shortest Hamming distance. Figure 4.1 shows the performance comparison between the cases with and without FEC in terms of PSNR versus the channel BER for luminance, blue chrominance and red chrominance.

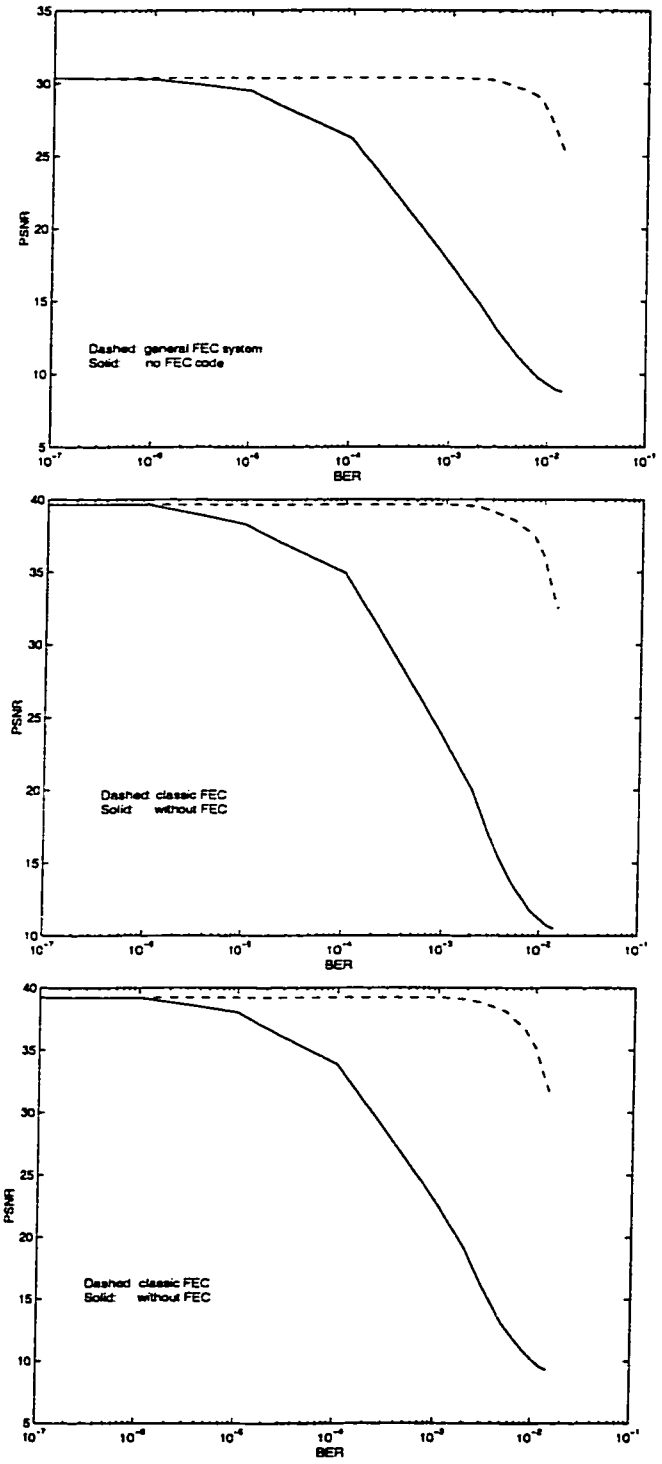

Figure 4.1: PSNR versus channel BER for "Table Tennis", without and with FEC (from top t0 bottom: Luminance, Blue Chrominance, Red Chrominance)

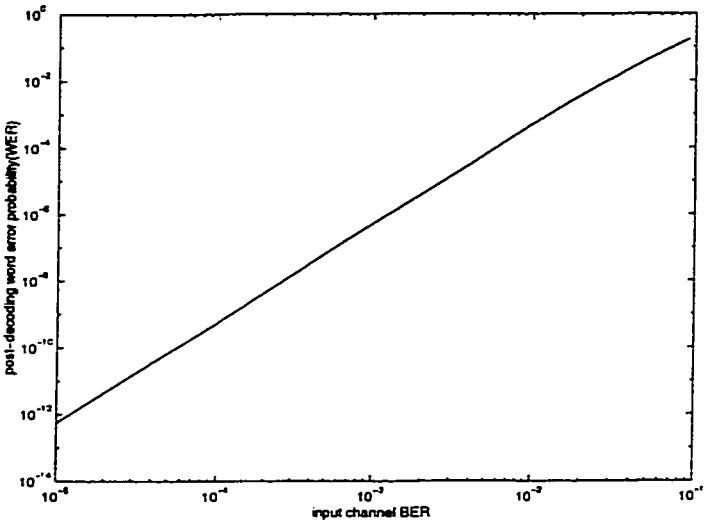

Figure 4.2: Post decoding error probability versus input channel BER for the (16,8) quasi-cyclic code

Without FEC, the PSNR is not degraded for the channel BER of 10^{-6} or better. By using the (16,8) quasi-cyclic code with ML decoder, the PSNR can be kept unchanged for a channel BER as high as 3×10^{-3}. The results are not surprising since for an input channel BER of 3×10^{-3}, the (16,8) quasi-cyclic code provides a post-decoding word error probability of 1.6×10^{-5} as shown in Figure 4.2 or a post-decoding BER of 10^{-6}.

As an indication of subjective quality, three frames in each sequence are given in Figures 4.3 - 4.5. As shown in the middle pictures, without FEC, the reconstructed picture is still recognizable for a channel BER of 10^{-4} (Figure 4.3) although there is a noticeable black stripe. As the channel BER increases to 2×10^{-3} (Figure 4.4) or more (Figure 4.5), the middle picture is not recognized. On the other hand, with a (16,8) quasi-cyclic code and ML decoding, the bottom pictures of Figures 4.3 - 4.5 are recognizable even for a channel BER of 1.2×10^{-2}(Figure 4.5). It is also noted that at the channel BER of 1.2×10^{-2}, there are noticeable black stripes.

When the sequence is played on the computer, an improvement over the corrupted sequence in terms of subjective quality can be seen. In the corrupted sequences, when channel BER is high, the errors cause the discarding of a portion of

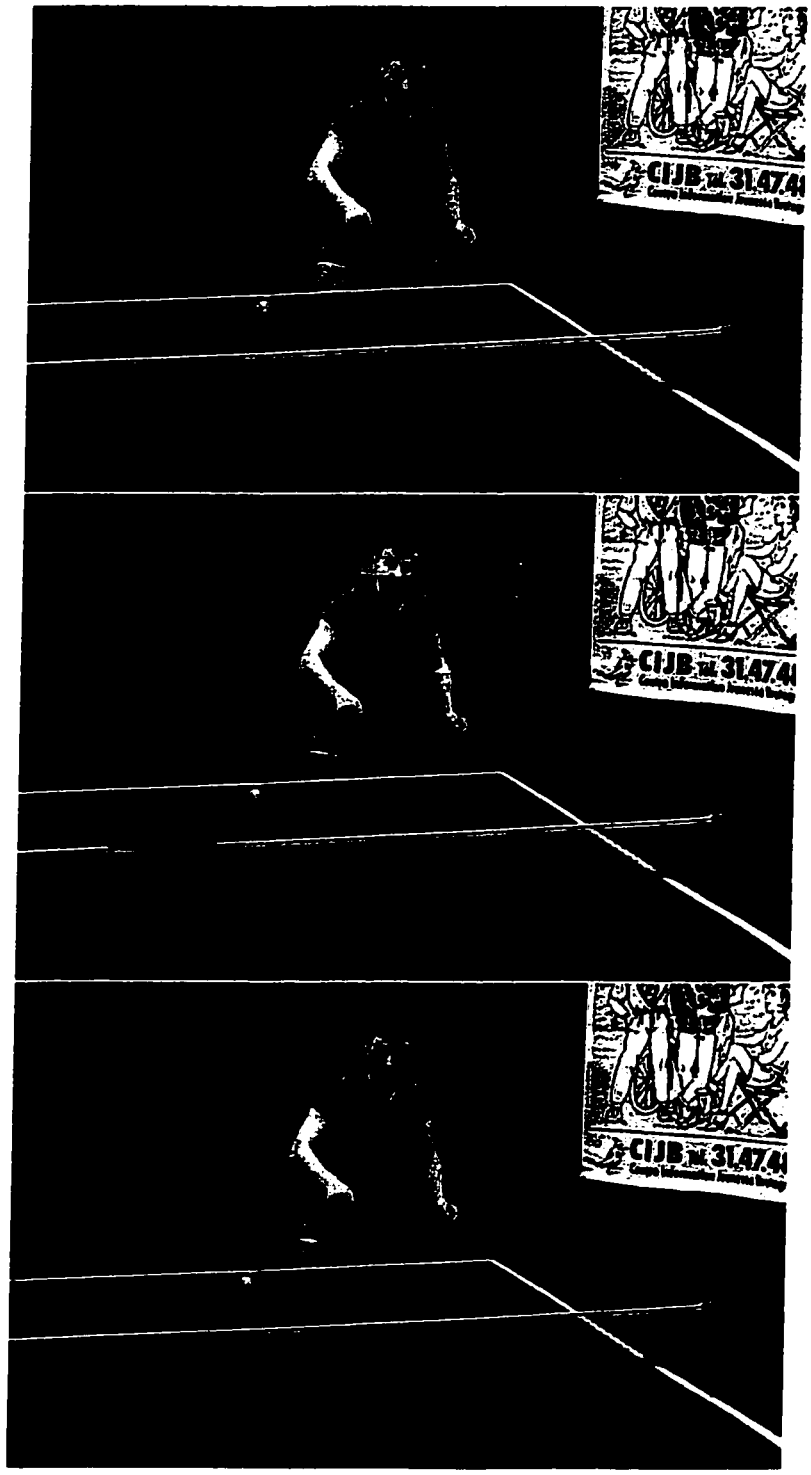

Figure 4.3: Frames 120 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, without FEC and with channel BER of 1×10^{-4}, with classic FEC and with channel BER of 1×10^{-4}

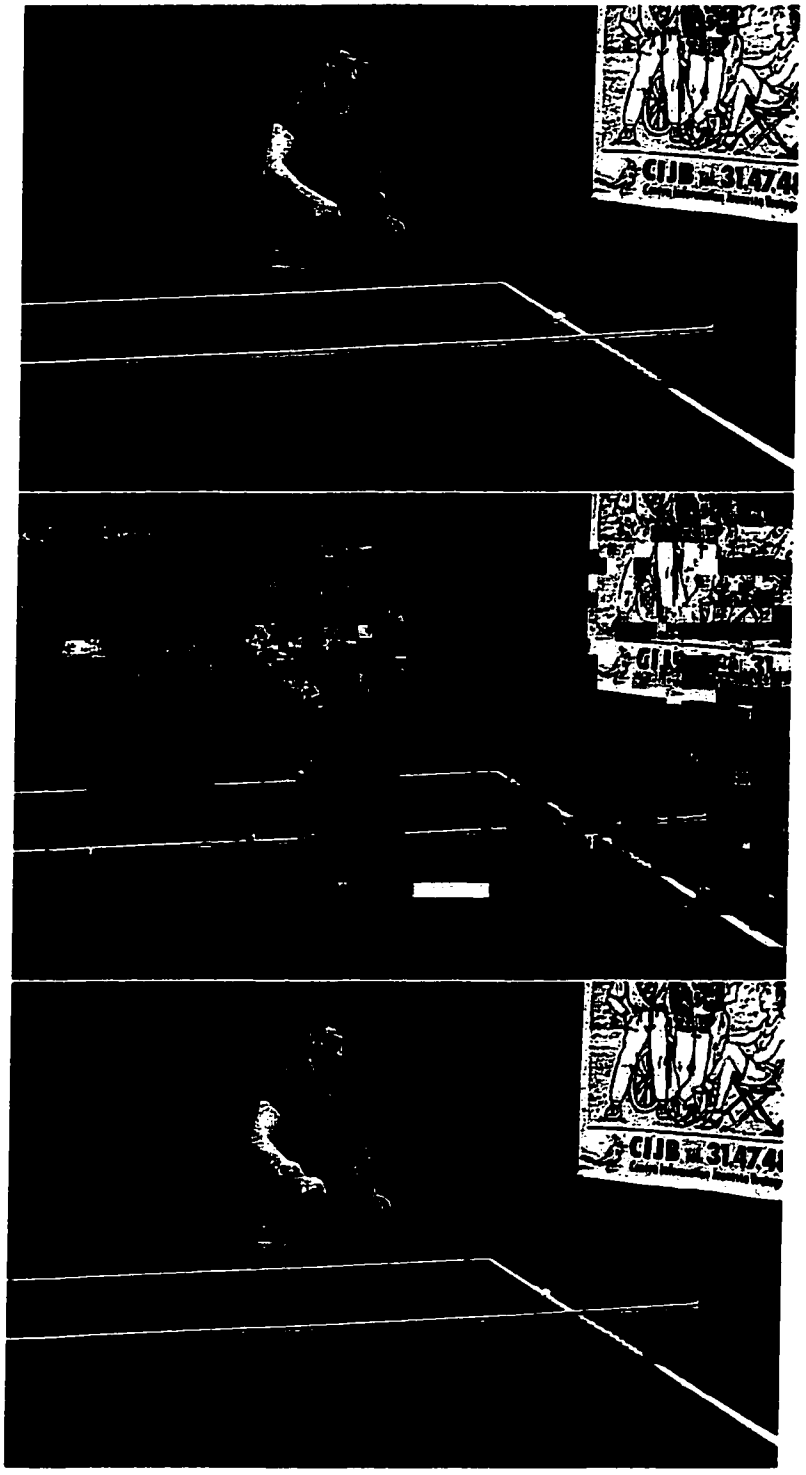

Figure 4.4: Frames 69 from compressed "Table Tennis" (2Mb/s): from top to bottom: no error, without FEC and with channel BER of 2×10^{-3}, with classic FEC and with channel BER of 2×10^{-3}

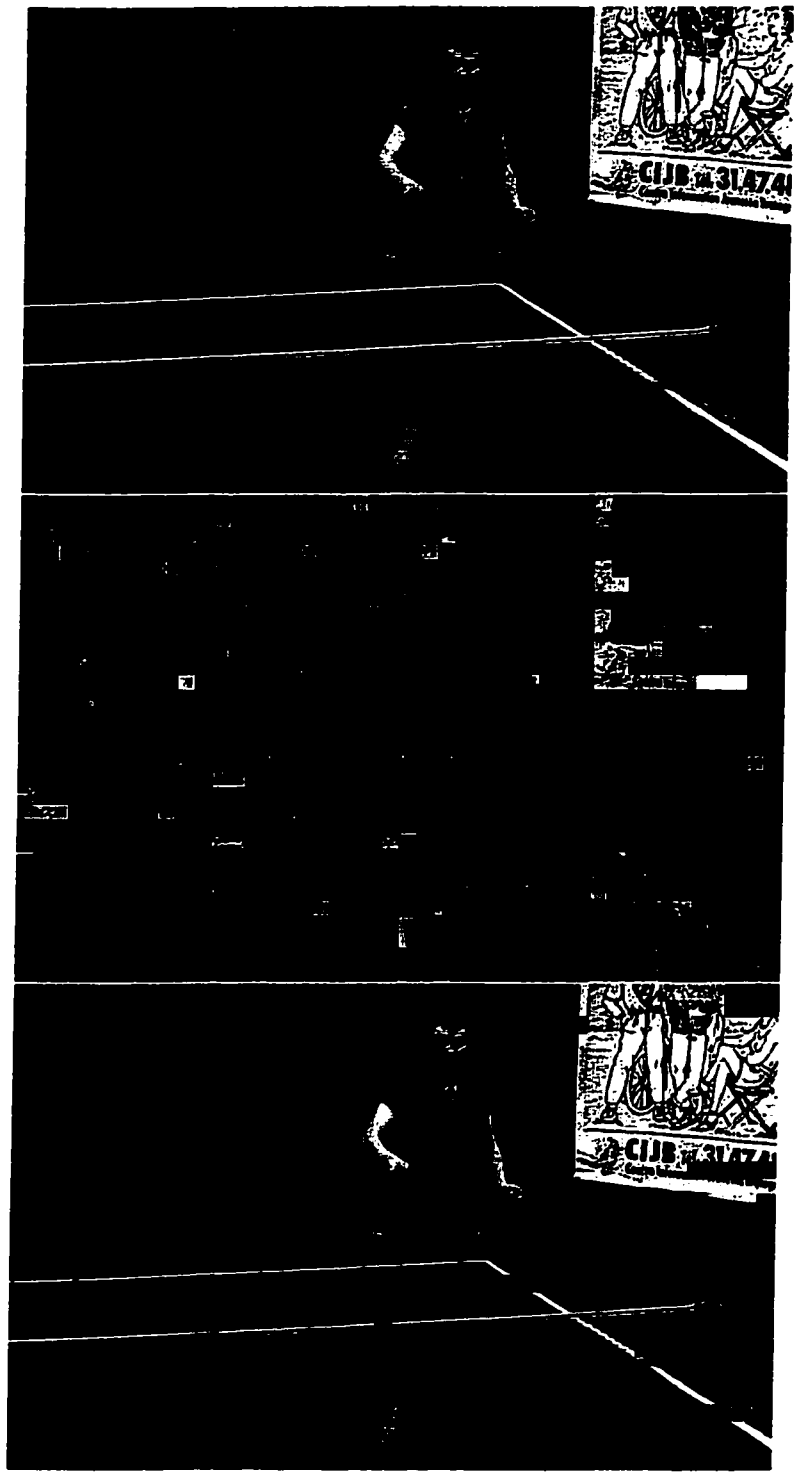

Figure 4.5: Frames 0 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, without FEC and with channel BER of 1.2×10^{-2}, with classic FEC and with channel BER of 1.2×10^{-2}

the slice or a whole slice in many places. There are some shifts in the sequences as well. The sequence is not recognized. But the sequence after error correction is much better. Many black stripes are removed and many shifts are corrected.

Simulation is also conducted for the sequence "Flower Garden". Figure 4.6 shows the PSNR versus channel BER for the case without and with FEC. Similar results to Figure 4.1 are obtained, i.e., the PSNR remains unchanged for BER upto 10^{-6}(without FEC) and 3×10^{-3} (with FEC).

Figures 4.30 - 4.9 show the sample pictures for different channel BER's. The results again indicate that with a (16,8) quasi-cyclic code and a traditional ML decoder, pictures are still recognized for a channel BER upto 1.2×10^{-2}. Without FEC, recognizable pictures can only be obtained for a channel BER better than 10^{-4}.

4.2 Performance of the proposed combined FEC/EC scheme using a (16,8) quasi-cyclic code

4.2.1 Using slice candidates with shortest distance

All slice candidates in group G_0 with shortest distance are examined by the syntax checker and discontinuity measure selector and the one with no syntax violation and the lowest discontinuity measure is selected as the solution. This experiment is called SSG0 (search space group G_0). The same (16,8) quasi-cyclic code is used for performance comparison of the classic FEC and proposed scheme.

Figure 4.10 indicates that the proposed SSG0 provides an improvement in PSNR as compared to the classical FEC for a range of channel BER from 2×10^3 to 1.4×10^{-2}.

As an illustrative example of subjective equality, Figures 4.11 - 4.13 show the comparison of pictures obtained by the classical ML decoder (classic FEC) and proposed MCL decoder with G_0 (SSG0) for a channel BER of 1.2×10^{-2}. In these

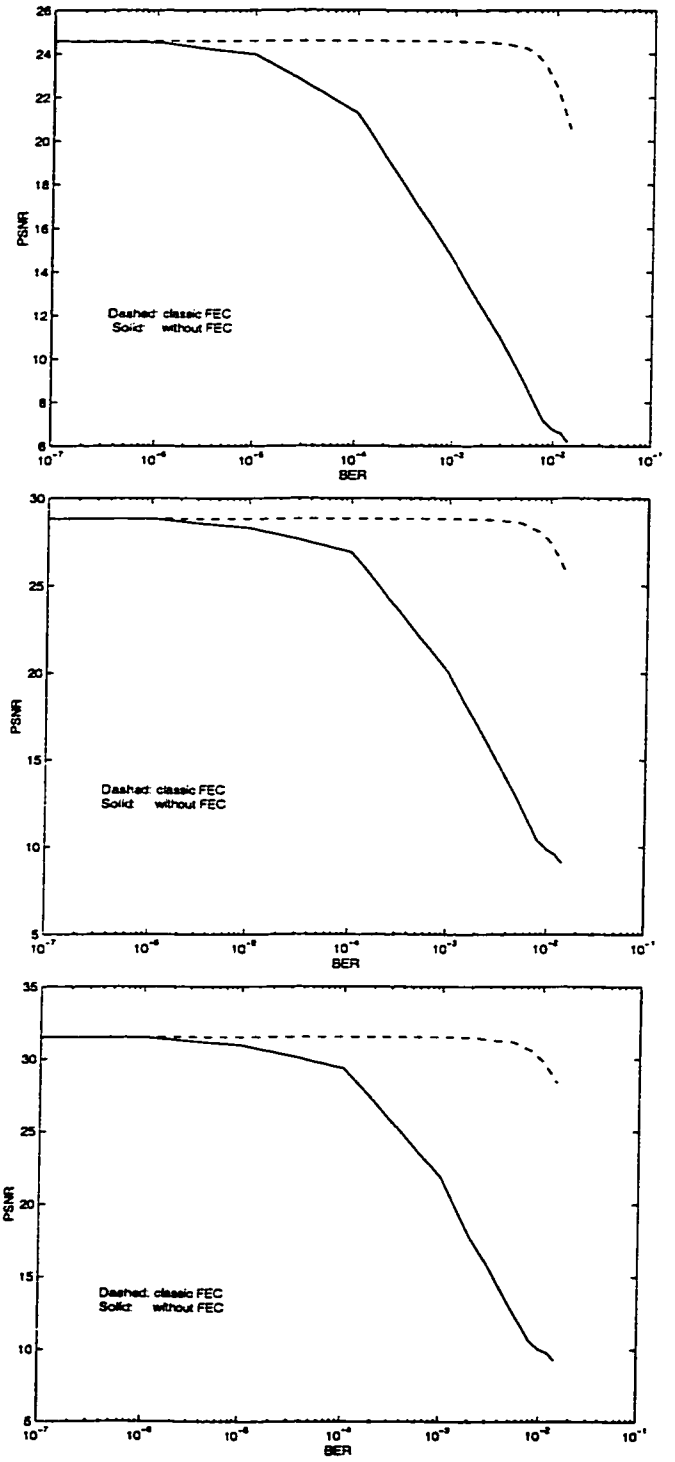

Figure 4.6: PSNR versus channel BER for "Flower Garden" , without FEC and with FEC (from top to bottom: Luminance, Blue Chrominance, and Red Chrominance)

Figure 4.7: Frames 0 from compressed "Flower Garden" (2Mb/s), from top to bottom: no error, without FEC and with channel BER of 1×10^{-4}, with classic FEC and with channel BER of 1×10^{-4}

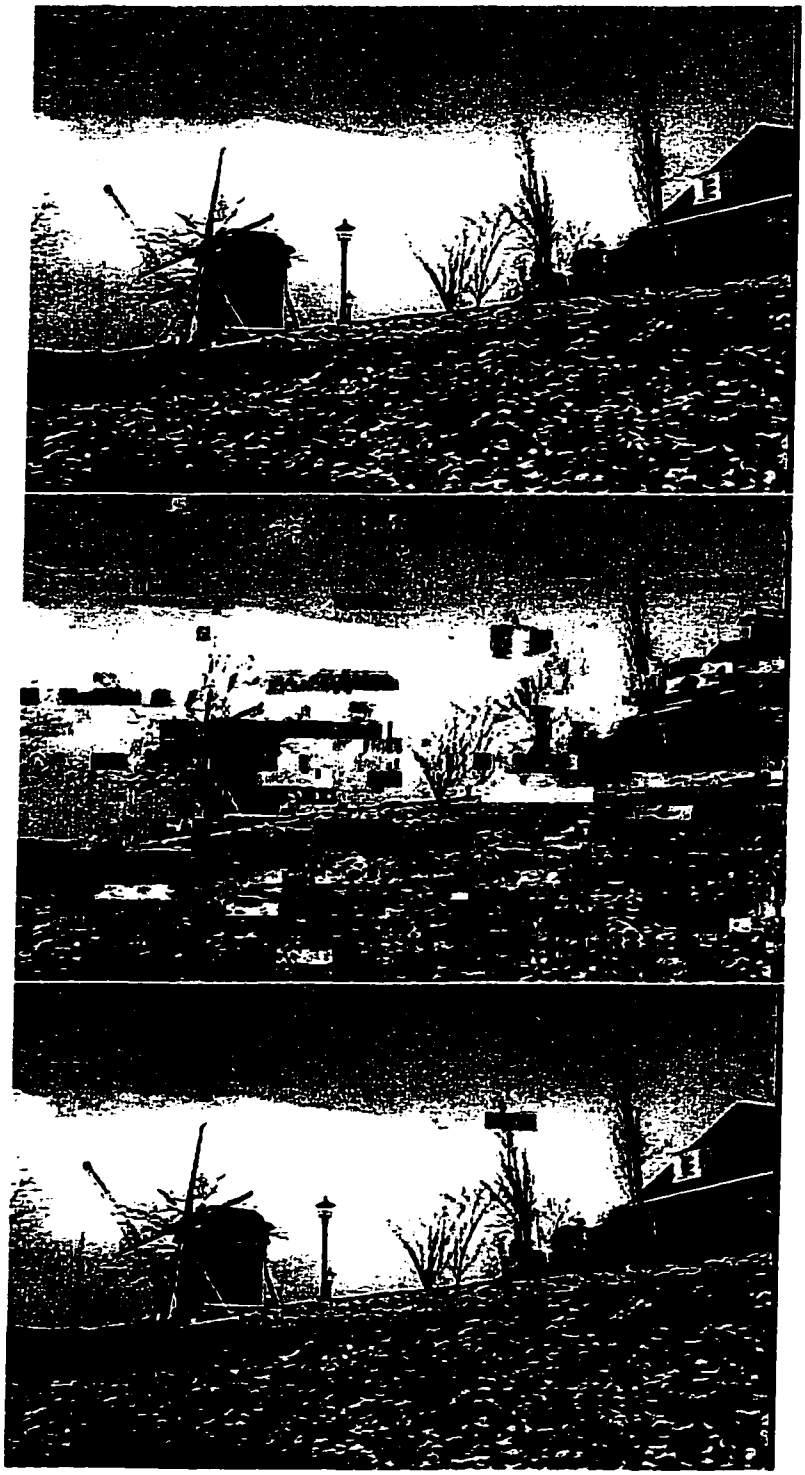

Figure 4.8: Frames 136 from compressed "Flower Garden" (2Mb/s), from top to bottom: no error, without FEC and with channel BER of 2×10^{-3}, with classic FEC and with channel BER of 2×10^{-3}

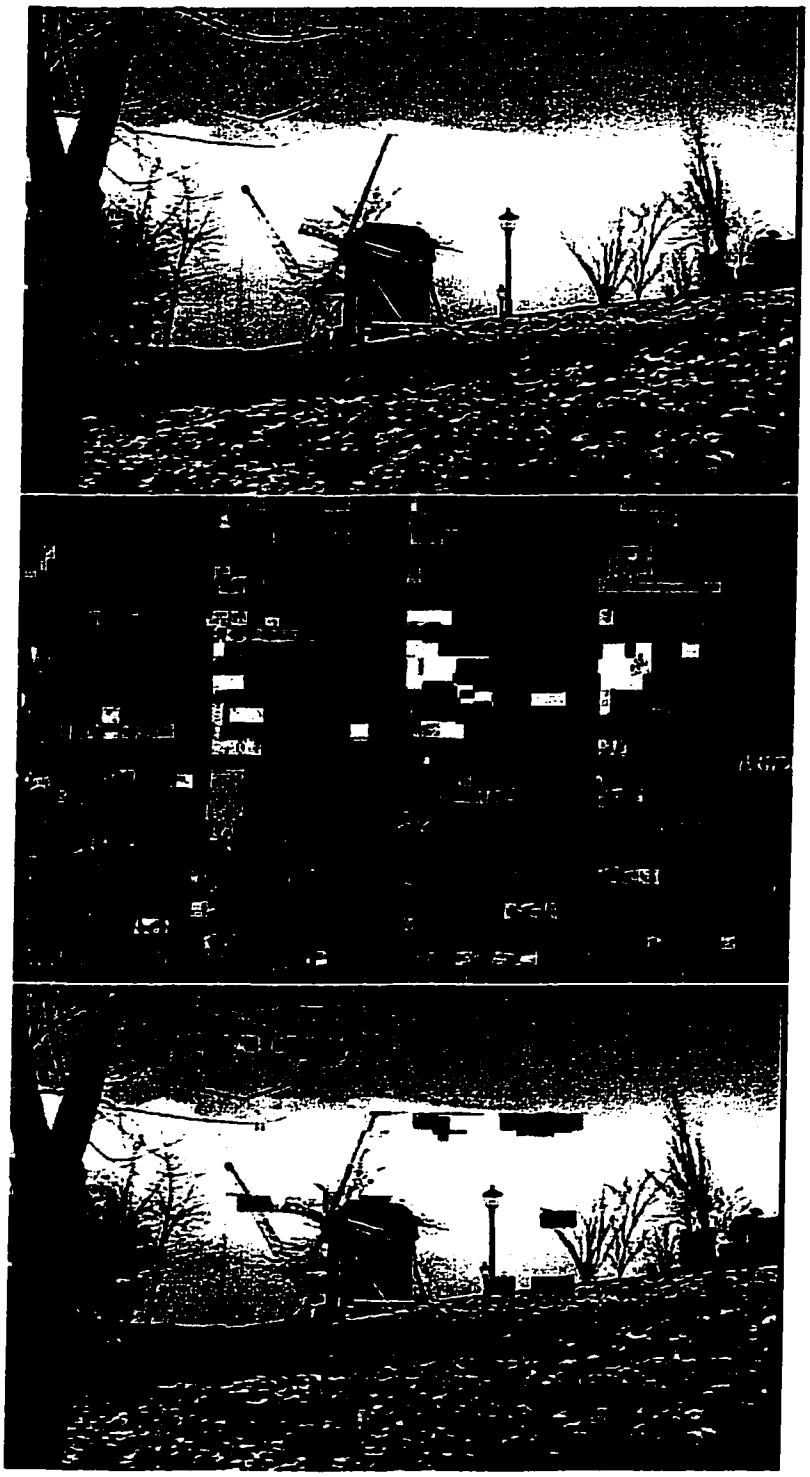

Figure 4.9: Frames 14 from compressed "Flower Garden" (2Mb/s), from top to bottom: no error, without FEC and with channel BER of 1.2×10^{-2}, with classic FEC and with channel BER of 1.2×10^{-2}

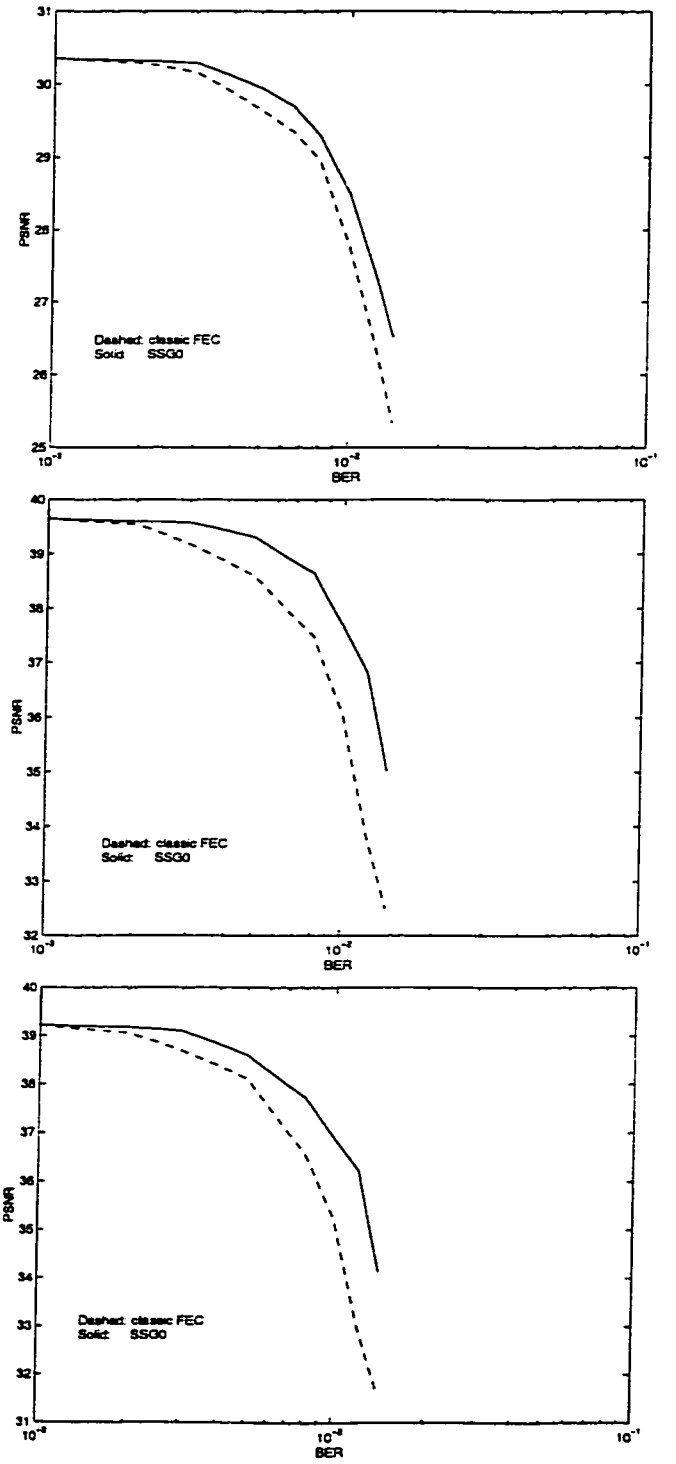

Figure 4.10: PSNR versus BER for "Table Tennis", classic FEC and SSG0, from top to bottom: Luminance, Blue Chrominance and Red Chrominance

pictures, the top pictures are the case of no channel error and can be used as reference, The proposed scheme (bottom pictures) offers a better performance than the classical ML decoder (middle pictures).

When the whole sequence is observed on the computer, the improvement of SSG0 over classic FEC in the subjective picture quality is also noticed. With the classic FEC, there are still some errors that are serious enough to cause the discarding of a portion of the slices. There are still some shifts. The sequence after SSG0 looks better. Most black stripes are removed and most shifts are corrected.

4.2.2 Using slice candidates in G_0 and G_1

In this experiment, called SSG1, the MCL decoder provides candidates in groups G_0 (shortest distance) and G_1 (shortest-distance-plus-one).

Let $\mathcal{S}_{0i}$'s and $\mathcal{S}_{1j}$'s denote the slice candidates in G_0 and G_1, respectively. The syntax/semantic checker consider all of them for validity. Only the slice candidates with valid syntax/semantic are further examined for the discontinuity measure. As discussed in Section 3.3.5, the modified discontinuity measure of a slice candidate $\mathcal{S}_{ki}$ defined as

$$\mathcal{M}_{ki} \triangleq M_{ki} + \alpha_k \tag{4.1}$$

where $k = 0$ or 1 and $\alpha_0 = 0$, $\alpha_1 > 0$. M_{ki} is the discontinuity measure defined in Section 3.1.2. The slice candidate $\mathcal{S}_{ki}$ is finally selected as the solution if it satisfied the syntax\semantic rules and has the lowest value $\mathcal{M}_{ki}$. It can be seen that G_1 has more candidates than G_0. The experiment SSG1 can provide a better PSNR than the previous SSG0, only if there are slice candidates in G_1, which are finally selected as solutions with larger PSNR. On the other hand, if slice candidates in G_1 are more often selected than slice candidates in G_0, the overall performance of SSG1 can be worse than SSG0. In this sense, α_1 is selected to maximize the PSNR. Figure 4.14 shows the PSNR versus α_1 for a channel BER of 1.2×10^{-2}. It indicates

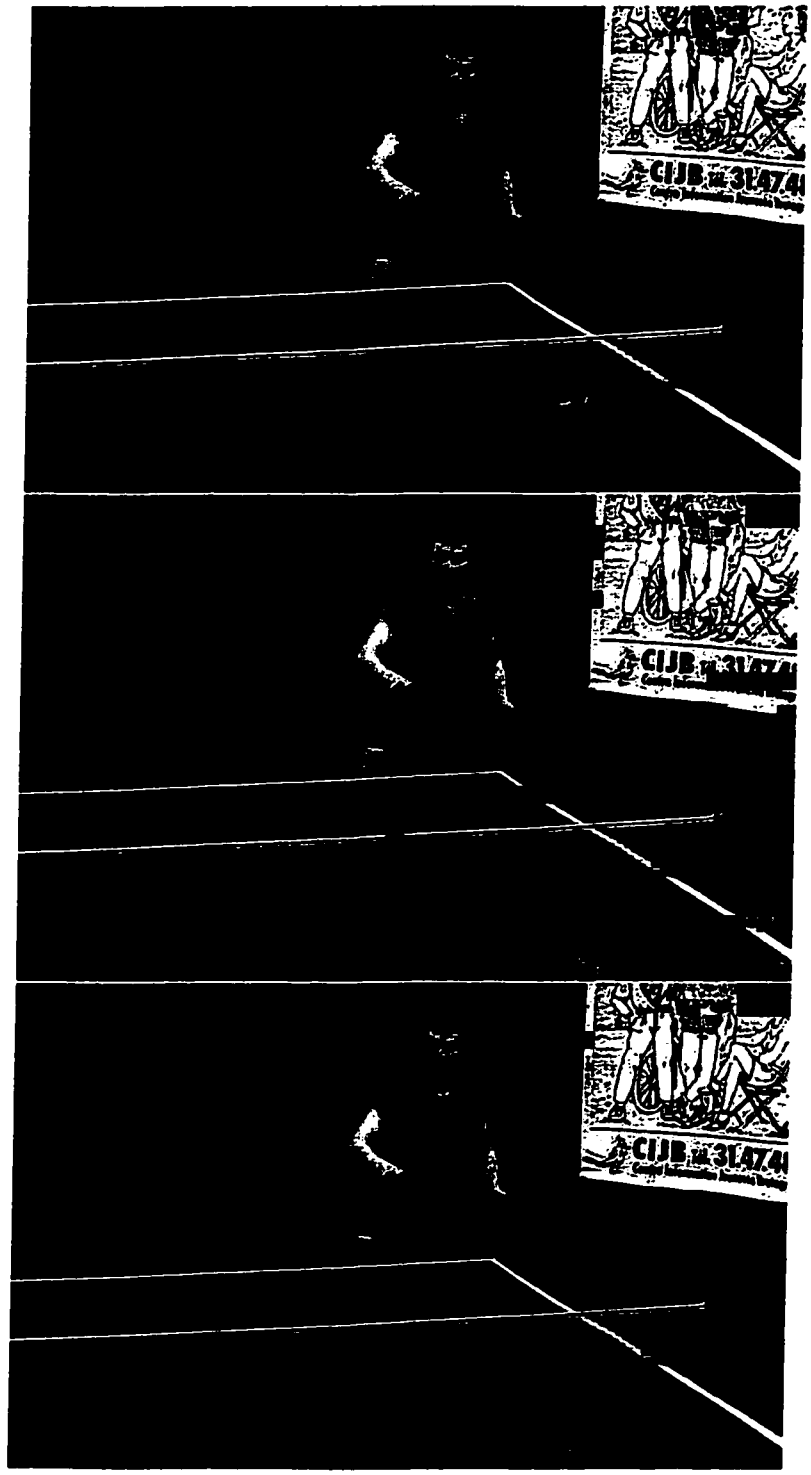

Figure 4.11: Frames 2 from compressed "Table Tennis" (2Mb/s): from top to bottom: no error, with classic FEC and with channel BER of 1.2×10^{-2}, with SSG0 and with channel BER of 1.2×10^{-2}

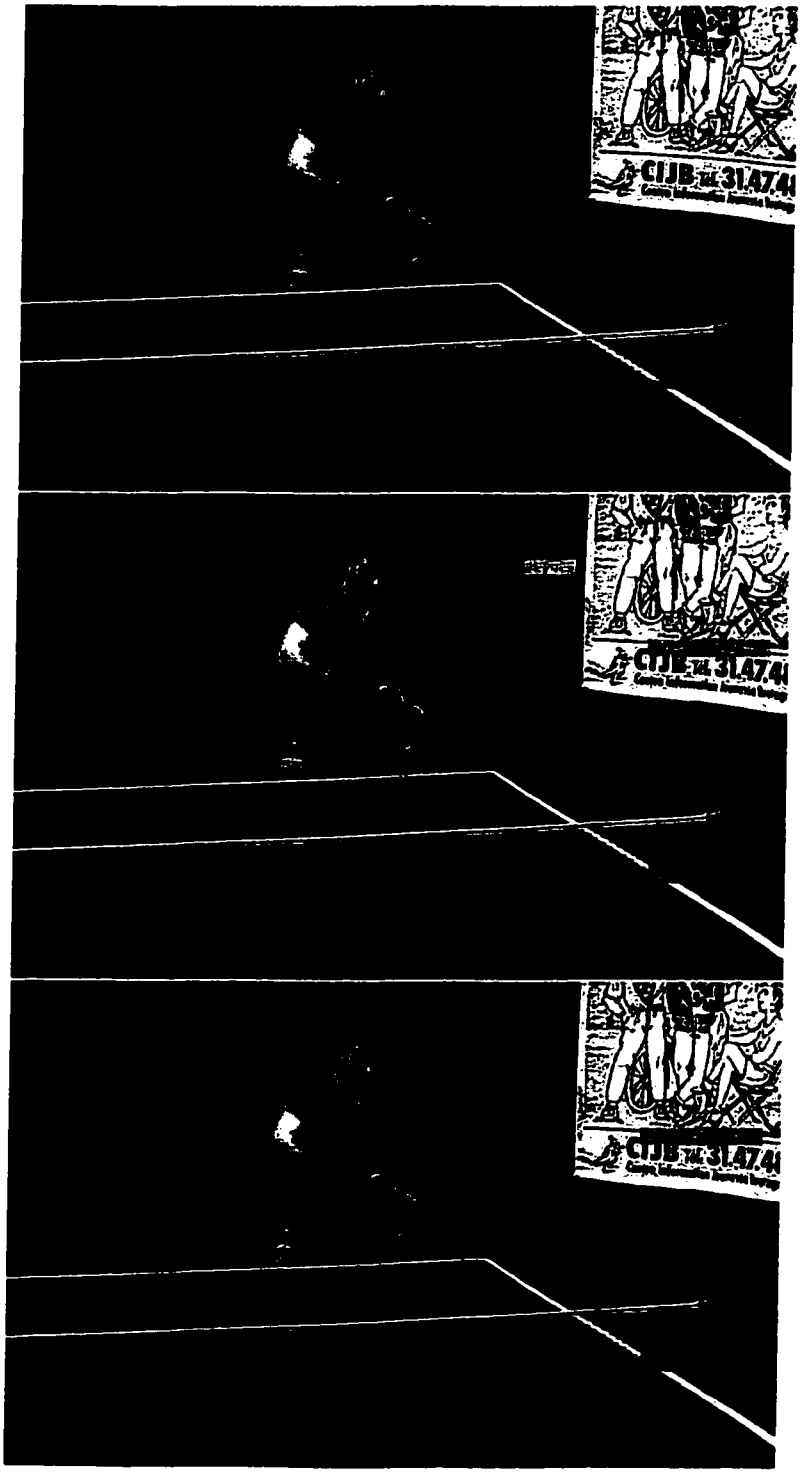

Figure 4.12: Frames 135 from compressed "Table Tennis" (2Mb/s): from top to bottom: no error, with classic FEC and with channel BER of 1.2×10^{-2}, with SSG0 and with channel BER of 1.2×10^{-2}

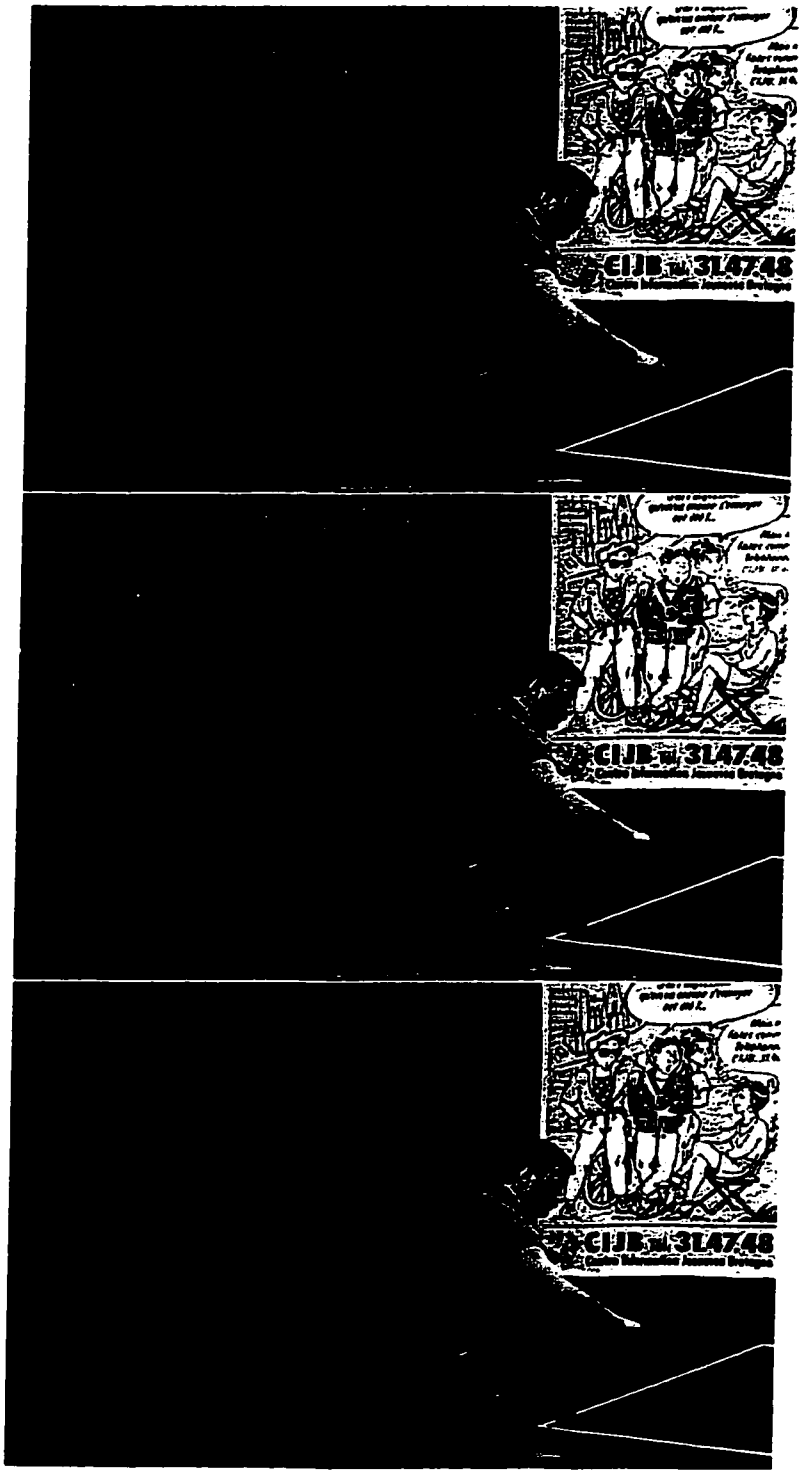

Figure 4.13: Frames 179 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with classic FEC and with channel BER of 1.2×10^{-2}, with SSG0 and with channel BER of 1.2×10^{-2}

α_1	PSNR(Y)	PSNR(Cb)	PSNR(Cr)
1	27.31	36.56	36.09
30	28.50	37.51	37.04
50	28.44	37.43	36.78
75	28.44	37.43	36.78

Table 4.1: PSNR for different α_1, at BER of 1.2×10^{-2}.

that for small value of α_1, the performance of SSG1 is worse than that of SSG0. This confirms that when α_1 is small, more slice candidates from G_1 are selected over those from G_0 with poorer overall PSNR. As α_1 increases, the PSNR offered by SSG1 increases and becomes better than that of SSG0 until the PSNR gets to its peak value at around $\alpha_1 = 30$ in the simulation. For $\alpha_1 > 30$, the PSNR offered by SSG1 drops. For larger α_1, the PSNR keeps unchanged.

The characteristics of the "PSNR versus α_1" curve can be explained as follows. By increasing α_1, the modified discontinuity measure of slice candidates from group G_1 is increased intentionally and hence reduce the chance that a slice candidate in group G_1 is selected over the one from group G_0. For a sufficiently large value of α_1, slice candidates from group G_1 is selected only when no slice candidate from G_0 satisfies the syntax\semantic rules. In this case, the selection algorithm tends to be the following. If there are some slice candidates from group G_0 with valid syntax\semantic, select the one with lowest discontinuity measure. Otherwise consider slice candidates from group G_1 using the same rule.

The results plotted in Figure 4.14 and summarized in Table 4.1 show that the scheme SSG1 is robust for large α_1, i.e., the PSNR remains unchanged and higher than that of SSG0.

Using $\alpha_1 = 30$, Figure 4.15 shows that SSG1 provides an improvement in PSNR as compared to SSG0 for a range of channel BER from 10^{-3} to 1.4×10^{-2}.

As an illustrative indication of subjective quality, Figures 4.16 - 4.18 show the comparison pictures obtained by SSG0 and SSG1. SSG1 (bottom pictures) offers a better performance than SSG0 (middle pictures).

When the whole sequence is observed on the computer, the improvement of

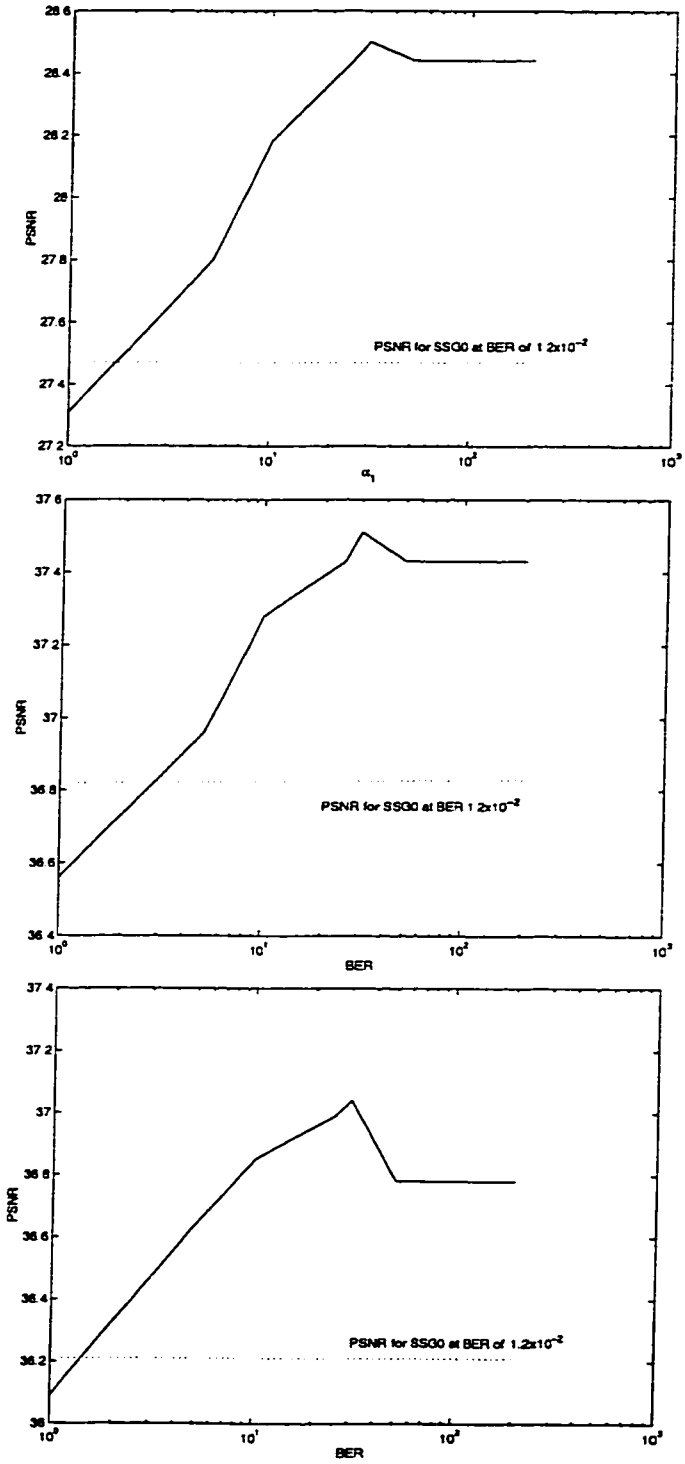

Figure 4.14: PSNR versus α_1 for "Table Tennis" at BER of 1.2×10^{-2}, from top to bottom: Luminance, Blue Chrominance, and Red Chrominance

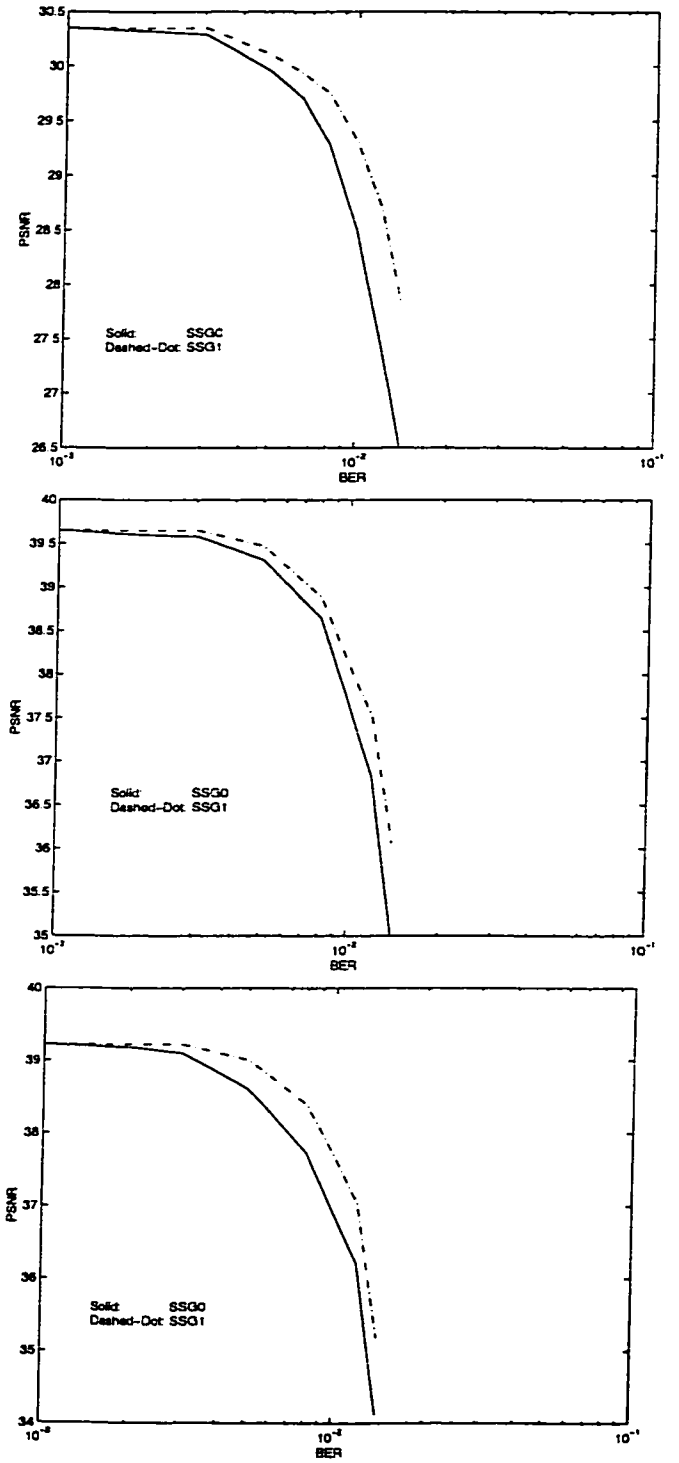

Figure 4.15: PSNR versus BER for "Table Tennis" (2Mb/s), with SSG0 and SSG1, from top to bottom: Luminance, Blue Chrominance, and Red Chrominance

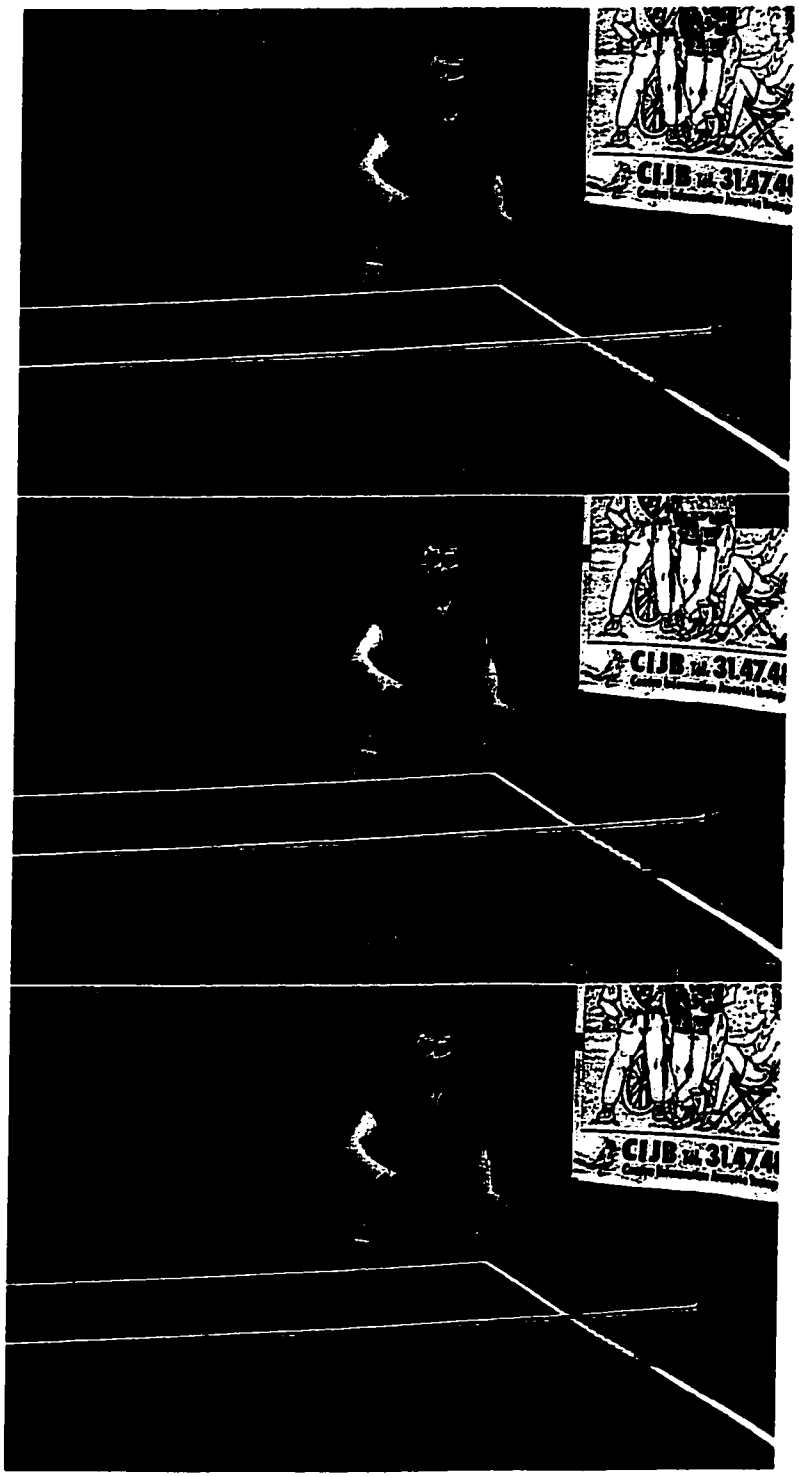

Figure 4.16: Frames 2 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with SSG0 and with channel BER of 1.2×10^{-2}, with SSG1 and with BER of 1.2×10^{-2}

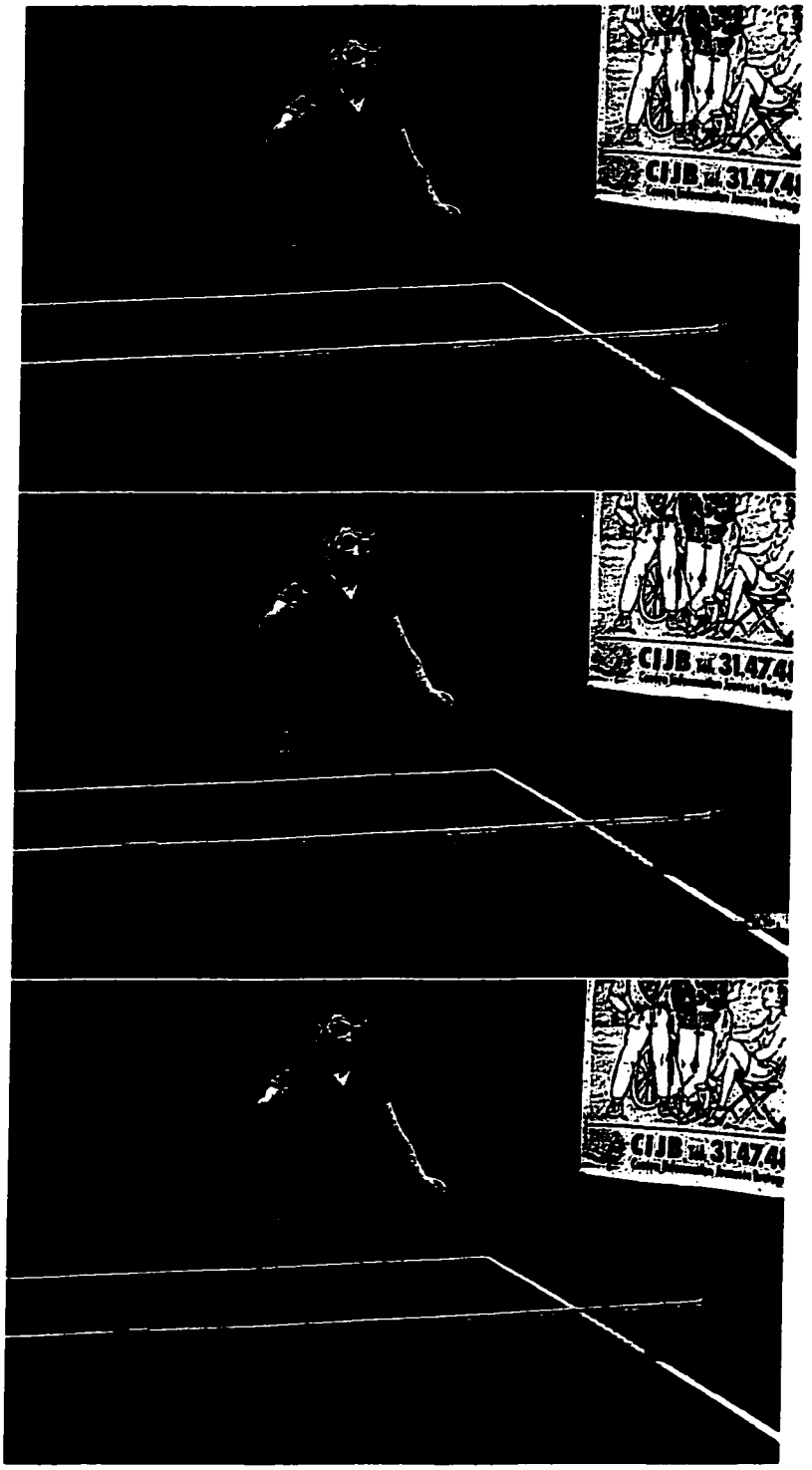

Figure 4.17: Frames 46 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with SSG0 and with channel BER of 1.2×10^{-2}, with SSG1 and with channel BER of 1.2×10^{-2}

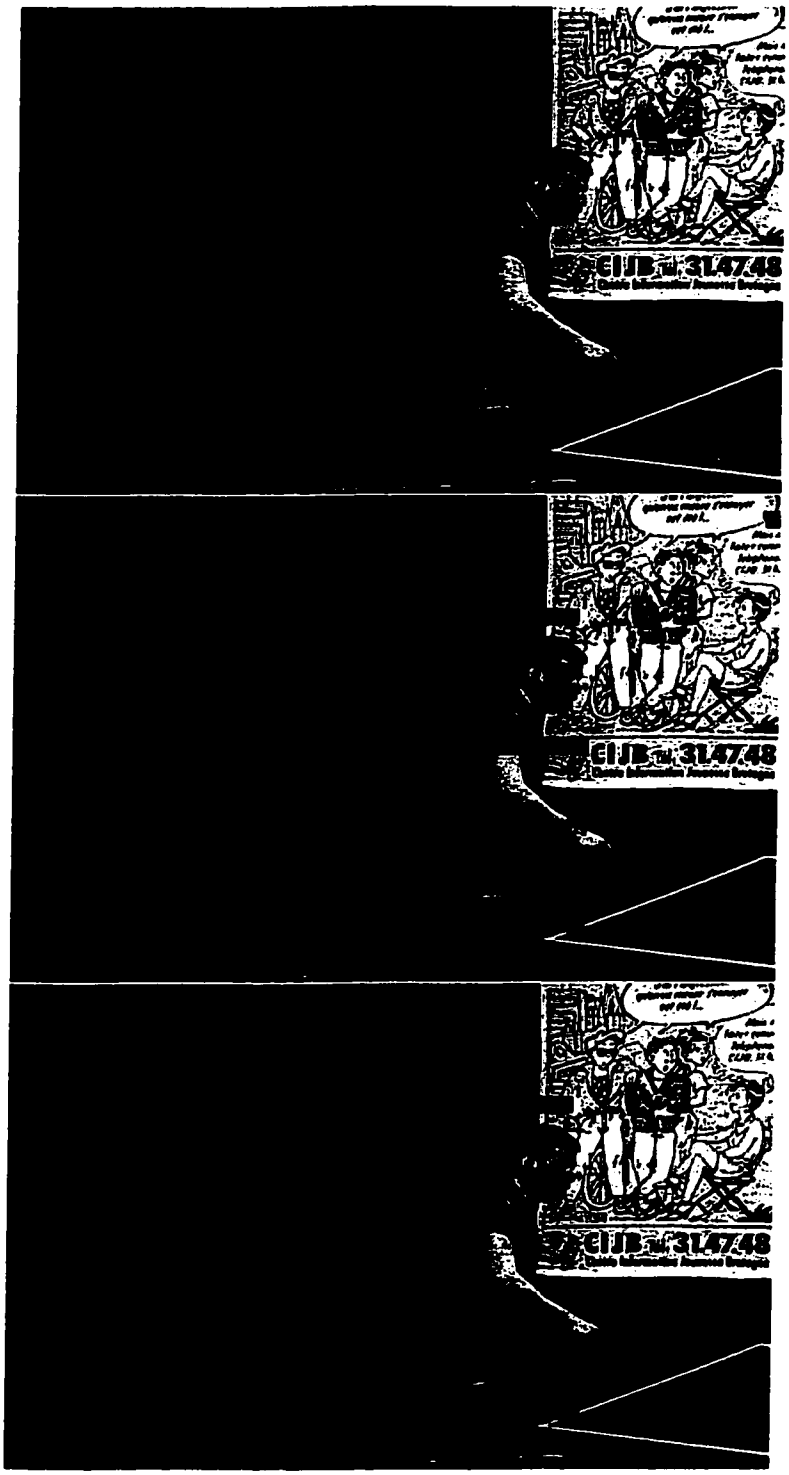

Figure 4.18: Frames 169 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with SSG0 and with channel BER of 1.2×10^{-2}, with SSG1 and with channel BER of 1.2×10^{-2}

α_1	α_2	PSNR(Y)	PSNR(Cb)	PSNR(Cr)
30	1	26.39	37.28	35.84
30	20	28.95	38.89	38.28
30	60	29.61	39.17	38.78
30	80	29.49	39.10	38.63
30	100	29.49	39.10	38.63

Table 4.2: PSNR at different α_2, at BER of 1.2×10^{-2}

SSG1 over SSG0 in the subjective picture quality can be seen. There is no black stripes and only some black blocks remained. Also some shifts and wrong DC values are left.

4.2.3 Using the slice candidates in G_0, G_1 and G_2

In this experiment, called SSG2, the MCL decoder provides slice candidates in G_0, G_1 and G_2. Based on the previous results, set $\alpha_0 = 0$, $\alpha_1 = 30$ and search for the best value for α_2.

Figure 4.19 and Table 4.2 show the simulation results in terms of PSNR versus α_2 for a channel BER of 1.2×10^{-2} and for the "Table Tennis" sequence. The results indicate the same tendency: as α_2 increases, the PSNR offered by SSG2 increases and becomes better than that of SSG1. The PSNR reaches the peak and then drops to an unchanged value as α_2 keeps increasing. However, compared to Figure 4.14, the peak (at $\alpha_2 \doteq 60$) and unchanged value (at $\alpha_2 \geq 80$) of the PSNR for large α_2 in Figure 4.19 are closer.

Figure 4.20 shows that SSG2 (with $\alpha_1 = 30$ and $\alpha_2 = 60$) provides an improvement in PSNR as compared to classic FEC, SSG0 and SSG1 for a range of channel BER from 3×10^{-3} to 1.4×10^{-2}.

As an illustrative indication of subjective quality, Figures 4.21 - 4.23 show the comparison pictures obtained by SSG2 and SSG1. SSG2 (bottom pictures) offers a better performance than SSG1 (middle pictures).

When the whole sequence is observed on the computer, the improvement in the subjective picture quality over SSG1 is also noticed. Almost all black stripes are removed and shifts are corrected. There are only checker board blocks on the

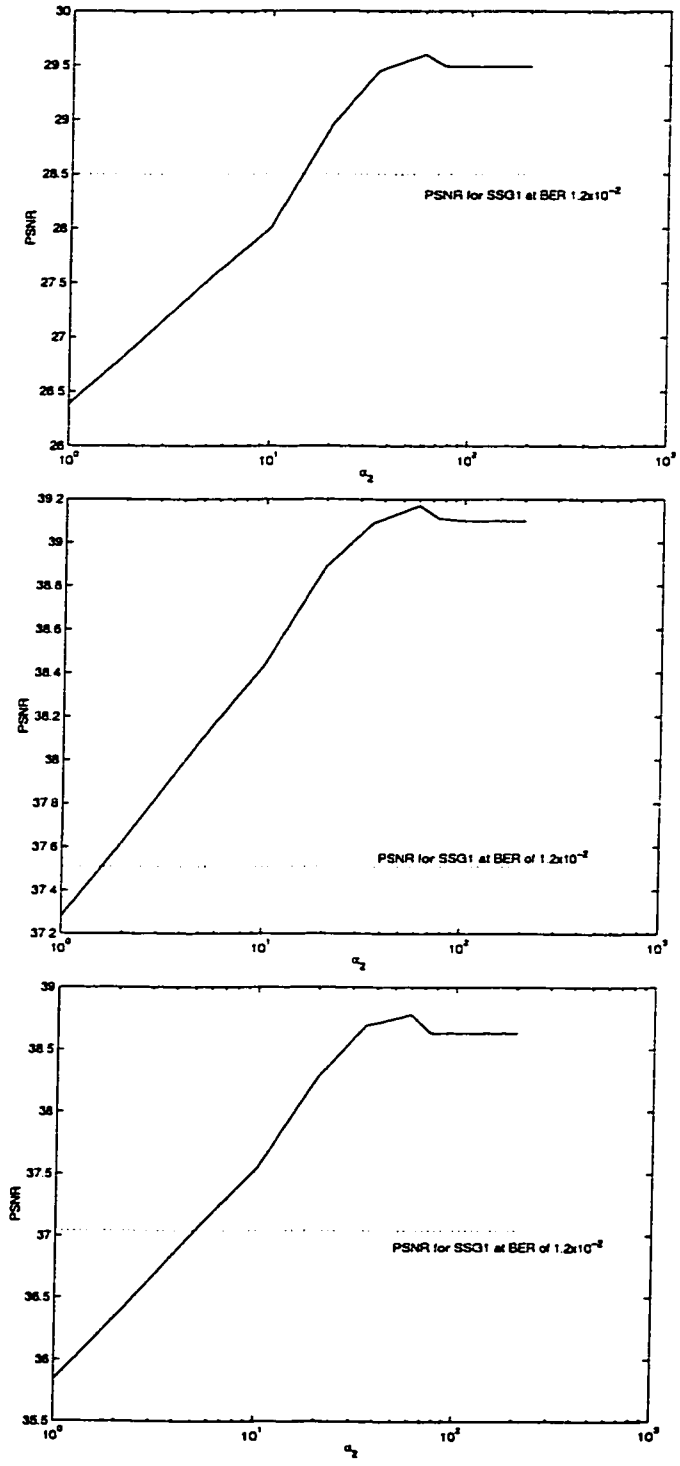

Figure 4.19: PSNR versus α_2 for "Table Tennis" (2Mb/s), at BER of 1.2×10^{-2}, from top to bottom: Luminance, Blue Chrominance, Red chrominance

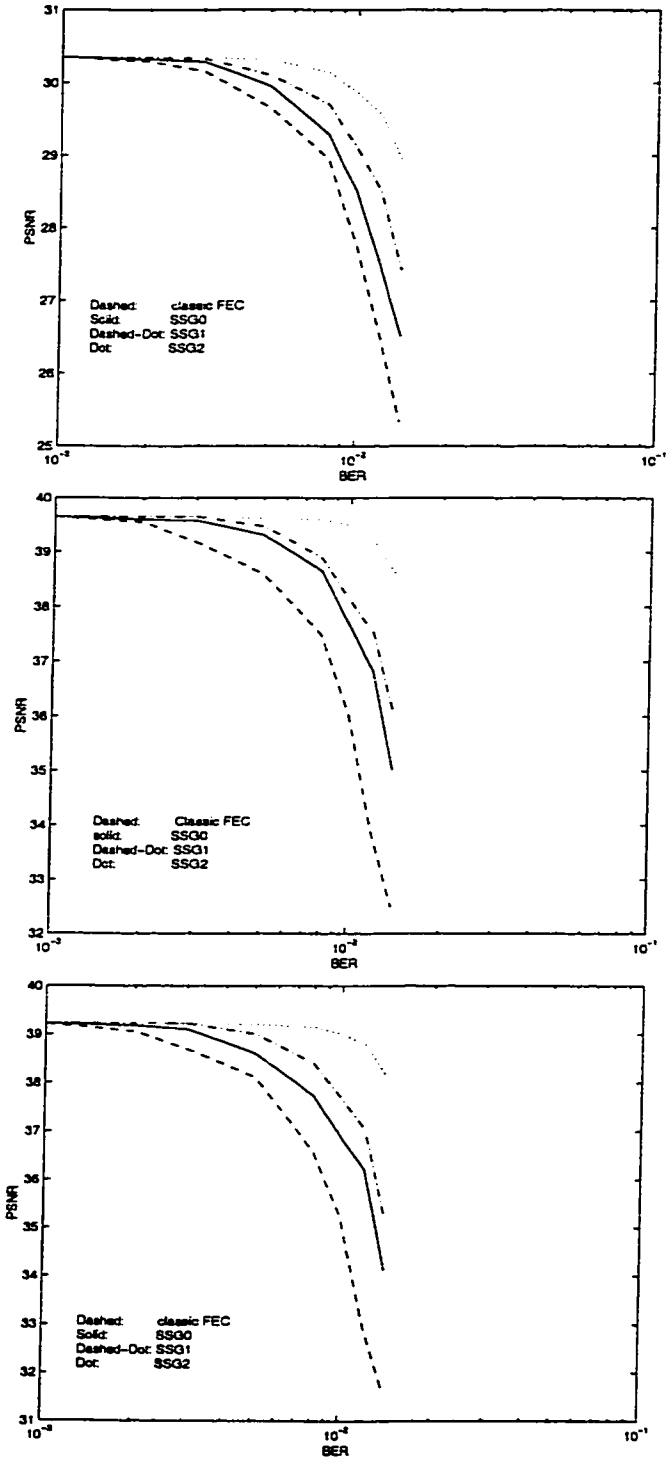

Figure 4.20: PSNR versus BER for "Table Tennis" (2Mb/s), from top to bottom: Luminance, Blue Chrominance and Red Chrominance

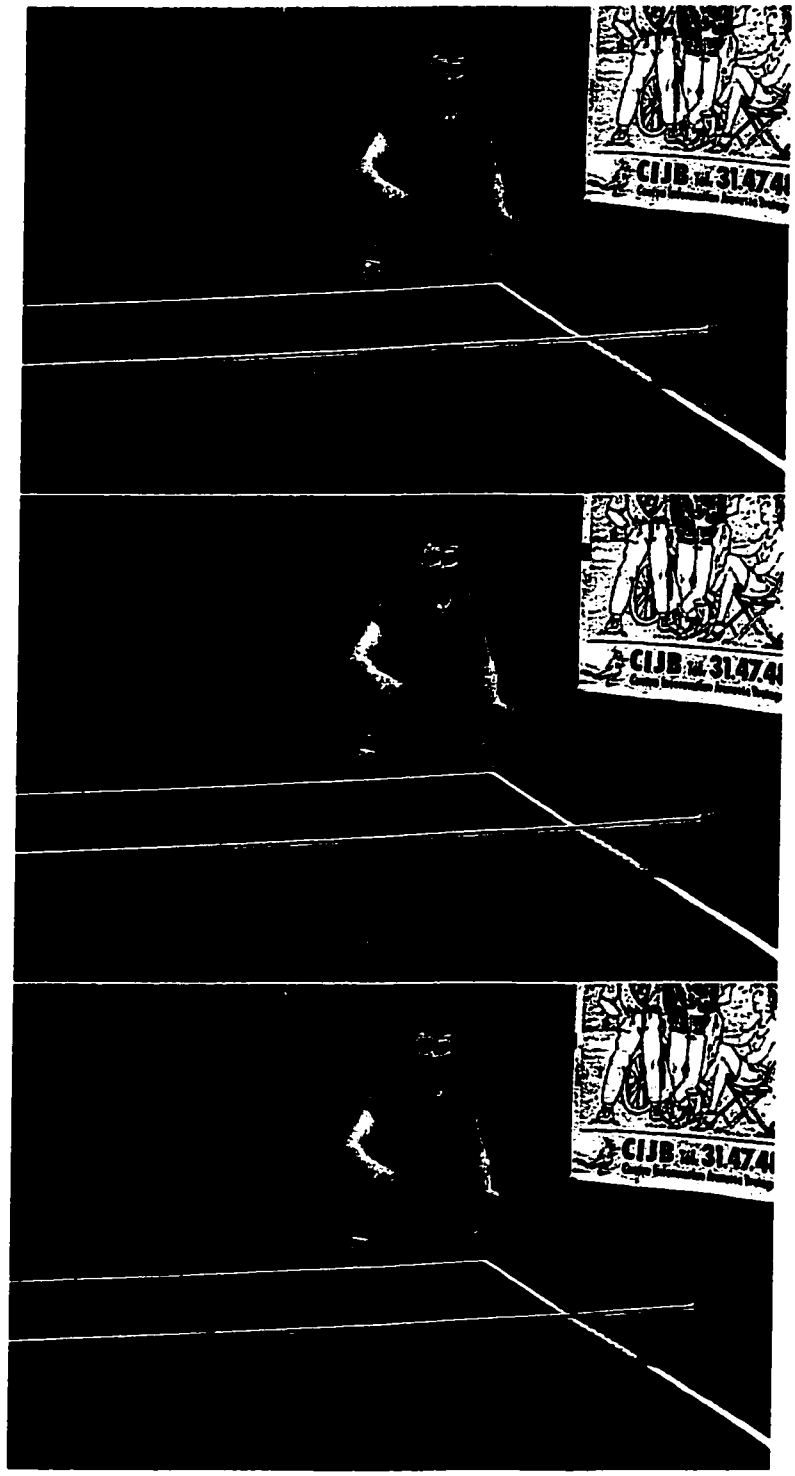

Figure 4.21: Frames 2 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with SSG1 and with BER of 1.2×10^{-2} , with SSG2 and with BER of 1.2×10^{-2}

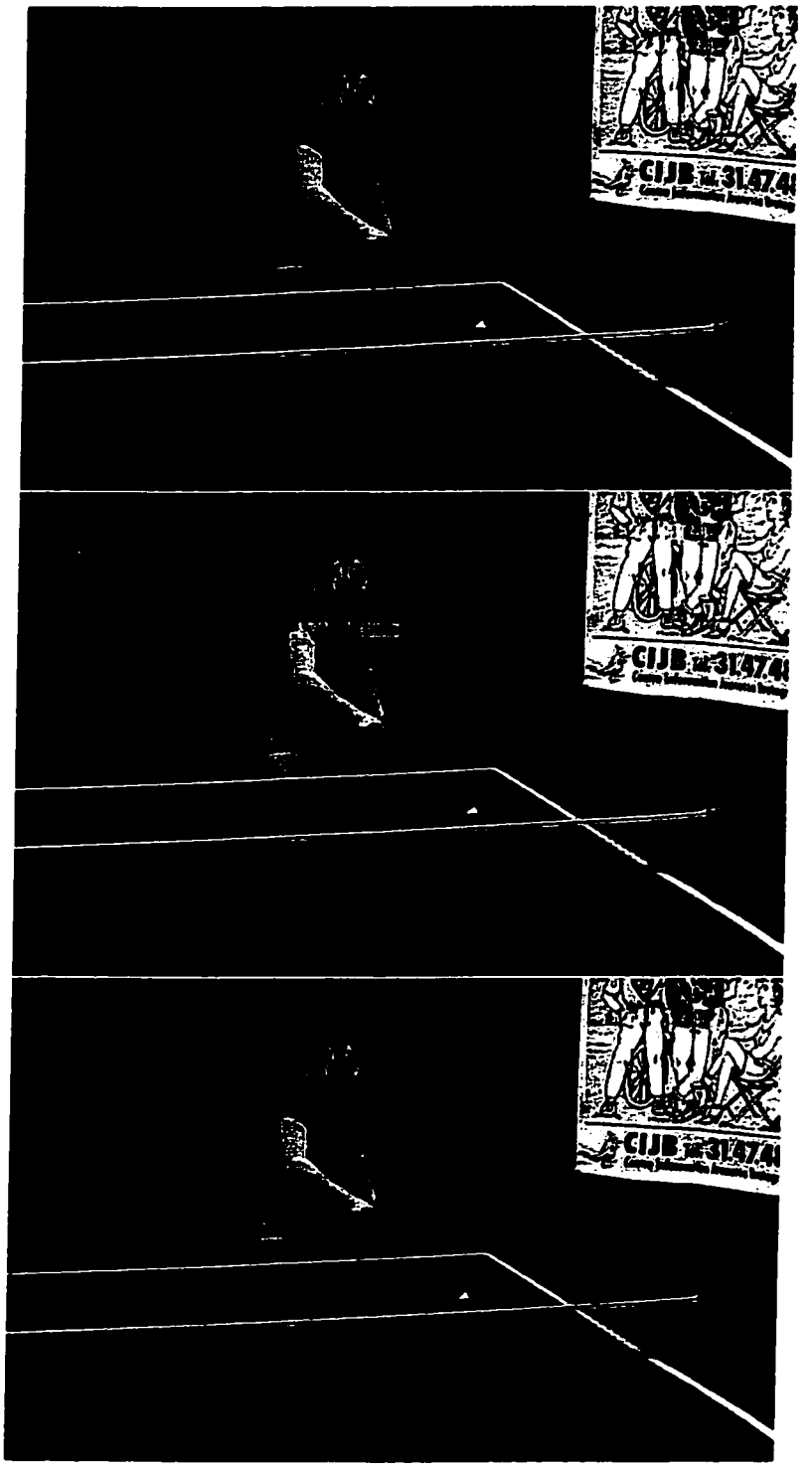

Figure 4.22: Frames 75 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with SSG1 and with BER of 1.2×10^{-2} , with SSG2 and with BER of 1.2×10^{-2}

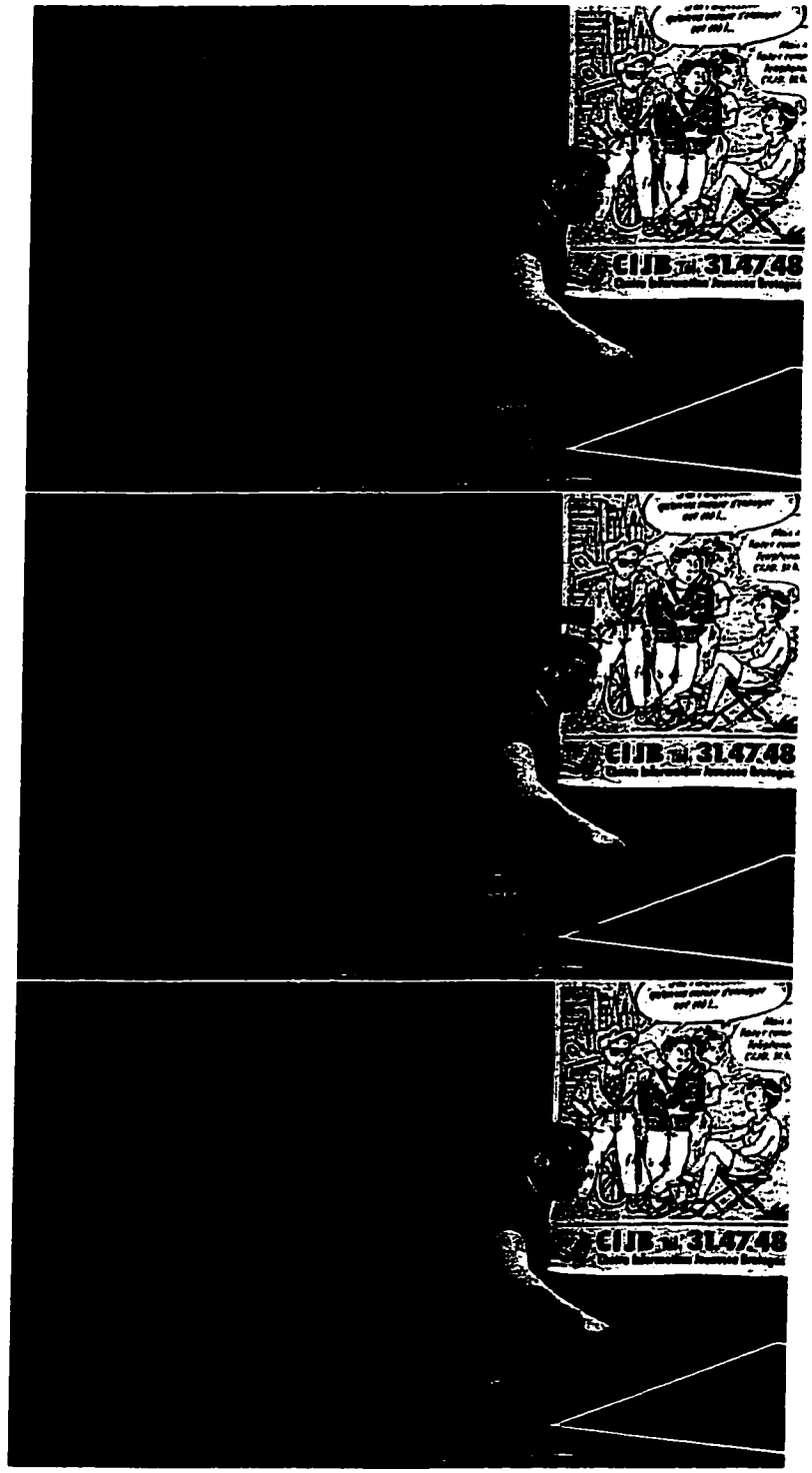

Figure 4.23: Frames 169 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with SSG1 and with BER of 1.2×10^{-2} , with SSG2 and with BER of 1.2×10^{-2}

pictures (which are not detected well by our discontinuity measure).

This experiment is also conducted with the sequence "Flower Garden". Figure 4.24 shows the PSNR versus channel BER for the cases with classic FEC and SSG2. Similar results to Figure 4.20 are obtained i.e., SSG2 provides an improvement over the classic FEC in PSNR.

Figures 4.32 - 4.27 show the sample pictures. The results again indicate the better performance of SSG2 over classic FEC.

4.3 Simulation results with a higher-rate (21,16) code

In this section, the performance of the proposed scheme is examined using a shortened BCH (21,16) code, derived from the single error correcting BCH (31,26) code. The minimum distance of this code is 3 so it can correct 1 error. The code rate is $\frac{16}{21}$ and the corresponding bandwidth expansion is $\frac{21}{16} = 1.3125$. The sequence used in the simulation is the same "Table Tennis" sequence used in the previous simulation.

4.3.1 Performance of a classical ML decoder

Figure 4.28 shows the performance comparison between the cases with and without FEC in terms of PSNR versus the channel BER for luminance, blue chrominance and red chrominance.

Without FEC, the PSNR is not degraded for the channel BER of 10^{-6} or better as previously indicated in Section 4.1. By using the (21,16) code with ML decoder, the PSNR can be kept unchanged for a channel BER as high as 3×10^{-4}. The results are not surprising since for an input channel BER of 3×10^{-4}, the (21,16) code provides a post-decoding word error probability of around 2.1×10^{-5} (equivalent to a post-decoding BER of 10^{-6}) as shown in Figure 4.29. This performance is inferior to the case using the (16,8) quasi-cyclic code (Section 4.1) because this (21,16) code has a weaker error correction capability than the (16,8) code.

Figure 4.30 shows the comparison pictures between the case with and without the (21,16) FEC code. The results indicate that with a (16,8) quasi-cyclic code and

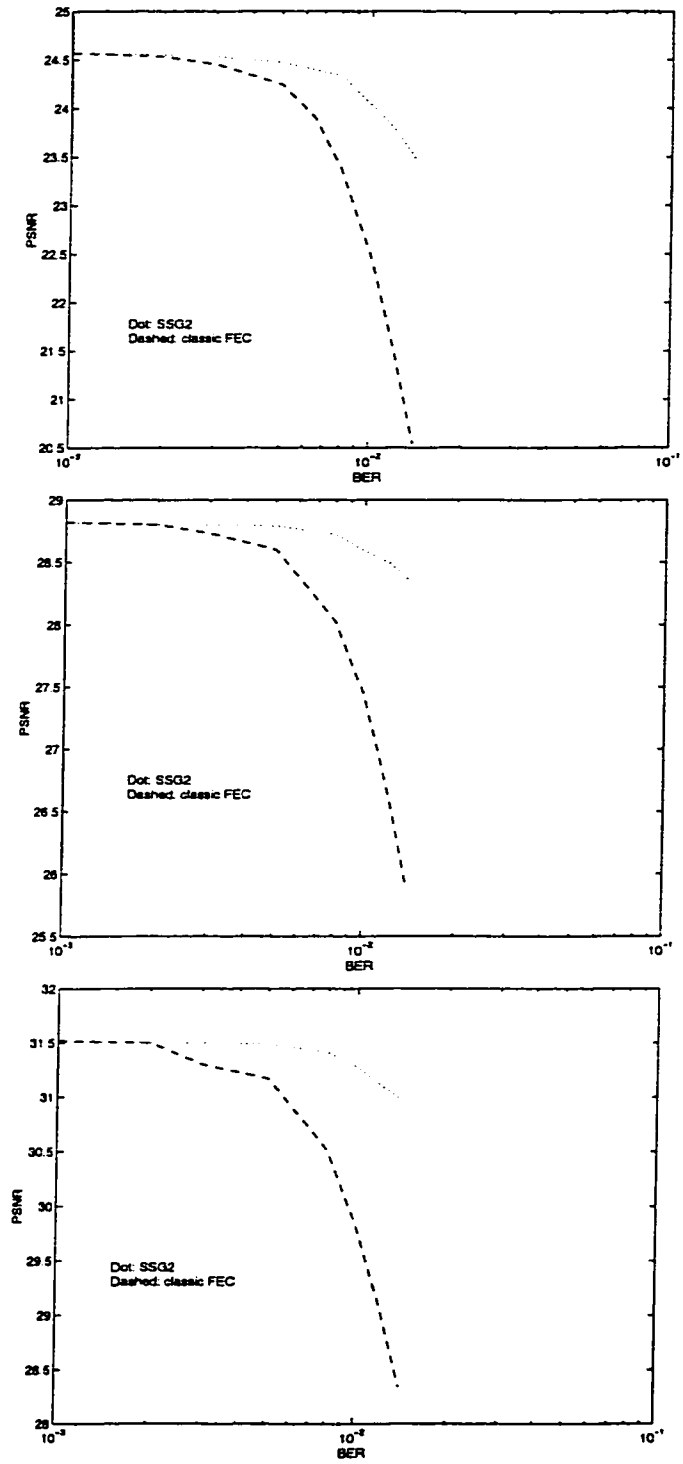

Figure 4.24: PSNR versus BER for " Flower Garden" (2Mb/s) with classic FEC and SSG2, from top to bottom: Luminance, Blue Chrominance, and Red Chrominance for " Flower Garden"

Figure 4.25: Frames 25 from compressed "Flower Garden" (2Mb/s), from top to bottom: no error, with classic FEC and with BER of 1.2×10^{-2}, with SSG2 and with BER of 1.2×10^{-2}

Figure 4.26: Frames 100 from compressed "Flower Garden" (2Mb/s), from top to bottom: no error, with classic FEC and with BER of 1.2×10^{-2}, with SSG2 and with BER of 1.2×10^{-2}

Figure 4.27: Frames 159 from compressed "Flower Garden" (2Mb/s), from top to bottom: no error, with classic FEC and with BER of 1.2×10^{-2}, with SSG2 and with BER of 1.2×10^{-2}

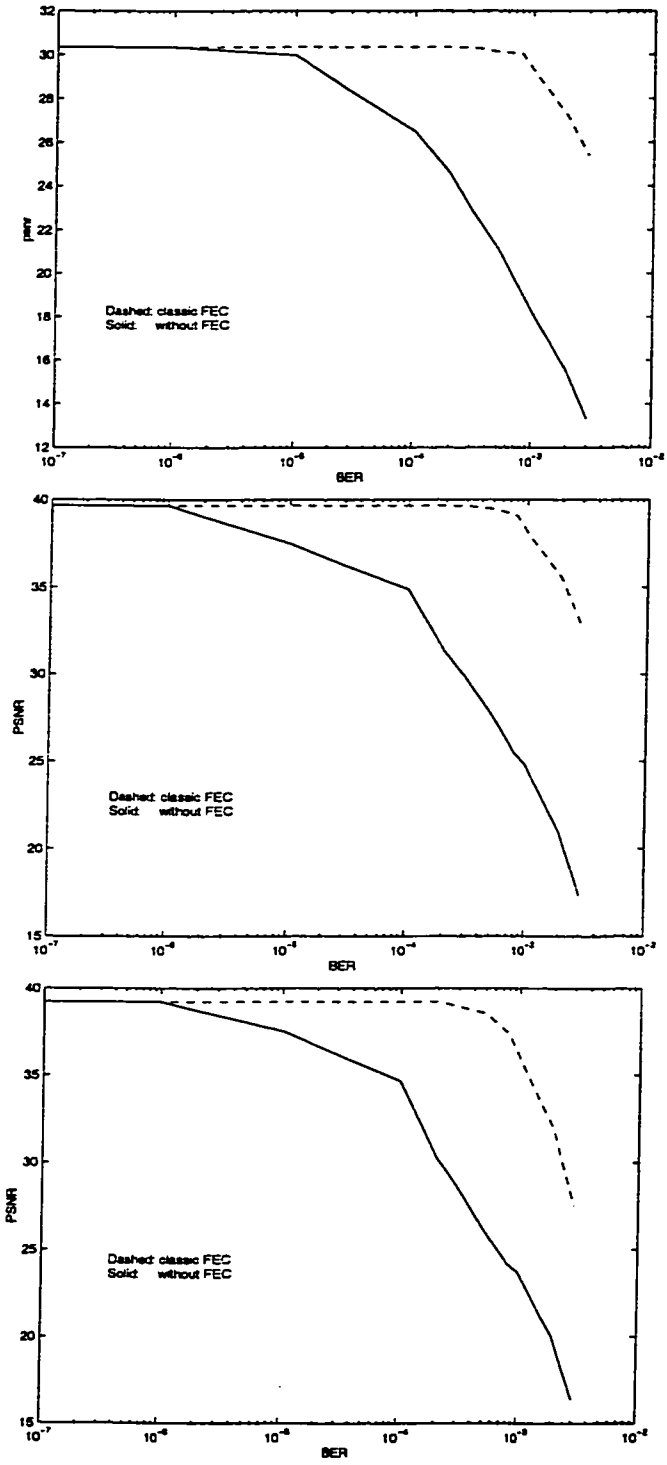

Figure 4.28: PSNR versus BER for "Table Tennis" (2Mb/s), without FEC and with (21,16) FEC code, from top to bottom: Luminance, Blue Chrominance and Red Chrominance

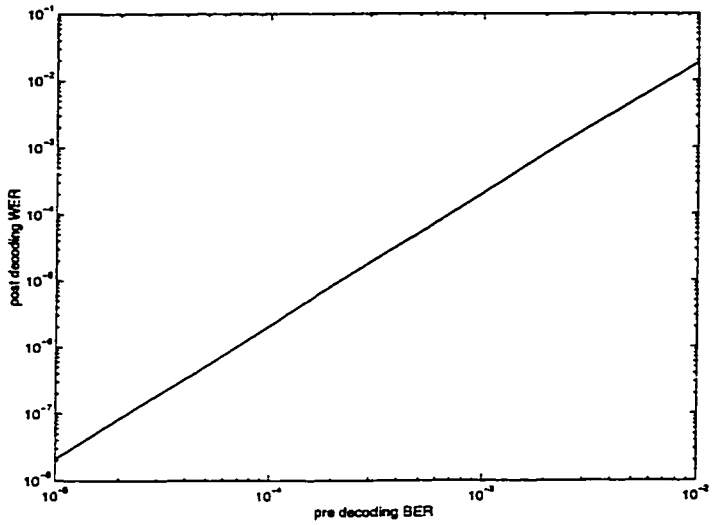

Figure 4.29: Post decoding error probability versus input channel BER for the (21,16) FEC code

a traditional ML decoder, the picture is much better than the one without FEC for the BER of 2×10^{-3}.

As evidence of subjective quality, the sequences are observed on the computer. In the corrupted sequences, when channel BER is high, the errors are serious and cause the discarding of a portion of the slice or a whole slice in many places. There are some shifts in the sequences as well. The sequences are not recognized. But the sequences after error correction are much better. Many black stripes are removed and many shifts are corrected.

4.3.2 Performance of the proposed scheme

Using $\alpha_0 = 0$, $\alpha_1 = 30$, $\alpha_2 = 60$ for the SSG2 with the shortened BCH (21,16) code, Figure 4.31 shows the simulation results in terms of PSNR versus channel BER. It indicates that SSG2 provides an improvement over the classic FEC using ML decoder in PSNR for the channel BER of 2×10^{-4} or higher.

Figure 4.32 shows the sample pictures obtained from classic FEC and SSG2. The results again indicate the better performance of SSG2 over classic FEC.

As an indication of subjective quality, the sequence after this experiment are observed on the computer. There is an improvement in the subjective picture quality

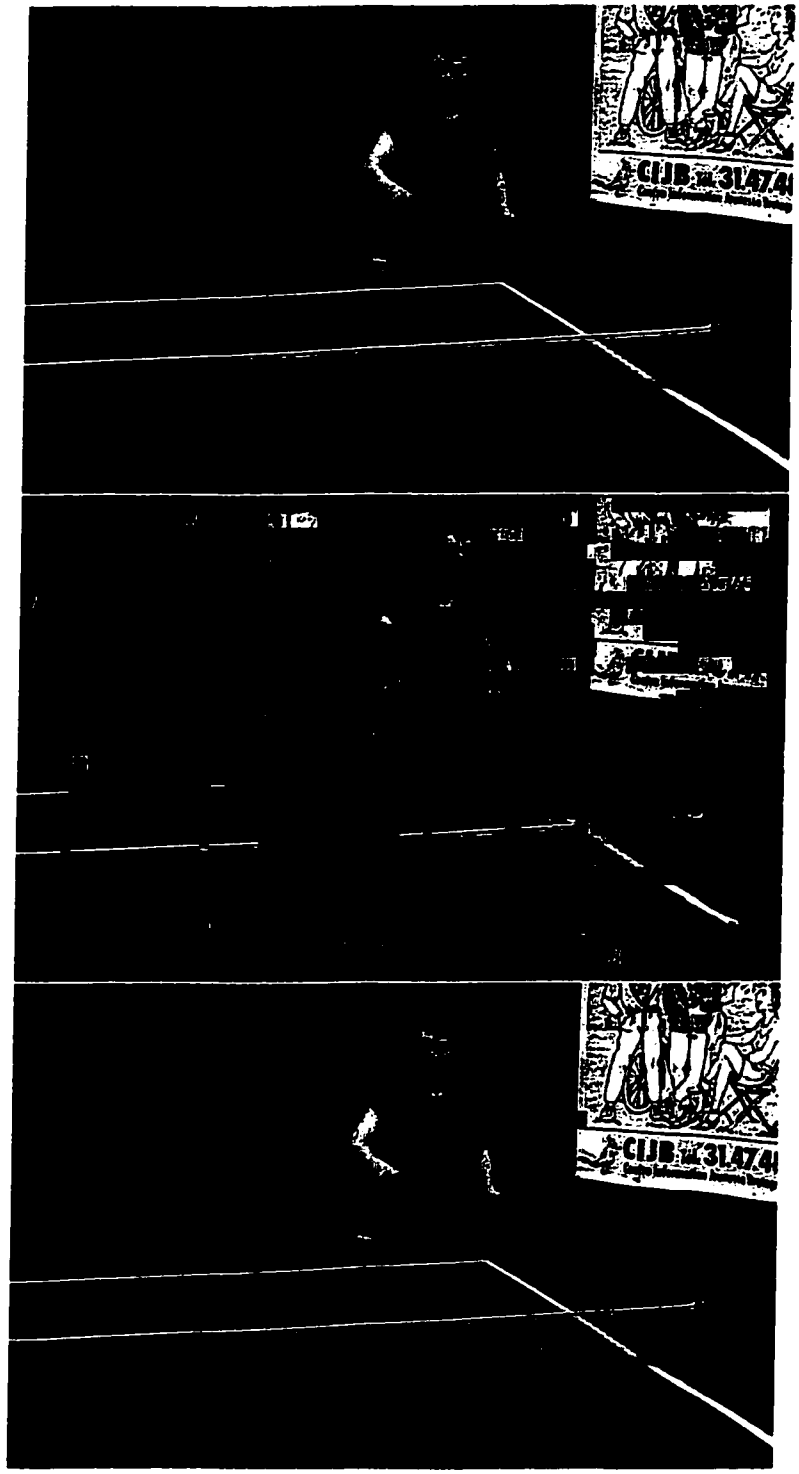

Figure 4.30: Frames 2 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, without FEC and with channel BER of 2×10^{-3}, with (21,16) FEC and with channel BER of 2×10^{-3}

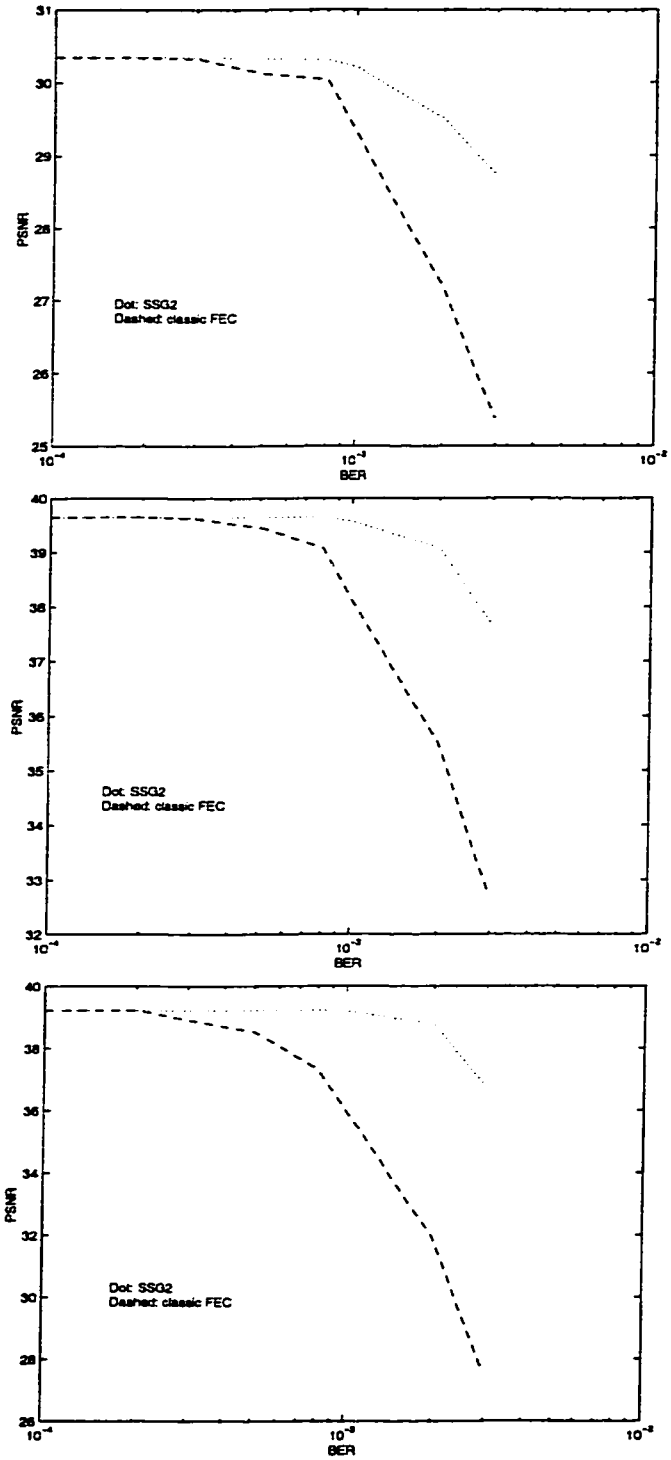

Figure 4.31: PSNR versus BER for "Table Tennis" (2Mb/s), with (21,16) FEC code and with SSG2, from top to bottom: Luminance, Blue Chrominance and Red Chrominance

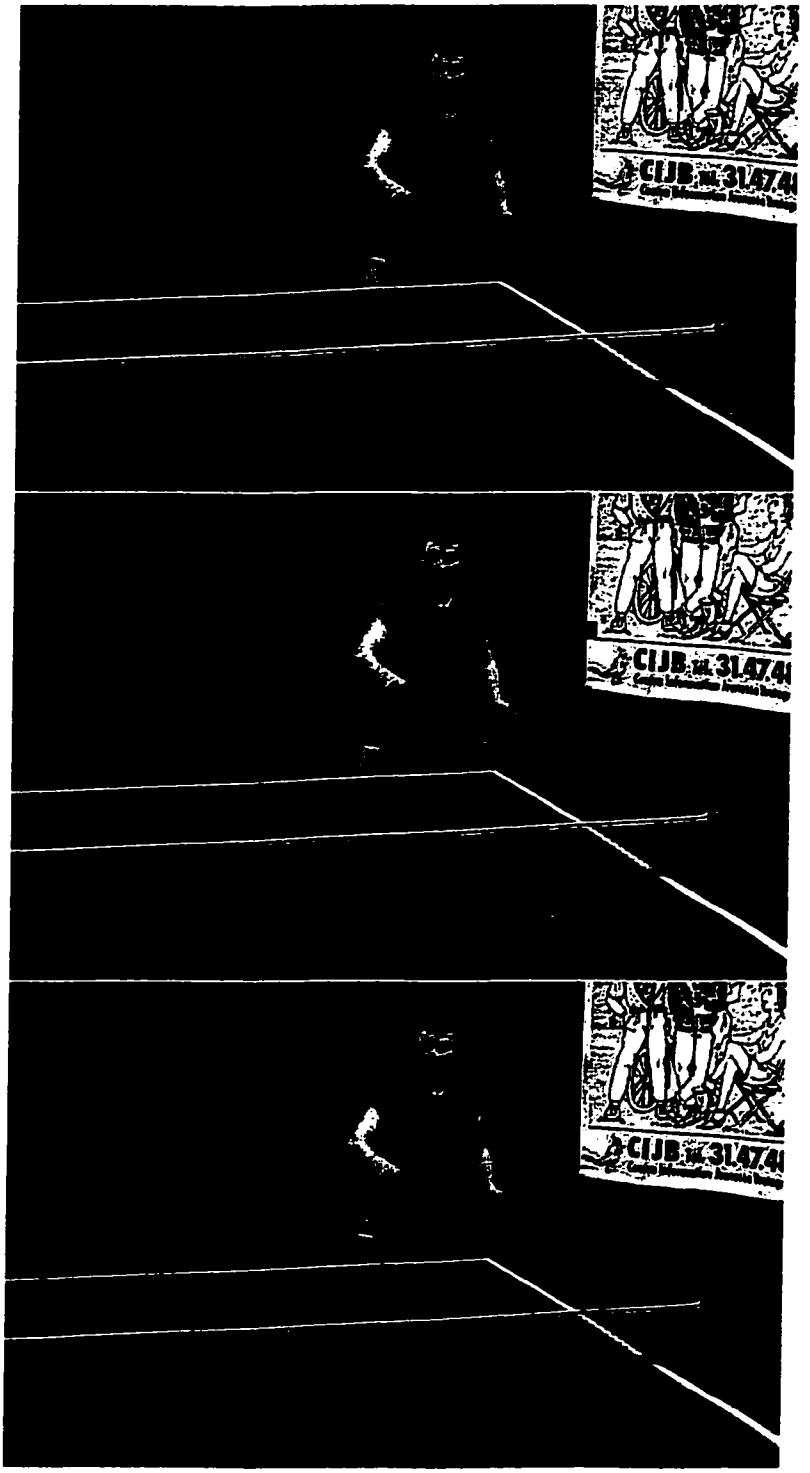

Figure 4.32: Frame 2 from compressed "Table Tennis" (2Mb/s), from top to bottom: no error, with (21,16) FEC and with BER of 2×10^{-3}, with SSG2 and with BER of 2×10^{-3}

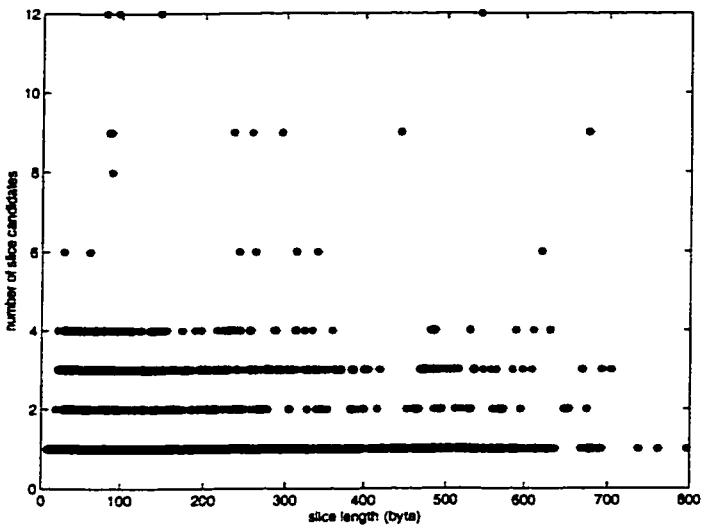

Figure 4.33: Statistical collection of numbers of slice candidates in G_0 versus slice length, at BER of 1.2×10^{-2}

over that of the classic FEC scheme. Almost all black stripes are removed and shifts are corrected. There are only some checker board blocks on the pictures.

4.4 Complexity of the proposed scheme

This thesis uses two measures for complexity: number of slice candidates to be decompressed and number of bits extracted. They are discussed in the next sections.

4.4.1 Number of slice candidates to be decompressed

Given BER of 1.2×10^{-2}, numbers of slice candidates for different slice lengths are collected and plotted in Figures 4.33 - 4.35. The sequence has a total of $120 \times 180 = 21600$ slices.

Let divide the slice length into a number of bins. Each bin has a size of 10. In each bin, the average number of slice candidates is calculated. The results based on the statistical collections in Figures 4.33 - 4.35 are plotted in Figures 4.36 - 4.38, respectively. They indicate that the average values are in a good agreement with the theoretical prediction.

The average simulation results are also compared with the confidence limits for

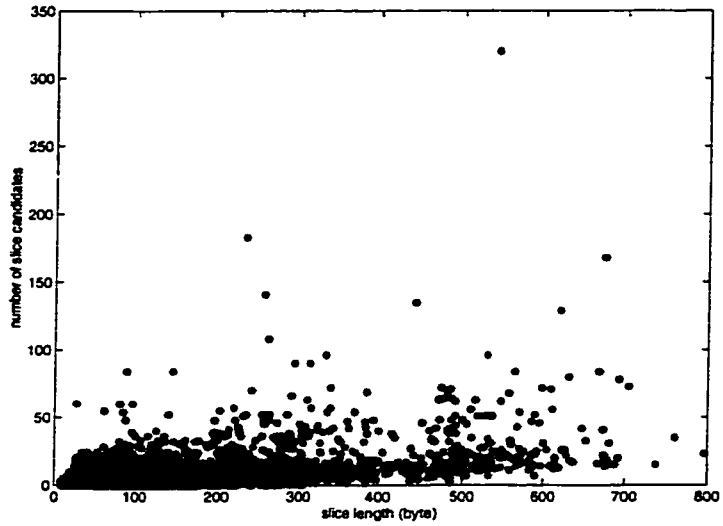

Figure 4.34: Statistical collection of numbers of slice candidates in G_1 versus slice length, at BER of 1.2×10^{-2}

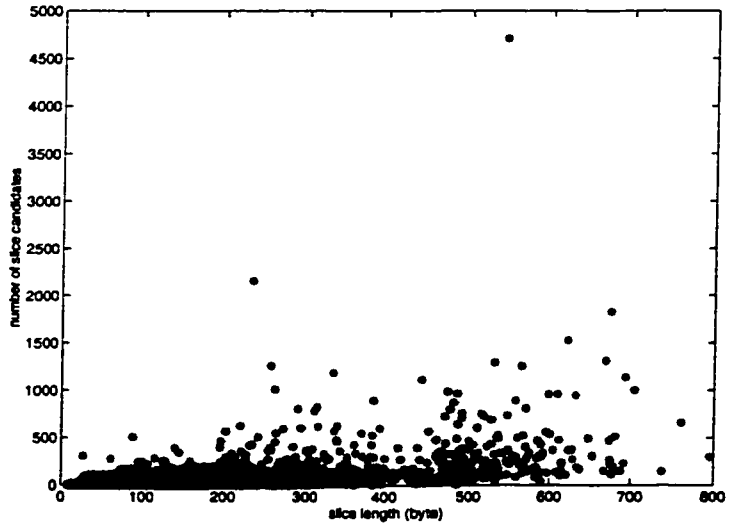

Figure 4.35: Statistical collection of numbers of slice candidates in G_2 versus slice length, at BER of 1.2×10^{-2}

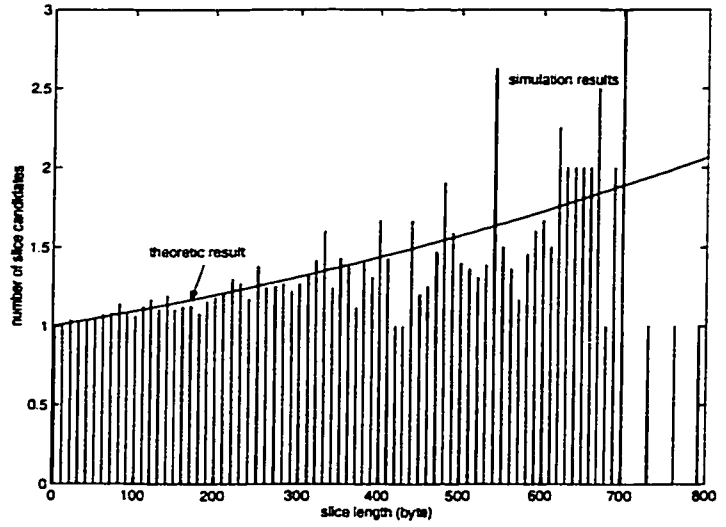

Figure 4.36: Average number of slice candidates in G_0 versus slice length

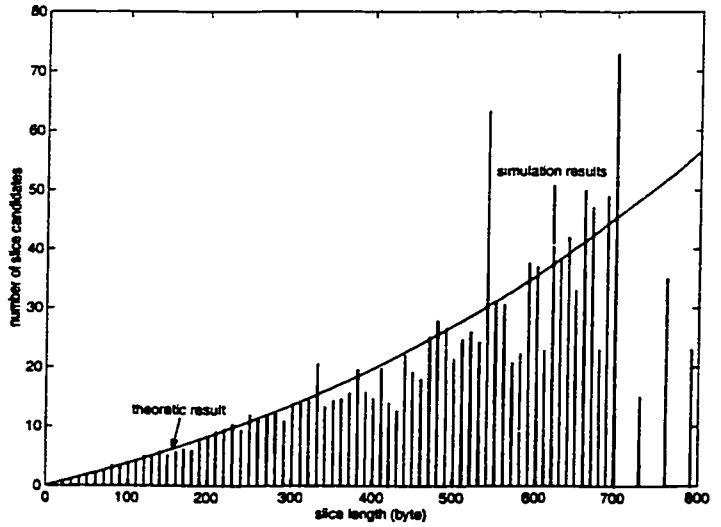

Figure 4.37: Average number of slice candidates in G_1 versus slice length

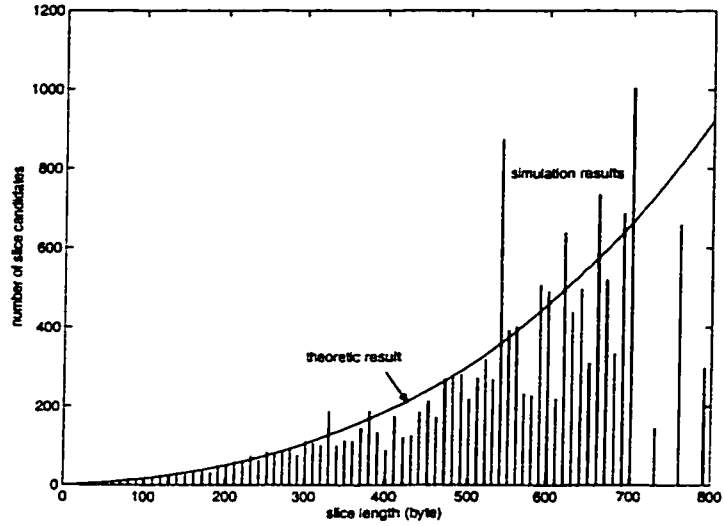

Figure 4.38: Average number of slice candidates in G_2 versus slice length

G_0 (calculated in Section 3.4.1) in Figure 4.39, 4.40.

To compare these two limits, they are plotted together in Figure 4.41.

It shows that the Chebyshev confidence limit is much looser than the central limit confidence limit.

The confidence limits for N_1 and N_2 are shown in Figure 4.42 and 4.43.

These figures indicates that the simulation results conform to the theoretic results well.

4.4.2 Number of bits extracted from the bitstream

A second proxy for complexity is number of bits extracted as explained in Section 3.4.2.

Figure 4.44 shows the number of bits extracted versus BER for different search spaces. As can be seen the cost of expanding the search space from G_0 to G_0,G_1 and then to G_0, G_1 and G_2 is very dramatic as the BER increases.

To see the effect of expanding search space on the number of bit extracted more clearly, the BER is fixed to 1.2×10^{-2}, and the numbers of bits extracted for different search space are listed in the Table 4.3.

Table 4.3 shows the increase in the bits extracted proxy for complexity as well as

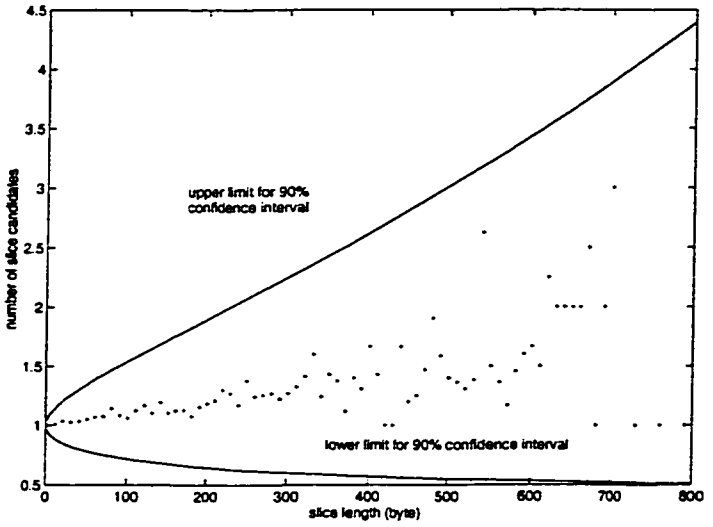

Figure 4.39: Central limit confidence limit of 90%confidence interval for N_0

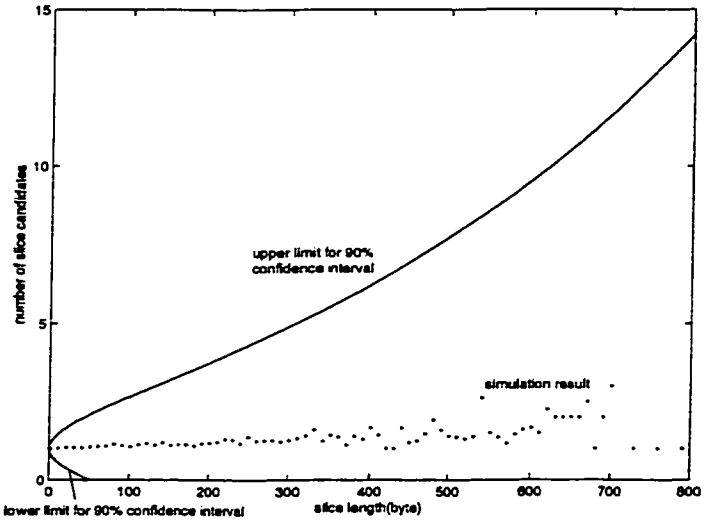

Figure 4.40: Chebyshev limit of 90%confidence interval for N_0

search space	Number of Bits Extracted	PSNR(Y)
classic FEC	24167808	26.49
SSG0	25593340	27.47
SSG1	86515424	28.50
SSG2	510091901	29.61

Table 4.3: Number of bits extracted for different search space

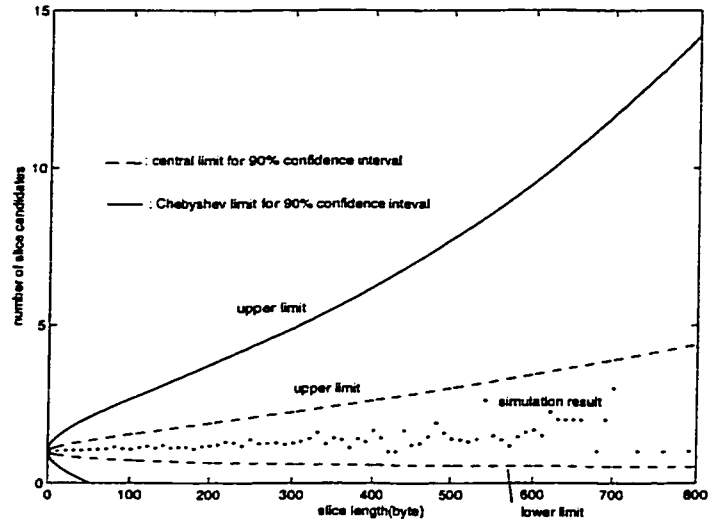

Figure 4.41: Central limit and Chebyshev limits of 90% confidence interval for N_0

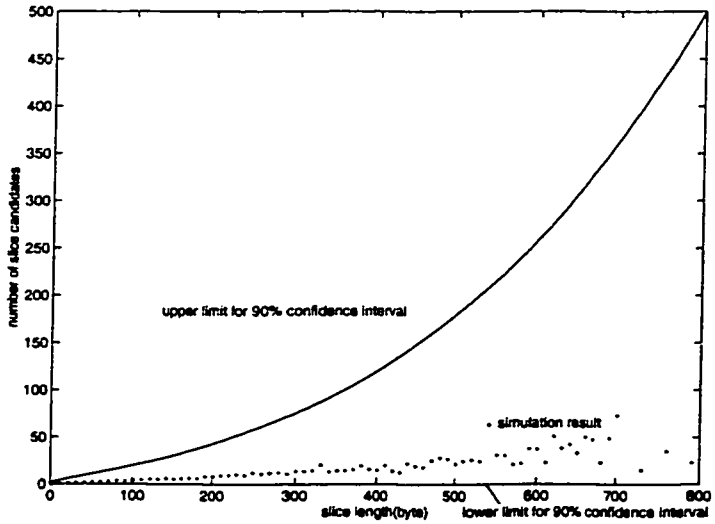

Figure 4.42: Chebyshev's limit for 90% confidence interval for N_1

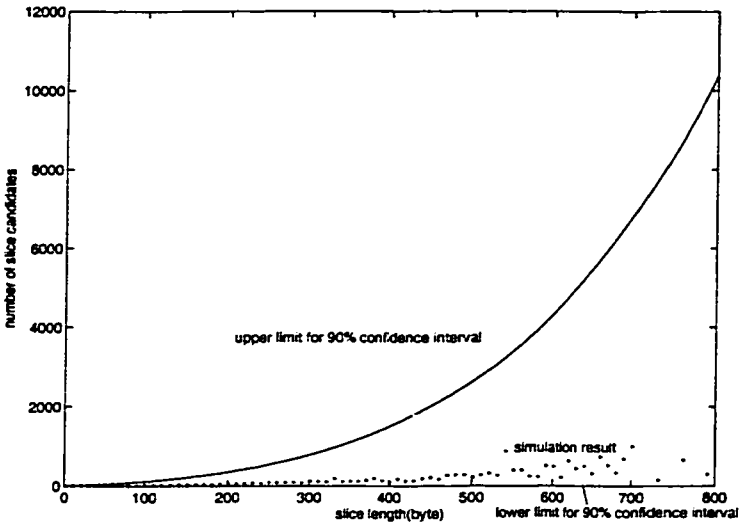

Figure 4.43: Chebyshev's limit for 90% confidence interval for N_2

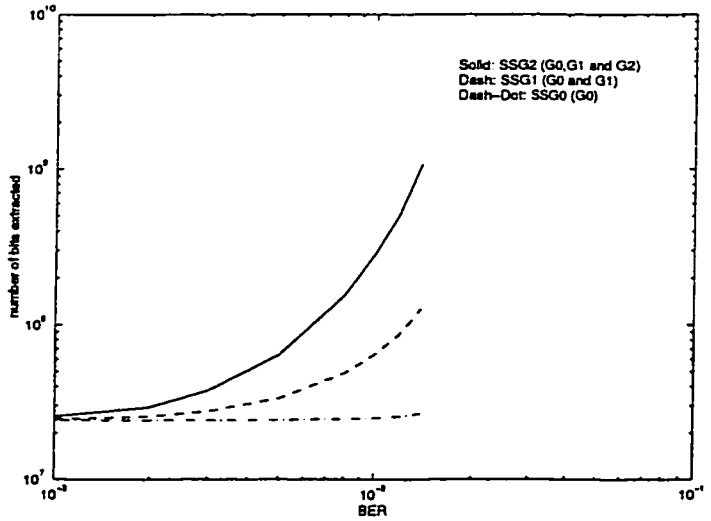

Figure 4.44: Number of bits extracted versus BER for different search space

the PSNR resulting from various search spaces. As can be seen there are diminishing returns in terms of PSNR as greater and greater costs are paid in complexity. With this in mind the search space is not further expanded beyond G_2. This decision may be less valid at higher BER's.

4.5 Further discussions

There are a number of questions that arise from the proposed technique: How big should the search space be? How do α_a and discontinuity affect the selection of slice candidates? Some discussions of these questions are presented in the following sections.

4.5.1 Value of expanding the search space

Already in Section 4.4.2 the value of expanding the search space was examined by looking at overall complexity and overall PSNR. In this section the size of the search space is examined by looking at the decisions made for individual slices. Here two questions are discussed: is it necessary to expand the search space; how big should the search space be?

To examine the decisions by the proposed scheme for an individual slice some definitions are necessary.

First, let S_{oi} be the ith slice of the original bitstream, i.e., the correct slice candidate, and let S_{ci} be slice i of the corrupted bitstream. Note that these two terms include both the information bits and the parity bits used to send a slice. Of course only the information bits carry video information.

The Hamming distance between them is the true Hamming distance d_{ti}, i.e.,

$$d_{ti} = d_H(S_{oi}, S_{ci}) \tag{4.2}$$

Then let S_{mi} be the shortest distance slice candidate for slice i. The shortest distance d_{mi} can be defined as

	BER			
	8×10^{-3}	1×10^{-2}	1.2×10^{-2}	1.4×10^{-2}
number of S_{oi}'s in G_0	21460	21320	21122	20834
number of S_{oi}'s in G_1	135	268	444	695
number of S_{oi}'s in G_2	4	9	28	58
number of S_{oi}'s in G_3	1	3	6	12
number of S_{oi}'s in G_4	0	0	0	1

Table 4.4: Number of slices whose correct slice candidate S_{oi}'s is in each group, for different BER's

	SSG0	SSG1	SSG2
number of slices choosing S_{oi}'s in G_0	20942	20946	20955
number of slices choosing S_{oi}'s in G_1	0	130	234
number of slices choosing S_{oi}'s in G_2	0	0	8

Table 4.5: Number of slices selecting slice candidates correctly, at BER 1.2×10^{-2}

$$d_{mi} = d_H(S_{ci}, S_{mi}) \tag{4.3}$$

The difference between these two distances $(d_{ti} - d_{mi})$ is the extra errors of the correct slice candidate S_{oi} over the shortest distance slice candidate S_{mi}. Then the correct slice candidate for slice i is in group $(d_{ti} - d_{mi})$, i.e., $G_{(d_{ti}-d_{mi})}$.

Out of a total of 21600 slices examined and for various BER's, Table 4.4 shows the number of slices whose correct slice candidate S_{oi} is in G_0, G_1, and G_2.

It shows that at each BER, possibly correct slice candidates S_{oi}'s lie outside of G_0. When the BER is high, one slice candidate S_{oi} even lies in G_4. This table indicates the value of expanding the search space as BER increases.

Next examine the number of slices for which the scheme correctly chooses the correct slice candidate S_{oi}. These numbers are shown in Table 4.5, at BER of 1.2×10^{-2}.

It can be seen from the table that as the search space is expanded, the number of slices that choose the correct slice candidates S_{oi}'s in each group increases too. It's interesting that the number that chooses S_{oi}'s in G_0 increases as search space increases. This is thought to be because of the fact that the discontinuity measure is done across slice boundaries: thus a better decoding of one slice can improve the

Slice	PSNR
frame 23 slice 112	38.60
frame 100 slice 71	32.38
frame 168 slice 88	32.92

Table 4.6: PSNR of slices out of search space, at BER 1×10^{-2}

Slice	PSNR
frame 13 slice 59	29.88
frame 23 slice 112	38.46
frame 100 slice 71	32.45
frame 103 slice 83	20.39
frame 118 slice 23	28.27
frame 168 slice 88	33.30

Table 4.7: PSNR of slices out of search space, at BER 1.2×10^{-2}

decoding of the slice below it.

The above discussion shows that as the search space is expanded, the video quality is improved. But as discussed in Section 4.4, the complexity increases fast as the search space is expanded. Then another question is how big should the search space be?

In Table 4.4, at the BER of 1.2×10^{-2}, the number of slices in G_2 is relatively large (28/21600). The number of slices in G_2 is even larger (58/21600) at 1.4×10^{-2}, so it's necessary to expand the search space to G_2 for these BER's.

Then let's consider if it is necessary to consider groups outside G_2.

At BER 8×10^{-3}, there is only one slice candidates in G_3 and its PSNR after SSG2 is 33.01 dB. The PSNR of the original sequence is 30.35 dB. The degradation caused by the slices outside of G_2 is not serious.

At BER 1×10^{-2}, there are 3 slice candidates in G_3. The PSNR of that slices after SSG2 is shown in Table 4.6: It can be seen from the table that the degradation is not serious for the the slices outside of the search space compared to the original PSNR 30.35 dB. Also it is not necessary to consider the slice candidates in G_3.

At BER 1.2×10^{-2}, there are 6 slice candidates in G_3. The PSNR of those slices after the simulation of SSG2 is shown in Table 4.7. It indicates that for most of the slices outside of the search space, the degradation is not serious (except for

frame 103 slice 83). The result from Section 4.2.3, the PSNR of the whole sequence is degraded less than 1 dB (0.67 dB).

The PSNR's of the slices in G_3 at different BER 8×10^{-3}, 1×10^{-2}, and 1.2×10^{-3} indicate that the slices in G_3 don't cause serious problem when BER is not higher than 1.2×10^{-2}. So it's unnecessary to expand the search space to G_3 for these BER's.

When the sequences after SSG2 (BER: 1.2×10^{-2}) are observed, which have been shown in Figure 4.21 - 4.23, it can be seen that the pictures are good and the slices in G_3 don't cause serious problem.

From the above discussion, two conclusions can be drawn:

1. It's necessary to expand the search space beyond G_0. The reconstructed video quality can be improved by expanding search space.

2. When BER $p \leq 1.2 \times 10^{-2}$, the search space does not need to be bigger than G_0, G_1 and G_3.

Simulations should be done across more sequences to see if these observations hold.

4.5.2 Evaluate discontinuity measure and α_a

The discontinuity measure and α_a play different roles in selecting slice candidates. The discontinuity measure is responsible for selection both within one group and between different groups, while α_a is only used to select slice candidates between different groups. So the selection of an incorrect slice candidate but in the correct group is the fault of discontinuity measure and not the α_a. The selection of slice candidates in an incorrect group may be due to both discontinuity measure and α_a.

The numbers of slices that select wrong groups and correct groups but wrong slice candidates at BER 1.2×10^{-2} are shown in Table 4.8.

As stated in Section 4.2.2 and 4.2.3, the α_a is selected based on PSNR. Table 4.8 indicates that as expected, the value that maximizes PSNR namely $\alpha_2 = 60$, also maximizes (by a thin margin) the number of slice candidates that are selected

α_2	correct	incorrect group	correct group but incorrect can.
35	21187	213	200
60	21197	204	199
80	21192	207	201

Table 4.8: Slices selecting wrong groups and correct groups but wrong slice candidates

correctly, and minimizes the number of slices that select a candidate in an incorrect group.

Therefore, it can be concluded that it is hard to improve the video quality further by revising α_a. But further improvement may be achieved by revising the discontinuity measure.

Chapter 5

Conclusions and Future Work

5.1 Contribution

This thesis presents a joint channel/source error resillience scheme which combines the soft outputs and reliability information given by the channel decoder together with the source information. How to make use of the information provided by the decompressor and FEC decoder is the main contribution of this work.

The traditional ML decoder does an arbitrary selection of one solution from the slice candidates with the shortest distance D^s. But with further consideration for error resilience using other measures, there may be some slice candidates that are equivalent or even worse from channel coding viewpoint but have other better measures. The traditional ML decoder prevents the chance to examine such slice candidates. Thus, an MCL channel decoding scheme is considered. Unlike the traditional ML decoder, the proposed MCL decoding strategy gives out all shortest-distance slice candidates and the shortest distance plus a ($a > 0$) slice candidates along with their reliability information. This enhances the performance because all equally good slice candidates with distance of D^s and some worse slice candidates (from the channel coding viewpoint) with distance $(D^s + a)$ are further considered using additional metrics based on the digital video signal properties.

The slice candidates given by MCL decoder are first selected using the syntax information of the digital video signal. Then the discontinuity property of the digital video is combined with the reliability information of the slice candidates to select among the candidates without syntax violation.

When the slice candidates that are not equally good from the channel coding viewpoint are considered, there is a balance between the discontinuity measure and the distance of the slice candidates. Thus an offset α_a based on the distance (D^s+a) is intentionally added to the discontinuity measure. The performance of the scheme is sensitive to the value of α_a. A best value for α_a is obtained empirically. With this value, PSNR gets the highest and it has an improvement over the traditional FEC and the scheme with a smaller search space.

5.2 Conclusions

The following conclusions are drawn:

- Objective and Subjective video quality can be improved by using residual redundancy in the video to improve classical FEC
- The worst type of degradations both subjectively and objectively is due to syntax errors
- The worst type of degradations both subjectively and objectively aside from syntax errors is due to shifts in slices and errors in the DC terms
- The interslice nature of the discontinuity measure means that better performance in one slice improves performance in slices below. This same observation can likely be made of many error concealment or resilience techniques that depend on the residual redundancy of the video.

5.3 Future work

The discontinuity measure can be improved by investigating a more systematic approach to define its formula. This can be part of the work in the future.

As BER increases the search space should be increased. However this can result in very large increases in complexity. To avoid the complexity cost unnecessarily the size of the search space can be adaptively decided by monitoring the channel BER.

Suboptimum schemes which save on complexity should be evaluated. Since most of the gain in PSNR from increasing the search space comes from slices who have no slice candidates that are syntax error free in the smaller search space, then one scheme would be to only search group G_i if there are no candidates without syntax errors in slice G_j where $j < i$.

Soft FEC decoding can give more than one slice candidates and can give reliability information for each slice candidate. Thus it is suitable to be used in this

scheme. In the future, soft decoding such as Viterbi decoding can be applied into the proposed scheme.

The proposed scheme just uses the forward error correcting scheme. There is no feedback and iteration in the scheme. In the future, one can use the idea of Turbo coding that corrects the errors iteratively.

There are some other jobs that can be left as the future work such as considering of burst errors and using different compressed scheme such as MPEG4.

Bibliography

[1] J.L.Mitchell, W.B.Pennebaker, C.E.Fogg, and D.J.Legall, *MPEG Video Compression Standard.* International Thomson Publishing, 1996.

[2] S. M. E. Group, *Generic Coding of Moving Pictures and Associated Audio Information: Video.* ISO-IEC/JTC1/SC29/WG11 N0702, May 1994.

[3] T.Sikora, "MPEG Digital Video Coding Standards," *IEEE Signal Processing Magazine*, pp. 82–100, September 1997.

[4] W.Luo and M.E.Zarki, "Analysis of Error Concealment Schemes for MPEG2 Video Transmission over ATM based Networks," *SPIE Video Coding and Image Processing*, vol. 2501, pp. 1358–1368, 1995.

[5] F.Jeng and S.Lee, "Concealment of Bit Error and Cell Loss in Inter Frame Video Transmission," in *IEEE International Conference on Communications*, pp. 496–500, 1991.

[6] Y.Wang and Q.F.Zhu, "Error control and concealment for video communication: A review," in *Proceedings of the IEEE*, vol. 86, pp. 974–996, May. 1998.

[7] A.K.Jain, *Fundamentals of Digital Image Processing.* Prentice Hall, 1989.

[8] A.N.Netravali and B.G.Haskell, *Digital Pictures Representation Compression and Standards.* Plenum Press, 1995.

[9] VQEG, *Draft New Recommendation J.144: Preceptual video quality measurement techniques for digital cable television in the presence of a full reference.*

[10] A.B.Watson, *Digital Images and Human Vision.* MIT Press, 1993.

[11] J.G.Proakis, *Digital Communications.* McGraw-Hill, 1995.

[12] M. H. Page, "http://www.cselt.it/mpeg."

[13] B.G.Haskell, A.Puri, and A.N.Netravali, *Digital Video: An Introduction to MPEG-2.* International Thomson Publishing, 1996.

[14] S. Lin and J. C. Jr., *Error Control Coding.* Prentice Hall, 1983.

[15] J.W.Park, J.W.Kim, and S.U.Lee, "Dct coefficients receovery-based error concealment technique and its application to the mpeg-2 bit stream error," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 845–854, December 1997.

[16] W.J.Chu and J.J.Leou, "Detection and Concealment of Transmission Errors in h.263 Images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 74–84, February 1998.

[17] W.Kwok and H.Sun, "Multi Directional Interpolation for Spatial Error Concealment," *IEEE Transactions on Consumer Electronics*, vol. 39, pp. 455–460, August 1993.

[18] K.L.Hung, C.C.Chang, and T.S.Chen, "Reconstruction of Lost Blocks Using Codeword Estimation," *IEEE Transactions on Consumer Electronics*, vol. 45, pp. 1190–1197, November 1999.

[19] Y. Wang, M. Orchard, and A. Reibman, "Multiple description image coding for noisy channels," in *Porceedings IEEE 1997 First Workshop on Multimedia Signal Processing*, 1997.

[20] Y.Wang, S.Wenger, J.Wen, and A.K.Katsaggelo, "Error resilient video coding techniques," *IEEE Signal Processing Magazine*, pp. 61–82, July 2000.

[21] Z.Peng, Y.F.Huang, D.J.Costello, and R.L.Stevenson, "Joint Channel and Source Decoding for Vector Quantized Images Using Turbo Codes," in *ISCAS-98*, June 1998.

[22] L.J.Wang and J.J.Leou, "Detection and Correction of Transmission Errors in Pyramid Images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, pp. 25–30, February 1998.

[23] P. Burlina and F. Alajaji, "An error resilient scheme for image transmission over noisy channels with memory," *IEEE Transactions on Image Processing*, vol. 4, pp. 593–600, April 1998.

[24] I.-T. R. H.261, *Video Coding for Audiovisual Services at 64p kbits/s*, 1990.

[25] I.-T. R. H.263, *Video Coding for Low Bitrate Communication*, 1996.

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

[27] E.Steinbach, N.Garber, and B.Girod, "Standard compatiable extension of h.263 for robust video transmission in mobile envionments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 872–881, December 1997.

[28] C.S.Kim, R.C.Kim, and S.U.Lee, "An Error Detection and Recovery Algorithm for Compressed Video Signal Using Source Level Redundancy," *IEEE Transactions on Image Processing*, vol. 9, pp. 209–219, February 2000.

[29] V.Papadakis, T.Le-Ngoc, and W.E.Lynch, "Combined Error Detection and Error Concealment for MPEG2 Over ATM," in *CCECE-97*, vol. I, pp. 137–140, May 1997.

[30] C.T.Chen, "Error detection and concealment with an unsupervised mpeg2 video decoder," *Journal of Visual Communication and Image Representation*, vol. 6, pp. 265–279, September 1995.

[31] W.E.Lynch, V.Papadakis, R.Krishnamurthy, and T.Le-Ngoc, "Syntax Based Error Concealment," in *ISCAS-99*, May 1999.

[32] V.Papadakis, W.E.Lynch, and T.Le-Ngoc, "Syntax Based Error Concealment," in *ISCAS-98*, pp. TPA13–9, June 1998.

[33] R.E.Walpole, R.H.Myers, and S.L.Myers, *Probability and Statistics for Engeers and Scientists*. Prentice Hall, 1989.

Appendix A

(16,8) quasi-cyclic code

The generator matrix of the systematic (16,8) quasi-cyclic code is given as

$$G = [I, P] \tag{A.1}$$

where I is an 8×8 identity matrix and P is an 8×8 binary circulant matrix of the form

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{A.2}$$

The valid codewords for this FEC code are given as follows:

0 279 558 825 1116 1355 1650 1893 2232 2479 2710 2945 3300 3571 3786 4061 4209 4454
4703 4936 5165 5434 5635 5908 6345 6622 6887 7152 7317 7554 7867 8108 8418 8693 8908 9179
9406 9641 9872 10119 10330 10573 10868 11107 11270 11537 11816 12095 12435 12676 12989 13226
13519 13784 14049 14326 14379 14652 14853 15122 15479 15712 15961 16206 16581 16850 17131
17404 17561 17806 18103 18336 18557 18794 19027 19268 19489 19766 19983 20248 20660 20899
21146 21389 21736 22015 22214 22481 22540 22811 23074 23349 23632 23879 24190 24425 24615
24880 25097 25374 25723 25964 26197 26434 26783 27016 27313 27558 27843 28116 28397 28666
28758 28993 29304 29551 29706 29981 30244 30515 30958 31225 31424 31703 31922 32165 32412
32651 32907 33180 33445 33714 34007 34240 34553 34798 34867 35108 35357 35594 35951 36216
36417 36694 37114 37357 37588 37827 38054 38321 38536 38815 38978 39253 39532 39803 39966
40201 40496 40743 41065 41342 41543 41808 42037 42274 42523 42764 43217 43462 43775 44008
44173 44442 44707 44980 45080 45327 45622 45857 46148 46419 46698 46973 47264 47543 47758
48025 48380 48619 48850 49093 49230 49497 49760 50039 50194 50437 50748 50987 51446 51681
51928 52175 52394 52669 52868 53139 53311 53544 53777 54022 54371 54644 54861 55130 55431
55696 55977 56254 56539 56780 57077 57314 57516 57787 57986 58261 58608 58855 59102 59337
59412 59651 59962 60205 60488 60767 61030 61297 61661 61898 62195 62436 62593 62870 63151
63416 63589 63858 64075 64348 64569 64814 65047 65280

Appendix B

Matlab source code to derive the distance profile for the (16,8) quasi-cyclic code

Let val contain the valid codewords. Codewords are 16 bits long (each array entry is "0" or "1".) and there are 256 valid codewords. Generate all 2^{16} received words in the "1", "0" format:

```
i=1;
allword=zeros(65536,16);
for j15=0:1
 for j14=0:1
  for j13=0:1
   for j12=0:1
    for j11=0:1
     for j10=0:1
      for j9=0:1
       for j8=0:1
        for j7=0:1
         for j6=0:1
```

```
for j5=0:1
 for j4=0:1
  for j3=0:1
   for j2=0:1
    for j1=0:1
     for j0=0:1
      allword(i,:)=[j15 j14 j13 j12 j11 j10 j9 j8 j7 j6 j5 j4 j3 j2 j1 j0];
      i=i+1;
     end
    end
   end
  end
 end
end
```

For the received words that are 0 bit away from the closest valid codeword (Class 0 in Table 3.1), its distance to each valid codeword is calculated.

```
dist0=zeros(256,17);
for i=1:256
 for j=1:256
```

```
 dist0(i,sum(xor(val(i,:),val(j,:)))+1)=dist0(i,sum(xor(val(i,:),val(j,:)))+1)+1 ;
 end
end
```

There are only one possible distance profiles for the 256 received words in Class 0. This can be tested as follows:

```
for i=1:256
 hold = sum(abs(dist0(1,:)-dist0(i,:)));
 if hold ~= 0
  i
 end
end
```

Nothing returns. So there are only one possible distance profiles for the 256 received words in Class 0.

For each of the possible received words, calculate how far it is to each each valid codeword.

```
distance=zeros(65536,17);
for i=1:65536
 for j=1:256
  d = sum(xor(val(j,:),allword(i,:)));
  distance(i,d+1) = distance(i,d+1) + 1;
 end
end
```

All the received words that are 1 bit away from the closest codeword (Class 1 in Table 3.1) are sort out. There should be $\begin{pmatrix} 16 \\ 1 \end{pmatrix} = 16$ for each valid codeword, thus 16*256 = 4096 of these.

```
j=1;
dist1=zeros(4096,18);
for i=1:65536
```

```
 if distance(i,1) == 0
  if distance(i,2) == 1
   dist1(j,1)=i;
   dist1(j,2:18)=distance(i,1:17);
   j=j+1;
  end
 end
end
```

The possible types of distance profile for Class 1 is obtained as follows:

```
dist1prof=[1 dist1(1,2:18)];
i=1;
for j=2:4096
 flag1 = 0;
 for k=1:i
  flag2 = 0;
  for l=2:18
   if dist1(j,l) ~= dist1prof(k,l)
    flag2 = 1;
   end
  end
  if flag2 == 0
   flag1 = 1;
   dist1prof(k,1) = dist1prof(k,1) + 1;
  end
end
if flag1 == 0
 i=i+1;
 dist1prof(i,1) = 1;
 dist1prof(i,2:18) = dist1(j,2:18);
```

```
 end
end
```

There are only two possible distance profiles for the 4096 received words in Class 1. They are split half/half between the two profiles. (Note number in first column is number of received words with this profile, subsequent columns are the distance profile. Column 2 is the number of valid code words 0 bit away from the received words with this profile, column 3 is the number of valid codewords that are 1 bit away, etc).

Now for the received words in Class 2, which are 2 bits away from the closest valid codewords, calculate how far it is to each valid codeword. Note that there should be $\binom{16}{2} = 120$ for each valid codeword, so there are 120*256 = 30720 of these.

```
j=1;
dist2=zeros(30720,18);
for i=1:65536
 if distance(i,1) == 0
  if distance(i,2) == 0
   if distance(i,3) == 1
    dist2(j,1)=i;
    dist2(j,2:18)=distance(i,1:17);
    j=j+1;
   end
  end
 end
end
```

The possible types of distance profile for Class 2 is obtained as follows:

```
dist2prof=[1 dist2(1,2:18)];
i=1;
```

```
for j=2:30720
 flag1 = 0;
 for k=1:i
  flag2 = 0;
  for l=2:18
   if dist2(j,l) ~= dist2prof(k,l)
    flag2 = 1;
   end
  end
  if flag2 == 0
   flag1 = 1;
   dist2prof(k,1) = dist2prof(k,1) + 1;
  end
 end
 if flag1 == 0
  i=i+1;
  dist2prof(i,1) = 1;
  dist2prof(i,2:18) = dist2(j,2:18);
 end
end
```

Of the received words in Class 2 there are 8 types.

Now to those received codewords in Class 3. Note here a received word may be 3 bits away from 2 or more valid codewords. So the code is changed. Also how many received words in Class 3 can no longer be calculated before hand since the correction capabilities of the code is exceeded.

```
j=1;
dist3=zeros(31000,18); (the number is not know so we guess high)
for i=1:65536
 if distance(i,1) == 0
```

```
  if distance(i,2) == 0
   if distance(i,3) == 0
    if distance(i,4) ~= 0
     dist3(j,1)=i;
     dist3(j,2:18)=distance(i,1:17);
     j=j+1;
    end
   end
  end
 end
end
```

The possible distance profiles for received words in Class 3 is obtained as follows:

```
dist3prof=[1 dist3(1,2:18)];
i=1;
for j=2:30208
 flag1 = 0;
 for k=1:i
  flag2 = 0;
  for l=2:18
   if dist3(j,l) ~= dist3prof(k,l)
    flag2 = 1;
   end
  end
  if flag2 == 0
   flag1 = 1;
   dist3prof(k,1) = dist3prof(k,1) + 1;
  end
end
if flag1 == 0
```

```
  i=i+1;
  dist3prof(i,1) = 1;
  dist3prof(i,2:18) = dist3(j,2:18);
 end
end
```

There are 10 distinct types in Class 3. For those received words in Class 4, calculate how far it is to each valid codeword.

```
j=1;
dist4=zeros(356,18); (the number is not known so be guessed high)
for i=1:65536
 if distance(i,1) == 0
  if distance(i,2) == 0
   if distance(i,3) == 0
    if distance(i,4) == 0
     if distance(i,5) ~= 0
      dist4(j,1)=i;
      dist4(j,2:18)=distance(i,1:17);
      j=j+1;
     end
    end
   end
  end
 end
end
```

The possible distance profile for Class 4 is obtained as follows:

```
dist4prof=[1 dist4(1,2:18)];
i=1;
for j=2:256
 flag1 = 0;
```

```
 for k=1:i
  flag2 = 0;
  for l=2:18
   if dist4(j,l) ~= dist4prof(k,l)
    flag2 = 1;
   end
  end
  if flag2 == 0
   flag1 = 1;
   dist4prof(k,1) = dist4prof(k,1) + 1;
  end
 end
 if flag1 == 0
  i=i+1;
  dist4prof(i,1) = 1;
  dist4prof(i,2:18) = dist4(j,2:18);
 end
end
```

All possible received words have been enumerated and there are no received words that are more than 4 away from all possible valid codewords. Put all the results in prof.

```
for i=1:18
 prof(1,i)=dist0prof(1,i);
 for j=2:3
  prof(j,i)=dist1prof(j-1,i);
 end
 for j=4:11
  prof(j,i)=dist2prof(j-3,i);
 end
```

```
 for j=12:21
  prof(j,i)=dist3prof(j-11,i);
 end
 prof(22,i)=dist4prof(j-21,i);
end
```

To calculate the probability for each type of distance profile, i.e., for each line in Table 3.1, assume BER=1.2×10^{-2}.

```
ber =0.012;
for k=1:22
 prof(k,19)=0;
 for i=2:18
  prof(k,19) = prof(k,19) + prof(k,i)*ber^(i-2)*(1-ber)^(18-i);
 end
 prof(k,19)= prof(k,19)*prof(k,1)/256;
end
```

That is the last column of Table 3.1.

Appendix C

Variances of N_0 , N_1 and N_2

From Table 3.1, the expectation $E\{f_0^2(x_j)\}$, $E\{f_1^2(x_j)\}$, $E\{f_2^2(x_j)\}$ and $E\{f_0(x_j)f_1(x_j)\}$, $E\{f_0(x_j)f_1(x_j)\}$, $E\{f_1(x_j)f_2(x_j)\}$ can be calculated. The variance $var\{N_i\}$ is:

$$var\{N_0\} = E\{N_0^2\} - E^2\{N_0\} = (E\{f_0^2(x_i)\})^L - (E\{f_0(x_i)\})^{2L} \tag{C.1}$$

For N_1^2,

$$\begin{aligned} N_1^2 &= \sum_{j=1}^{L} f_1(x_j) \prod_{i=1, i\neq j}^{L} f_0(x_i) \sum_{k=1}^{L} f_1(x_k) \prod_{m=1, m\neq k}^{L} f_0(x_m) \\ &= \sum_{j=1}^{L}\sum_{k=1}^{L} f_1(x_j)f_1(x_k) \prod_{i=1, i\neq j}^{L} \prod_{m=1, m\neq k}^{L} f_0(x_i)f_0(x_m) \end{aligned} \tag{C.2}$$

Taking the expectation of N_1^2,

$$
\begin{aligned}
E\{N_1^2\} &= \sum_{j=1}^{L}\sum_{k=1}^{L} E\{f_1(x_j)f_1(x_k)\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq k}^{L} f_0(x_i)f_0(x_m)\} \\
&= \sum_{j=1}^{L} E\{f_1(x_j)f_1(x_j)\}\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq j}^{L} E\{f_0(x_i)f_0(x_m)\} \\
&\quad + \sum_{j=1}^{L}\sum_{k=1,k\neq j}^{L} E\{f_1(x_j)f_1(x_k)\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq k}^{L} f_0(x_i)f_0(x_m)\} \\
&= L(E\{f_1^2(x_j)\})(E\{f_0^2(x_i)\})^{L-1} \\
&+ (L^2-L)(E\{f_1(x_j)f_0(x_j)\})^2(E\{f_0^2(x_i)\})^{L-2}
\end{aligned}
$$

$$
var\{N_i\} = E\{N_i^2\} - E^2\{N_i\} \tag{C.3}
$$

$$
N_2 = \sum_{j=1}^{L}(f_2(x_j)\prod_{i=1,i\neq j}^{L} f_0(x_i)) + \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}(f_1(x_i)f_1(x_j)\prod_{k=1,k\neq i,j}^{L} f_0(x_k)) \tag{C.4}
$$

Call the first term of N_2 a , and call the second term b, *i.e.*,

$$
a = \sum_{j=1}^{L} f_2(x_j)\prod_{i=1,i\neq j}^{L} f_0(x_i)
$$

$$
b = \sum_{i=1}^{L-1}\sum_{j=i+1}^{L} f_1(x_i)f_1(x_j)\prod_{k=1,k\neq i,j}^{L} f_0(x_k)
$$

then

$$
E\{N_2^2\} = E\{a^2\} + E\{b^2\} + 2E\{ab\} \tag{C.5}
$$

where

$$
\begin{aligned}
E\{a^2\} &= \sum_{j=1}^{L}\sum_{k=1}^{L} E\{f_2(x_j)f_2(x_k) \prod_{i=1,i\neq j}^{L} \prod_{m=1,m\neq k}^{L} f_0(x_i)f_0(x_m)\} \\
&= \sum_{j=1}^{L} E\{f_2(x_j)f_2(x_j)\} \prod_{i=1,i\neq j}^{L} \prod_{m=1,m\neq j}^{L} E\{f_0(x_i)f_0(x_m)\} \\
&+ \sum_{j=1}^{L}\sum_{k=1,k\neq j}^{L} E\{f_2(x_j)f_2(x_k) \prod_{i=1,i\neq j}^{L} \prod_{m=1,m\neq k}^{L} f_0(x_i)f_0(x_m)\} \\
&= L(E\{f_2^2(x_j)\})(E\{f_0^2(x_i)\})^{L-1} \\
&+ (L^2-L)(E\{f_2(x_j)f_0(x_j)\})^2(E\{f_0^2(x_i)\})^{L-2} \qquad \text{(C.6)}
\end{aligned}
$$

and

$$
\begin{aligned}
E\{b^2\} &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1}^{L-1}\sum_{l=k+1}^{L} E\{f_1(x_i)f_1(x_j)f_1(x_k)f_1(x_l) \\
&\prod_{m=1,m\neq i,j}^{L} \prod_{n=1,n\neq k,l}^{L} f_0(x_n)f_0(x_m)\} \qquad \text{(C.7)}
\end{aligned}
$$

The relationship between i, j, k, l is shown in Figure C.1. Fixing i, j, each term in the two inner summations (k, l summations) can be represented by a point in the (k, l) plane. Since $l = k + 1$ *to* L , points inside the triangular area that is bounded by the three solid lines represent the terms of the inner summation. $i = k$, $j = l$: In Figure C.1, this is the cross point (point A in Figure C.1) of the curve $k = i$ and $l - j$. Let $E\{b^2\}_1$ be the portion of $E\{b^2\}$ in this case,

$$
\begin{aligned}
E\{b^2\}_1 &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L} E\{f_1(x_i)f_1(x_j)f_1(x_i)f_1(x_j)\} \prod_{m=1,m\neq i,j}^{L} E\{f_0(x_m)f_0(x_m)\} \\
&= (L^2-L)/2(E\{f_1^2(x_j)\})^2(E\{f_0^2(x_i)\})^{L-2} \qquad \text{(C.8)}
\end{aligned}
$$

$k = i, l \neq j$: This case means the curve $k = i$ except the cross point (point A) in Figure C.1 with $l = j$. Let $E\{b^2\}_2$ be the portion of $E\{b^2\}$ in this case, it can be

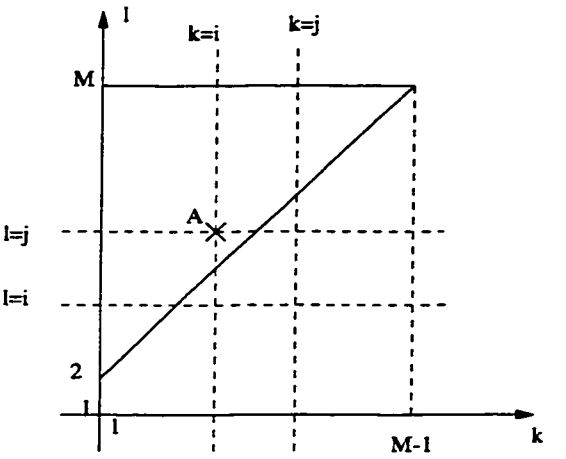

Figure C.1: Calculation figure for $E\{b^2\}$

calculated as:

$$
\begin{aligned}
E\{b^2\}_2 &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{l=i+1,l\neq j}^{L} E\{f_1(x_i)f_1(x_j)f_1(x_i)f_1(x_l) \\
&\quad \prod_{m=1,m\neq i,j}^{L}\prod_{n=1,n\neq i,l}^{L} f_0(x_n)f_0(x_m)\} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{l=i+1,l\neq j}^{L} E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}(L-i-1)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}(L-i)(L-i-1)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&\quad E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \frac{L(L-1)(L-2)}{3}E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3}
\end{aligned}
\tag{C.9}
$$

$l = j, k \neq i$: This case means the curve $l = j$ except the cross point with $k = i$ (point A) in Figure C.1. Let $E\{b^2\}_3$ be the portion of $E\{b^2\}$ in this case, it can be

calculated as:

$$
\begin{aligned}
E\{b^2\}_3 &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1,k\neq i}^{j-1} E\{f_1(x_i)f_1(x_j)f_1(x_k)f_1(x_j) \\
&\quad \prod_{m=1,m\neq i,j}^{L}\prod_{n=1,n\neq k,j}^{L} f_0(x_n)f_0(x_m)\} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1,k\neq i}^{j-1} E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}(j-2)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \frac{L(L-1)(L-2)}{3}E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3}
\end{aligned}
\tag{C.10}
$$

$l = i, k \neq j$: In Figure C.1, when $i = l$, since $j = i + 1$, and $k = l - 1$, so k is impossible to equal to j, i.e., there is no cross point for the two curves $l = i$ and $k = j$ within the triangle area. Let $E\{b^2\}_4$ be the portion of $E\{b^2\}$ in this case. It can be calculated as:

$$
\begin{aligned}
E\{b^2\}_4 &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1}^{i-1} E\{f_1(x_i)f_1(x_j)f_1(x_k)f_1(x_j) \\
&\quad \prod_{m=1,m\neq i,j}^{L}\prod_{n=1,n\neq k,i}^{L} f_0(x_n)f_0(x_m)\} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1}^{i-1} E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}(i-1)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}(L-i)(i-1)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \frac{L(L-1)(L-2)}{6}E\{f_1^2(x_i)\}(E(f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3}
\end{aligned}
\tag{C.11}
$$

$k = j, l \neq i$: In Figure C.1, when $k = j$, since $i = j - 1$, and $l = k + 1$, so l is impossible to equal to i, i.e., there is no cross point for the two curves $l = i$ and $k = j$ within the triangle area. Let $E\{b^2\}_5$ be the portion of $E\{b^2\}$ in this case. It

can be calculated as:

$$
\begin{aligned}
E\{b^2\}_5 &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{l=j+1}^{L} E\{f_1(x_i)f_1(x_j)f_1(x_k)f_1(x_j) \\
&\quad \prod_{m=1,m\neq i,j}^{L}\prod_{n=1,n\neq j,l}^{L} f_0(x_n)f_0(x_m)\} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{l=j+1}^{L} E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}(L-j)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \sum_{i=1}^{L-1}\frac{(L-i)(L-i-1)}{2}E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\
&= \frac{L(L-1)(L-2)}{6}E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3}
\end{aligned}
\tag{C.12}
$$

$k \neq i, j, l \neq i, j$: In Figure C.1, these are the points in the triangle area except for the curves $k = i$, $k = j$, $l = i$, and $l = j$. Let $E\{b^2\}_6$ be the portion of $E\{b^2\}$ in this case. It can be calculated as:

$$
\begin{aligned}
E\{b^2\}_6 &= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1,k\neq i,j}^{L}\sum_{l=k+1,l\neq i,j}^{L} E\{f_1(x_i)f_1(x_j)f_1(x_k)f_1(x_j) \\
&\quad \prod_{m=1,m\neq i,j}^{L}\prod_{n=1,n\neq k,l}^{L} f_0(x_n)f_0(x_m)\} \\
&= \sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\sum_{k=1,k\neq i,j}^{L}\sum_{l=k+1,l\neq i,j}^{L}(E\{f_1(x_j)f_0(x_i)\})^4(E\{f_0^2(x_i)\})^{L-4} \\
&= \frac{L(L-1)(L-2)(L-3)}{4}(E\{f_1(x_j)f_0(x_i)\})^4(E\{f_0^2(x_i)\})^{L-4}
\end{aligned}
\tag{C.13}
$$

Add all the results, the expectation of b^2 can be calculated as:

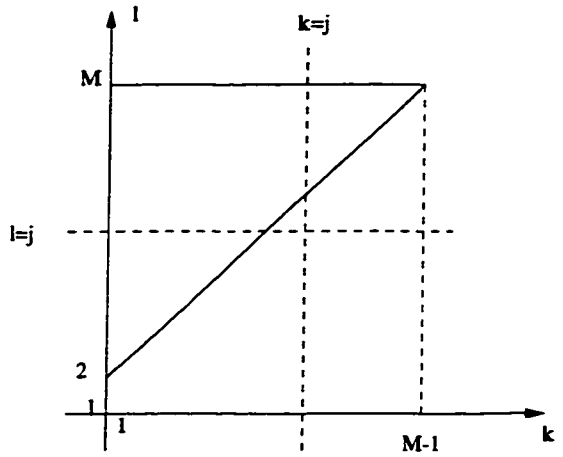

Figure C.2: Calculation figure for $E\{ab\}$

$$\begin{aligned} E\{b^2\} &= (L^2-L)/2(E\{f_1^2(x_j)\})^2(E\{f_0^2(x_i)\})^{L-2} \\ &+ L(L-1)(L-2)E\{f_1^2(x_i)\}(E\{f_1(x_i)f_0(x_i)\})^2(E\{f_0^2(x_i)\})^{L-3} \\ &+ \frac{L(L-1)(L-2)(L-3)}{4}(E\{f_1(x_j)f_0(x_i)\})^4(E\{f_0^2(x_i)\})^{L-4} \end{aligned} \tag{C.14}$$

Next the expectation of ab is calculated.

$$E\{ab\} = \sum_{j=1}^{L}\sum_{k=1}^{L-1}\sum_{l=k+1}^{L} E\{f_2(x_j)f_1(x_k)f_1(x_l)\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq k,l}^{L} f_0(x_i)f_0(x_m)\} \tag{C.15}$$

It also can be divided into several cases. The relationship between j, k, l is shown in Figure C.2. As C.1, since $l = k+1$ to L, the triangle area that is bounded by the three solid lines represents the possible points here.

$k = j$: In this case, l is impossible to equal to j. Let $E\{ab\}_1$ be the portion of $E\{ab\}$ in this case. It is calculated as:

$$
\begin{aligned}
E\{ab\}_1 &= \sum_{j=1}^{L}\sum_{l=j+1}^{L} E\{f_2(x_j)f_1(x_k)f_1(x_l)\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq k,l}^{L} f_0(x_i)f_0(x_m)\} \\
&= \sum_{j=1}^{L}\sum_{l=j+1}^{L} E\{f_1(x_i)f_2(x_i)\}E\{f_1(x_i)f_0(x_i)\})(E\{f_0^2(x_i)\})^{L-2} \\
&= (L^2-L)/2E\{f_1(x_i)f_2(x_i)\}E\{f_1(x_i)f_0(x_i)\})(E\{f_0^2(x_i)\})^{L-2}
\end{aligned}
\tag{C.16}
$$

$l = j$: In this case, k is impossible to equal to j. Let $E\{ab\}_2$ be the portion of $E\{ab\}$ in this case. It is calculated as:

$$
\begin{aligned}
E\{ab\}_2 &= \sum_{j=1}^{L}\sum_{k=1}^{j-1} E\{f_2(x_j)f_1(x_k)f_1(x_l)\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq k,l}^{L} f_0(x_i)f_0(x_m)\} \\
&= \sum_{j=1}^{L}\sum_{k=1}^{j-1} E\{f_1(x_i)f_2(x_i)\}E\{f_1(x_i)f_0(x_i)\})(E\{f_0^2(x_i)\})^{L-2} \\
&= \frac{(L^2-L)}{2}E\{f_1(x_i)f_2(x_i)\}E\{f_1(x_i)f_0(x_i)\})(E\{f_0^2(x_i)\})^{L-2}
\end{aligned}
\tag{C.17}
$$

$k \neq j$, $l \neq j$: In this case, the points are the triangle area in Figure C.2except for the two curves $k = j$, $l = j$. Let $E\{ab\}_3$ be the portion of $E\{ab\}$ in this case. It is calculated as:

$$
\begin{aligned}
E\{ab\}_3 &= \sum_{j=1}^{L}\sum_{k=1,k\neq j}^{L-1}\sum_{l=k+1}^{L} E\{f_2(x_j)f_1(x_k)f_1(x_l)\prod_{i=1,i\neq j}^{L}\prod_{m=1,m\neq k,l}^{L} f_0(x_i)f_0(x_m)\} \\
&= \frac{L(L-1)(L-2)}{2} \\
&\quad (E\{f_1(x_i)f_0(x_i)\})^2E\{f_2(x_i)f_0(x_i)\}(E\{f_0^2(x_i)\})^{L-3}
\end{aligned}
\tag{C.18}
$$

Add these three cases, then the expectation of ab is:

$$\begin{aligned} E\{ab\} &= (L^2 - L)E\{f_1(x_i)f_2(x_i)\}E\{f_1(x_i)f_0(x_i)\})(E\{f_0^2(x_i)\})^{L-2} \\ &+ \frac{L(L-1)(L-2)}{2}(E\{f_1(x_i)f_0(x_i)\})^2 E\{f_2(x_i)f_0(x_i)\}(E\{f_0^2(x_i)\})^{L-3} \end{aligned} \tag{C.19}$$

Using Equation C.5, the expectation of N_2^2 can be calculated.