ERROR RESILIENT STEREOSCOPIC VIDEO STREAMING USING MODEL-BASED FOUNTAIN CODES

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BİLKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

A. Serdar Tan

January 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Erdal Arıkan (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Levent Onural

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Gözde Bozdağı Akar

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assoc. Prof. Dr. Uğur Güdükbay

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Asst. Prof. Dr. Defne Aktaş

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Science

ABSTRACT

ERROR RESILIENT STEREOSCOPIC VIDEO STREAMING USING MODEL-BASED FOUNTAIN CODES

A. Serdar Tan
Ph.D. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Erdal Arıkan
January 2009

Error resilient digital video streaming has been a challenging problem since the introduction and deployment of early packet switched networks. One of the most recent advances in video coding is observed on multi-view video coding which suggests methods for the compression of correlated multiple image sequences. The existing multi-view compression techniques increase the loss sensitivity and necessitate the use of efficient loss recovery schemes. Forward Error Correction (FEC) is an efficient, powerful and practical tool for the recovery of lost data. A novel class of FEC codes is Fountain codes which are suitable to be used with recent video codecs, such as H.264/AVC, and LT and Raptor codes are practical examples of this class. Although there are many studies on monoscopic video, transmission of multi-view video through lossy channels with FEC have not been explored yet. Aiming at this deficiency, an H.264-based multi-view video codec and a model-based Fountain code are combined to generate an efficient error resilient stereoscopic streaming system. Three layers of stereoscopic video with unequal importance are defined in order to exploit the benefits of Unequal Error Protection (UEP) with FEC. Simply, these layers correspond to intra

frames of left view, predicted frames of left view and predicted frames of right view. The Rate-Distortion (RD) characteristics of these dependent layers are defined by extending the RD characteristics of monoscopic video. The parameters of the models are obtained with curve fitting using the RD samples of the video, and satisfactory results are achieved where the average difference between the analytical models and RD samples is between 1.00% and 9.19%. An heuristic analytical model of the performance of Raptor codes is used to obtain the residual number of lost packets for given channel bit rate, loss rate, and protection rate. This residual number is multiplied with the estimated average distortion of the loss of a single Network Abstraction Layer (NAL) unit to obtain the total transmission distortion. All these models are combined to minimize the end-to-end distortion and obtain optimal encoder bit rates and UEP rates. When the proposed system is used, the simulation results demonstrate up to 2dB increase in quality compared to equal error protection and only left view error protection. Furthermore, Fountain codes are analyzed in the finite length region, and iterative performance models are derived without any assumptions or asymptotical approximations. The performance model of the belief-propagation (BP) decoder approximates either the behavior of a single simulation results or their average depending on the parameters of the LT code. The performance model of the maximum likelihood decoder approximates the average of simulation results more accurately compared to the model of the BP decoder. Raptor codes are modeled heuristically based on the exponential decay observed on the simulation results, and the model parameters are obtained by line of best fit. The analytical models of systematic and non-systematic Raptor codes accurately approximate the experimental average performance.

Keywords: Fountain Codes, Forward Error Correction, Video Streaming, Stereoscopic Video.

ÖZET

MODEL TABANLI FOUNTAIN KODLARI KULLANARAK HATAYA DAYANIKLI STEREO VİDEO AKITIMI

A. Serdar Tan
Elektrik ve Elektronik Mühendisliği Bölümü Doktora
Tez Yöneticisi: Prof. Dr. Erdal Arıkan
Ocak 2009

Hataya dayanıklı sayısal video akıtımı, paket anahtarlamalı ağların ortaya çıkmasından ve yayılmasından bu yana ilgi çekici ve zor bir problem olmuştur. Video kodlama alanındaki en yeni gelişmelerden biri ilinliti çoklu imge dizilerinde sıkıştırma için metodlar öneren çok-görüşlü kodlayıcı-çözücülerde görülmektedir. Mevcut çok-görüşlü sıkıştırma teknikleri yitimlere olan duyarlılığı arttırmakta ve yitim kurtarma yöntemlerinin kullanımını gerektirmektedir. Gönderme Yönünde Hata Düzeltimi (GYHD) yitik verilerin kurtarılması için verimli, kuvvetli ve uygulanabilir bir araçtır. Fountain kodları GYHD kodlarının yeni bir sınıfıdır ve LT ve Raptor kodları bu sınıfın uygulanabilir örnekleridir. Bu kodlar H.264/AVC gibi en yeni video kodlayıcı-çözücüler ile uyumlu çalışabilmektedir. Tek görüşlü video ile ilgili bir çok çalışma olmasına rağmen, çok-görüşlü videonun yitimli kanallarda GYHD ile iletimi yeteri kadar incelenmemiştir. Bu eksikliği hedef alarak, verimli ve hataya dayanıklı bir stereo video akıtım sistemi oluşturmak için H.264 temelli çok-görüşlü bir video kodlayıcı-çözücü ve model temelli bir Fountain kodu bir arada kullanılmıştır. GYHD ile birlikte eşit olmayan hata koruması (EOHK) kullanmanın faydalarından yararlanmak için stereo videonun farklı önemlere sahip üç katmanı tanımlanmıştır. Temel olarak, bu üç

katman sol görüşün çerçeve içi kodlanmış çerceveleri, sol görüşün öngörülmüş çerçeveleri ve sağ görüşün öngörülmüş çerçevelerinden oluşur. Tek görüşlü videonun Hız-Bozulum (HB) karakteristiği genişletilerek bağımlı katmanların HB karakteristiği tanımlanmıştır. Videonun HB örneklerini kullanarak eğri oturtma tekniği ile model parametreleri elde edilmiştir, ve HB örnekleri ile analitik model arasındaki ortalama farkın %1.00 ve %9.19 arasında olduğu tatminkar sonuçlar elde edilmiştir. Kanal bit hızı, kayıp oranı ve koruma oranı verildiğinde kalan kayıp paket sayısını elde etmek için Raptor kodlarının buluşsal bir analitik modeli kullanılmıştır. Bu kalan sayı, tek bir Ağ Soyutlama Katmanı (ASK) biriminin yitiminden kaynaklanan kestirilmiş ortalama bozulum ile çarpılarak toplam iletim bozulumu elde edilmiştir. Bütün bu modeller, uçtan-uca bozulumu enküçültmek ve en iyi kodlayıcı bit hızlarını ve EOHK oranlarını elde etmek için birleştirilmiştir. Önerilen sistem kullanıldığında, eşit hata koruması ve sadece sol görüş koruması yöntemleri ile karşılaştırıldığında, benzetim sonuçları 2dB'ye varan kalite artışı göstermiştir. Bundan başka, Fountain kodları sonlu uzunluk bölgesinde analiz edilmiş ve varsayımlar ya da asimtotik yaklaşıklamalar olmadan döngülü başarım modelleri türetilmiştir. İnanç-Yayılım (İY) kodçözücüsünün başarım modeli LT kodunun parametrelerine bağlı olarak ya tek bir benzetim sonucunu ya da bu sonuçların ortalamasını yaklaşıklamıştır. En büyük olabilirlik kodçözücüsünün başarım modeli, İY kodçözücüsünün modeli ile kıyaslandığında, benzetim sonuçlarının ortalamasını daha iyi yaklaşıklamıştır. Raptor kodları, benzetim sonuçlarında görülen üstel azalmaya dayanarak, buluşsal olarak modellenmiş ve model parametreleri en iyi oturan doğru ile elde edilmiştir. Sistematik ve sistematik olmayan Raptor kodlarının anatilik modelleri deneysel ortalama başarımı doğru bir şekilde yaklaşıklamıştır.

Anahtar Kelimeler: Fountain Kodları, Gönderme Yönünde Hata Düzeltimi, Kesintisiz Video İletimi, Stereo Video.

ACKNOWLEDGMENTS

I gratefully thank to my supervisor Prof. Dr. Erdal Arıkan for his guidance, support and keen concern throughout the development of this thesis. I would like to express my gratitude to the members of the thesis monitoring committee Prof. Dr. Levent Onural and Prof. Dr. Gözde Bozdağı Akar for their valuable comments and guidance. I would also like to thank the committee members Assoc. Prof. Dr. Uğur Güdükbay and Asst. Prof. Dr. Defne Aktaş for accepting to read the manuscript and commenting on the thesis.

I would also like to thank the members of the Multimedia Research Group of Prof. Dr. Gözde Bozdağı Akar in METU. I specially thank to Anıl Aksay for his collaboration.

Special thanks to Namık Şengezer, Ayça Özçelikkale, Mehmet Köseoğlu, Kıvaç Köse, Gökhan Bora Esmer, Yiğitcan Eryaman, Avşar Polat Ay, Onur Taşçı, Kaan Doğan, Güray Gürel, Sıla Kurugöl and Onur Afacan for making the past 5 years of the office life enjoyable with their endless support and friendship.

It is a great pleasure to express my gratitude to my family Kezban, Mustafa, Ayşe and Mehmet Tan. Without their support, this work would not be possible. Finally, I sincerely thank to my love Selin for making me love the life and making the thesis process bearable.

This thesis work is supported by the EC under contract FP6-511568 3DTV and by the graduate scholarship program of TÜBİTAK (Scientific and Technical Research Council of Turkey).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Video streaming through lossy channels has received considerable attention for the past 20 years. The worldwide increases in the number of Internet access, dedicated bandwidth and video sharing websites have triggered the research.

1.1 Problem Statement

Increasingly more and more data have to be distributed over lossy transmission channels with limited bandwidth. Stereoscopic video is emerging as a new source of data for the next generation communication systems. Owing to its high bandwidth and loss sensitivity, optimal transmission strategies for stereoscopic video should be derived to obtain efficiency. Furthermore, optimal transmission through lossy channels necessitates the use of the most advanced error correction schemes and their corresponding analysis.

There are some constraints on such a system in order to be deployed over existing infrastructure. First of all, it has to be simple and piecewise analyzable. The system has to be efficient and optimal in a sense that it compensates the simple design. Flexibility and scalability, which are required for the ease of bit

rate adaptability, are also important constraints. Finally, the system has to be robust against transmission errors, the main reason of the distortion in video quality. Thus, the chosen error correction scheme has to be fully analyzed and understood.

In this thesis, we consider optimal transmission strategies for an end-to-end stereoscopic streaming system that uses an H.264-based multi-view video codec. We use a Fountain code for recovery from packet losses, and investigate their performance in detail.

1.2 Background

The data losses during transmission in Internet are observed due to several reasons. In order to understand the sources of error, we should look at the protocol stacks. In Figure 1.1, the protocol stacks specific to a recent video codec, H.264, are presented. The transmission errors occur only in the IP and physical layers. In the physical layer, bit errors occur due to the noise in the transmission. In the IP layer, packet losses occur due to congestion in the routers. Caused by either physical or IP layer, the error is observed as packet loss in the upper layers. In order to protect from the packet losses in video streaming, application layer Forward Error Correction (FEC) can be used. In such a scenario, the place of the FEC is demonstrated in Figure 1.1. In the following sections, we briefly define error correction and video streaming.

1.2.1 Error Correction

After Shannon defined the fundamentals of channel capacity in the landmark paper [1] in 1948, many researchers around the world studied different techniques to approach the channel capacity in order to achieve the best error protection.

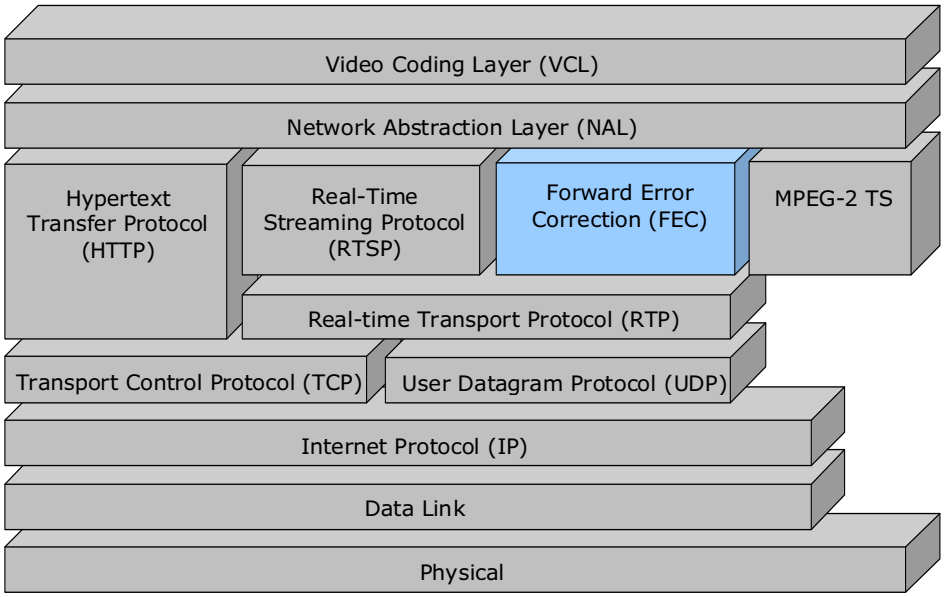


Figure 1.1: The protocol stacks for video streaming

Among these, FEC scheme came out to be one of the best way of approaching channel capacity in a feedback-free transmission system. In an FEC scheme, the encoder introduces redundant bits to the original message. Then, in case of any losses, the decoder tries to recover the original message with the help of these redundant bits. Some of the primary examples of FEC codes are Hamming codes [2], BCH codes [3], Reed-solomon (RS) codes [4] etc.. Among these primary examples RS coding is the most notable due to its widespread deployment in current storage media such as CDs, DVDs and HDDs. The encoding and decoding of the primary examples are problematic for large message lengths due to their high complexities. One of the recently used, but not recently invented examples of FEC codes is low-density parity-check (LDPC) codes. Invented in 1960 by Gallager [5], LDPC codes were impractical to implement, because they are capacity approaching for very large message lengths. Three decades later, LDPC codes were rediscovered and took place in many standards such as Digital Video Broadcasting - Satellite - Second Generation (DVB-S2) [6] in 2003. In this standard, message length varies from 16200 to 64800 bits. Using the belief propagation decoding [7] with large message lengths, LDPC coding is nearly capacity

achieving as shown in [8]. Operating on the sparse (low density) parity check matrix, the belief propagation decoder can achieve a computational complexity linear with the message length. In the early 90s a novel FEC scheme, Turbo coding [9] which combined two or more convolutional codes and a block interleaver that has led to another capacity approaching channel code, was proposed.

The general application area of LDPC and Turbo codes are on noisy channels such as Additive White Gaussian Noise Channel (AWGNC), Binary Symmetric Channel (BSC) or Binary Erasure Channel (BEC). These codes are implemented on the physical layer of transmission systems. However, the most widely deployed networks are wired packet switched networks, such as Internet, and the source of error in these networks is packet loss which is generally caused by congestion or other network problems, and not usually by physical layer errors. Thus applying channel coding in physical layer is not a proper way of protection in these networks. The underlying channel in packet switched networks is denoted as packet erasure channel (PEC). In the PEC a packet is either received completely intact and error free or it is lost. Thus, a lost packet with all its bits is the erasure.

The most widely deployed error detection/correction technique for PEC is the Automatic Repeat Request (ARQ) scheme that the Transport Control Protocol (TCP) utilizes. In the ARQ scheme the receiver detects missing packets and automatically requests their retransmission from the source. The transmission of the packets is ordered, so that when a packet is lost subsequent packets wait the retransmission of the lost packet. This scheme may cause feedback implosion in a broadcasting scenario, especially when the loss rate is high.

A novel technique that recently became popular for error protection in lossy packet networks is Fountain codes which is also called rateless codes. The Fountain coding idea is proposed in [10] and followed by practical realizations such as (Luby Transform) LT codes [11] and Raptor codes [12]. Raptor codes are extended from LT codes by inserting a fixed-rate pre-code before LT coding stage.

Fountain codes remove the necessity of orderly transmission and retransmissions which prevents the feedback implosion that occurs when ARQ is used. They also produce as many parity packets as needed on-the-fly. This approach is different than the general idea of FEC codes where channel encoding is performed for a fixed channel rate and all encoded packets are generated prior to transmission. In the traditional FEC codes, if the number of losses exceeds a certain amount, then the code cannot be extended on-the-fly to have a lower rate to achieve higher loss protection.

LT codes achieve on-the-fly rate extension capability and low complexity in exchange of some performance. Their performance depends on two factors. First one is the degree distribution which is used for generating the output symbols from the input symbols, and it has direct effect on encoding and decoding complexities. Thus, the degree distribution is designed so that it minimizes the complexity. Second factor is the block length, namely the number of input symbols. The performance of Raptor codes are affected by another factor which is the pre-coding stage. Fountain codes are asymptotically optimal, hence they operate efficiently when the number of input symbols is very high. However, Raptor codes are an exception, because their performance is quite acceptable for low number of input symbols, as well. The analysis of the performance, depending on the degree distribution and block length, needs detailed study to obtain accurate results. In [13], an analysis of the performance of LT codes yielded an iterative analytical model of their performance with significant high complexity. In [12], rather than exact analysis and modeling, some bounds on the performance of LT codes and Raptor codes are presented. The analysis and modeling of fountain codes are beneficial for optimization in end-to-end transmission systems.

Although being recently proposed, fountain codes are protected by several patents and appear in several standards. Raptor codes appear in the standards

Digital Video Broadcast for Hand-held (DVB-H) [14] and 3^{rd} Generation Partnership Project (3GPP) [15] in multimedia broadcast. In both of the standards Raptor coding is applied as an application layer FEC to recover the lost packets and the details on the Raptor codec is described in the IETF draft in [16].

1.2.2 Video Coding and Streaming

First practically applicable video coding standards appeared in the early 90s such as H.261 [17] in 1990 and MPEG-1 [18] in 1991. Since then, the compression efficiency of the codecs have increased and many novel tools have been introduced. The main idea in video compression in the recent video codecs is similar to the early ones. Currently, all of the available video codecs use the temporal dependency between subsequent frames to achieve compression. Fundamentally, the frames are divided into two groups: Intra-frames and inter-frames. The intra-frames are compressed by standard still image compression techniques, similar to JPEG [19], thus they are self-decodable. Generally, the location of the intra-frames determines the beginning of group of pictures (GOP). The remaining frames that reside between intra-frames are called inter-frames. Inter-frames are compressed by using one or more previous or subsequent frames, eventually using intra-frames, and most of the gain in video compression is achieved by this feature.

Video streams often have some scalability or unequal loss sensitivity. Basically, the loss of the packets of intra-frames reduce the quality of video more than the loss of the packets of inter-frames. Because inter-frames are useless without the intra-frames. Thus, intra-frames need more protection against losses. Similarly, some parts of the images in a video may have more dominant motion characteristics. This leads to Flexible Macroblock Ordering (FMO) [20], [21] which partitions the macroblocks according to their impact on the video quality. Data Partitioning [22], [23] is another method for prioritization of bitstream

elements of video, such as dc and ac coefficients, motion vectors, headers etc., based on their sensitivity to errors. Unequal loss sensitivity exists in scalable video codecs where the bitstream is composed of base layer and enhancement layers. Scalable video coding uses techniques such as spatial, temporal or SNR scalability. In [24], the Scalable Video Coding (SVC) extension of H.264/AVC is explained in a broader sense. In the cases where the video data parts have unequal loss sensitivity, utilization of unequal error protection (UEP) offers significant increase in video quality.

In order to further improve the visual experience, 3-dimensional video coding techniques are also proposed. Multi-view video coding is an extension of standard single-view (monoscopic) video coding techniques to more than one views (cameras) [25], [26]. Multi-view video is formed by the simultaneous capture of a scene by more than one cameras which are separated with an acceptable distance. Eventually, capturing more than one video sequence increases the amount of source data. Existing multi-view coding techniques tries to reduce the size by the compression techniques that exploit the dependency between these views. For this purpose, a new technique, inter-spatial frame coding is introduced and used, besides intra and inter-frame coding. The sophisticated structure of multi-view compression and increased dependencies between frames deduces even more data groups with different loss sensitivities.

1.3 Contributions

In this thesis, we specifically focus on stereoscopic video streaming. Numerous components have to be combined to work harmoniously in order to realize streaming in lossy channels. Consequently, it is very difficult, if not impossible, to obtain an optimal end-to-end streaming system. In this sense, since there exist numerous parameters, stereoscopic video streaming in lossy transmission

channels with FEC becomes a quite difficult joint source-channel coding problem. Thus, one needs to simplify the problem and handle it by dividing into smaller pieces in exchange of moving away from optimality a little bit. In this thesis, we specify the most important contributions as; basic partitioning of the video according to unequal importance, obtaining rate-distortion properties of these partitions, analysis of the utilized FEC scheme for UEP, and end-to-end distortion minimization. We provide separate analysis for each part and obtain mathematical models that we use for distortion minimization. The proposed analytical models for RD curve of the layers and performance of Raptor codes are quite accurate. After estimating the average distortion of a single lost packet, we define a model for end-to-end distortion, namely the sum of encoder and transmission distortions, and minimize it to obtain the optimal encoder and protection bit rates.

Apart from the stereoscopic video streaming, we also focus on the analysis of Fountain codes in detail. We aimed at the deficiency of analytical modeling in the literature for fountain codes that would be beneficial in the optimization of video streaming systems for lossy transmission. We give special attention to LT codes with Belief Propagation (BP) and Maximum Likelihood (ML) decoder and systematic and non-systematic Raptor codes. We contribute in two ways, first by providing a detailed analysis of these codes, and second, by proposing analytical models for their performances.

1.4 Outline

The organization of the thesis follows. In Chapter 2, we present the basics of Fountain codes and video streaming with their historical overviews. We describe the operation of Fountain codes with intuitive examples and explain video coding in brief. In Chapter 3, we start with the motivation of the use of FEC with video

streaming. Then, we analyze the components of stereoscopic video streaming and propose an error-resilient system with Fountain codes. In Chapter 4, initially, we describe the operation of Fountain codes in detail and present performance results with simulations. Then, we analyze them and propose analytical performance models. In Chapter 5, we conclude and state possible future work.

Chapter 2

Related Work and Definitions

2.1 Basics of Fountain Coding

The idea of Fountain coding is different from the original FEC idea where channel encoding is performed for a fixed channel rate and all encoded packets are generated prior to transmission. The Fountain encoder is an imaginary fountain of limitless supply of water drops (output symbols). Any person who wants to reconstruct all of the input symbols has to wait to fill their bucket with slightly more water drops than the number of input symbols. Thus, the main idea behind Fountain coding is to produce as many output symbols as needed on-the-fly. This property gave another name to fountain codes, rateless codes.

The principle of Fountain codes is illustrated in Figure 2.1. The encoder generates potentially limitless output symbols (water drops) from the input data. The decoder (bins) try to collect enough number of output symbols to complete decoding and reconstruct the input symbols. During the transmission (collecting water drops), output symbols may get lost due to the channel conditions. In such a case, decoders do not send any retransmission request messages back to

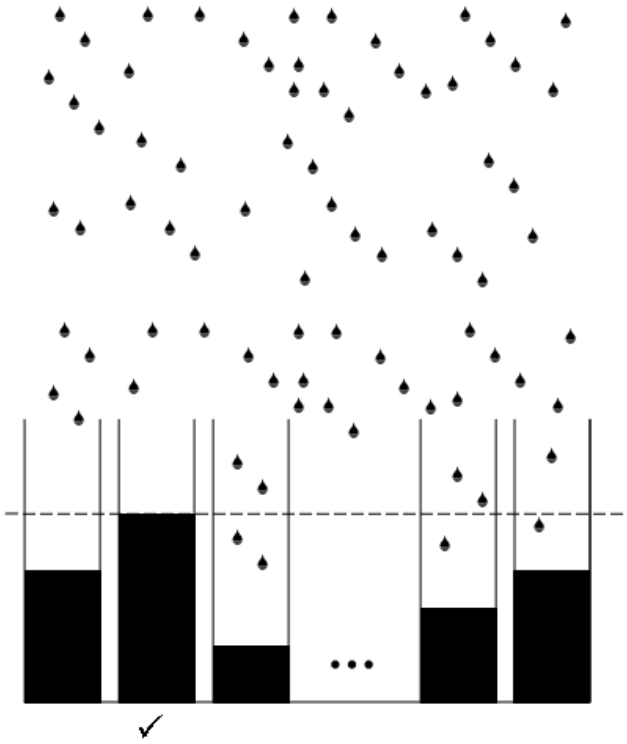


Figure 2.1: Overview of Fountain coding: The decoders (bins) try to collect sufficient number of output symbols (water drops)

the encoder. They just wait to receive enough number of output symbols to complete the decoding.

Digital Fountain approach is first described in 1998 as a novel technique for reliable distribution of bulk data [10]. In the same year, a company named Digital Fountain Inc. is founded by Charlie Oppenheimer and Dr. Michael Luby in CA, U.S. with the aim of commercializing and standardizing the fountain coding approach. In 1999, first patent on fountain coding appeared [27], and it has been revised several times under same title [28], [29], [30], [31]. In 2002, Luby Transform (LT) codes, named after Michael Luby, are published in [11] that described the coding scheme in the patents. The coding scheme attracted significant interest and various papers on LT codes have been published since then.

Raptor coding is proposed as a multi-stage extension of LT coding. They are invented in 2000 and patented [32]. Publication of Raptor codes appeared first in a technical report in 2003 [33] and then in 2006 [12]. They are still one of the most advanced fountain coding scheme, and similar to LT coding, attracted a wide interest.

As mentioned in [11], the original idea and purpose of LT codes is the distribution of bulk data to many users. Most common application area is the distribution of a service pack of an operating system to many computers in the world. In such a scenario the number of users will be huge and each user will experience different channel characteristics. If standard techniques are used such as TCP, the server and client has to communicate for each lost packet. Fountain codes are a candidate for this situation for reliable feedback-free transmission where users just wait to receive enough packets.

LT codes are considered suitable also for data storage in hard drives. Originally, the data in hard drives are stored in successive segments, and when the reading head misses one track it has to retrace to read again which causes significant delays. On the other hand, if fountain approach is used, the head does not have to retrace; it just skips and reads enough number of next segments. Thus, the hard drive can operate faster.

The encoding and decoding of LT codes are also defined in [11]. However, the proposed decoding algorithm is asymptotically optimal, which necessitates large block lengths. Thus, the problematic operation of LT codes on short and medium block lengths lead the introduction of ML decoder. Eventually, ML decoder performs better and its complexity is higher compared to original decoder, but for short block lengths its complexity might be acceptable.

The inefficiency of LT codes for short block lengths also lead the introduction of Raptor codes. After Raptor codes, fountain coding idea has taken part in

many applications, because Raptor codes have reasonably better performance than their prior LT codes. The main target became distribution of multimedia data, such as video. Because video has sufficiently large block size and Raptor codes have low latency in encoding and decoding, which allows real-time error protection.

In order to be more suitable for multimedia distribution, the structure of fountain codes are also modified. Their original structure was non-systematic, namely each generated symbol is obtained by transforming the input symbols to new symbols. This property causes an all or nothing code where all of the symbols are either undecoded or decoded according to the loss rate of the transmission channel. For time limited applications, where destination may not wait for more output symbols, such as video streaming, this may become devastating. Treating the fountain codes as a fixed rate FEC code, systematic fountain coding is also proposed. By this way, partial recovery of data is still possible in case of excessive packet losses. This process requires limited time which may seem controversial to fountain coding approach; however, fast (low complexity) encoding and decoding of fountain codes render them suitable for time limited applications, as well.

In the following sections, we describe the widely known Fountain codes; random X-OR, Tornado, LT and Raptor codes which have attracted significant interest in the recent years.

2.2 Random X-OR codes

Random X-OR coding is the simplest of Fountain codes where each output symbol is generated as a random X-OR sum of the input symbols. Let I_i be the row vector os size s that denotes the i^{th} input symbol ($1 \leq i \leq k$), and O_j be the row vector of size s that denotes the j^{th} output symbol generated ($1 \leq j$). In order

to represent the generation of random output symbols, let $\mathrm{A}_j = [a_{j,1}, a_{j,2}, ..., a_{j,k}]$ denote the generator vector of j^{th} output symbol, where

$$a_{j,i} = \begin{cases} 0 & w.p.\ 1/2 \\ 1 & w.p.\ 1/2 \end{cases} \quad \text{for } 1 \leq j,\ 1 \leq i \leq k\ . \tag{2.1}$$

Furthermore, let

$$\mathbf{I} = \begin{bmatrix} \mathrm{I}_1 \\ \vdots \\ \mathrm{I}_k \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} \mathrm{A}_1 \\ \mathrm{A}_2 \\ \vdots \end{bmatrix}, \quad \mathbf{O} = \begin{bmatrix} \mathrm{O}_1 \\ \mathrm{O}_2 \\ \vdots \end{bmatrix}. \tag{2.2}$$

Then, the generation of j^{th} output symbol and the whole output symbols are given, respectively as

$$\mathrm{O}_j = \bigoplus_{i=1}^{k} a_{j,i}\mathrm{I}_i = \mathrm{A}_j \cdot \mathbf{I}\ , \tag{2.3}$$

$$\mathbf{O} = \mathbf{A} \cdot \mathbf{I}\ . \tag{2.4}$$

In (2.4), infinitely many output symbols, 2^k of them being distinct, are generated from k input symbols with the semi-infinite generator matrix $\mathbf{A}$. The decoder tries to collect as many of these output symbols to reconstruct the input symbols.

Assume that the generator matrix for $k = 3$ is

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{matrix} \rightarrow \\ \rightarrow \\ \rightarrow \\ \times \\ \rightarrow \\ \vdots \end{matrix} , \tag{2.5}$$

where $\times$ denotes loss and $\rightarrow$ denotes success during transmission. Having received the first three symbols, the decoder cannot reconstruct the input symbols since the A_j's of the received symbols do not add up to a matrix with rank 3. When the fifth symbol arrives, after the fourth equation which was erased, the received generator matrix becomes full rank, namely all input symbols are determined.

In the decoding process, k independent output symbols must arrive for successful recovery. Let $p(j)$ denote the probability that an arriving symbol is dependent when j independent symbols arrived previously, which can be calculated as $p(j) = (1 - 2^{-(k-j)})$. Then, the average waiting time to receive the next independent symbol can be calculated as $W_j = 1/p(j)$ by geometric random series analysis. The total time to receive k independent symbols can be calculated as

$$\begin{aligned} \sum_{j=0}^{k-1} W_j &= \sum_{j=0}^{k-1} \frac{1}{(1 - 2^{-(k-j)})} \\ &= \sim (k + 2) \, . \end{aligned} \tag{2.6}$$

Thus, on the average, arrival of $k + 2$ output symbols ensures successful decoding of the input symbols. However, due to the $O(k^3)$ complexity of standard elimination algorithms for dense matrices, the complexity of the decoder is the main drawback of X-OR codes. Lower complexity and efficient decoding algorithms can be achieved by using degree distributions for the generated output

symbols, where the number of X-OR summed input symbols will be chosen from a distribution, so that

$$\sum_{i=1}^{k} a_{j,i} = d_j \ , \tag{2.7}$$

where d_j is the degree of the j^th output symbol chosen from a degree distribution.

2.3 Tornado Codes

Tornado codes [10], [34] are proposed to be the first example of fountain codes. They can generate infinitely many encoded symbols from k input symbols. When $k(1+\varepsilon)$ of these input symbols arrive to the decoder, all of the k input symbols can be decoded. However, the code is designed for a fixed number of encoded symbols n. The encoder can generate at most n output symbols and cannot produce more symbols on-the-fly. Moreover, the required overhead, $k\varepsilon$, is at least a constant fraction of the number of input symbols. Due to this property we do not consider Tornado codes as an ideal fountain code.

2.4 LT Codes

LT codes proposed in [11] have initiated a novel research area. Instead of operating on low density parity check matrices, LT codes operate on low density generator matrices. In [11], it is shown that for large message length k LT codes are capacity achieving.

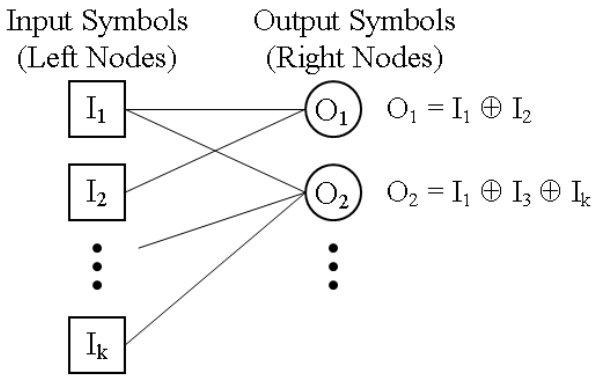


Figure 2.2: Bipartite graph for LT coding

2.4.1 Encoder

The unit of encoding in LT codes is called a symbol which is a data packet of size m bits for some fixed $m \geq 1$. Let the number of input symbols be k, then the input data size is $(k \cdot m)$ bits. The number of the output symbols that can be generated from these input symbols is virtually limitless, i.e. the code is rateless. In Figure 2.2, generation of the first two output symbols are demonstrated on the encoding graph, where $A_1 = [1, 1, 0, ..., 0]$ and $A_2 = [1, 0, 1, 0, ..., 1]$. In Figure 2.2, the number of edges connected to an output symbol is its degree d, for example O_1 has degree-2. The edges emanating from an output symbol of degree d are connected randomly to d input symbols. These input symbols are denoted as the neighbors of the output symbols to which they are connected.

The number of lines that connect the input symbols to output symbols defines the complexity of the encoding process. Let $\bar{d}$ represent the average degree of an output symbol and assume that $\bar{d} = \ln(k)$. Then, based on the fact that $k(1 + \varepsilon)$ output symbols are enough to reconstruct the input symbols, the complexity is $O(k \ln(k))$.

2.4.2 Decoder

There are two different LT decoding schemes: Belief Propagation (BP) decoder and Maximum Likelihood (ML) decoder. BP decoder is the original low complexity decoder proposed in [11], but it is optimal for large values of k. ML decoder is the optimal decoder with high complexity.

Belief Propagation (BP) Decoder

Belief propagation decoding of LT codes is an iterative process best explained by an example. Assume that the number of input symbols is four, and the generator matrix and the received output symbols are given as

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{array}{l} \rightarrow \quad \mathrm{O}_1 = \mathrm{I}_1 \oplus \mathrm{I}_2 \\ \rightarrow \quad \mathrm{O}_2 = \mathrm{I}_1 \\ \rightarrow \quad \mathrm{O}_3 = \mathrm{I}_2 \oplus \mathrm{I}_4 \\ \times \\ \rightarrow \quad \mathrm{O}_5 = \mathrm{I}_3 \oplus \mathrm{I}_4 \\ \vdots \end{array} . \tag{2.8}$$

Having received O_1, the decoder can not decode I_1 or I_2. However, with O_2, a degree-1 output symbol, the decoder recovers I_1 and I_2 by $\mathrm{I}_1 = \mathrm{O}_2$, $\mathrm{I}_2 = \mathrm{O}_1 \oplus \mathrm{O}_2$. All of the steps of decoding for this specific example are given below.

- $\mathrm{O}_2 = \mathrm{I}_1 \Rightarrow \mathrm{I}_1$ is determined,
- $\mathrm{O}_1 \oplus \mathrm{I}_1 \Rightarrow \mathrm{I}_2$ is determined,
- $\mathrm{O}_3 \oplus \mathrm{I}_2 \Rightarrow \mathrm{I}_4$ is determined,
- $\mathrm{O}_5 \oplus \mathrm{I}_2 \oplus \mathrm{I}_4 \Rightarrow \mathrm{I}_3$ is determined.

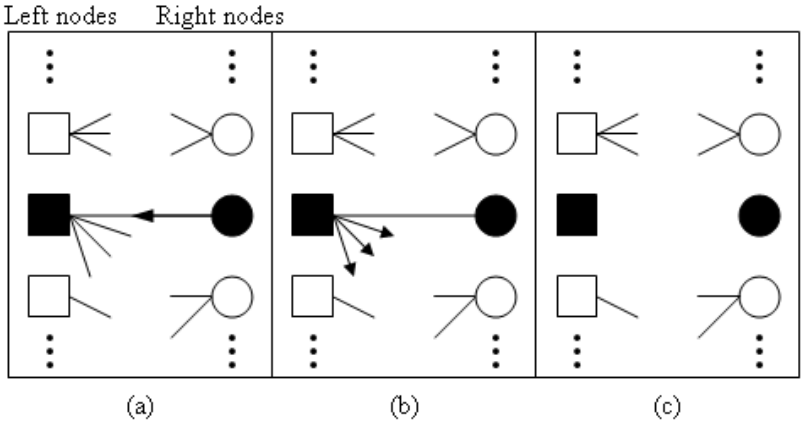


Figure 2.3: A step in the LT BP decoder

Initially, there exists an output symbol with degree 1 in the above example. If, there were no output symbols with degree 1 then the LT BP decoding procedure could not decode the input symbols with these set of output symbols. In such a case, the decoder has to wait for more output symbols to succeed in decoding. The algorithm of the BP decoding is given in the following steps, which are associated with the decoding phases in Figure 2.3.

1. Find an output symbol O_j that is connected to only one input symbol I_i, i.e. find an output symbol with degree 1. If there is no such symbol the decoding fails (Phase-(a)).

2. Set $I_i = O_j$ (Phase-(a)).

3. X-OR sum all of the output symbols that are connected to i^{th} input symbols with its value I_i (Phase-(b)).

4. Remove the edges of the input symbol $\mathbf{I}_i$ from the graph (Phase-(c)).

5. Go to step 1 until all I_i are determined.

Similar to the encoder, the decoding complexity of this algorithm is proportional to the number of edges. When the average degree of an output symbol is

$\bar{d}$, and assuming that $\bar{d} = \ln(k)$, the complexity is $O(k \ln(k))$ which is the same as LT encoder.

Maximum Likelihood (ML) Decoder

In the belief propagation decoder, single degree output symbols have to exist throughout the decoding process. Consider the set of received output symbols

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{matrix} \rightarrow & \mathrm{O}_1 = \mathrm{I}_1 \oplus \mathrm{I}_2 \\ \times & \\ \rightarrow & \mathrm{O}_3 = \mathrm{I}_2 \oplus \mathrm{I}_4 \\ \rightarrow & \mathrm{O}_4 = \mathrm{I}_2 \oplus \mathrm{I}_3 \oplus \mathrm{I}_4 \\ \rightarrow & \mathrm{O}_5 = \mathrm{I}_3 \oplus \mathrm{I}_4 \\ \vdots & \end{matrix} \tag{2.9}$$

with the same generator matrix as (2.8) but with different received output symbols. Since a degree one output symbol does not exist, when these set of symbols are received the LT-BP decoder cannot determine any of the input symbols.

The ML decoder solves the system of linear equations in modulo-2 domain to reconstruct the input symbols I_i as

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{I}_1 \\ \mathrm{I}_2 \\ \mathrm{I}_3 \\ \mathrm{I}_4 \end{bmatrix} = \begin{bmatrix} \mathrm{O}_1 \\ \mathrm{O}_3 \\ \mathrm{O}_4 \\ \mathrm{O}_5 \end{bmatrix}. \tag{2.10}$$

The LT-ML decoder solves the set of linear equations by straightforward linear algebra techniques. The difference from the random X-OR coding is the low density, namely sparsity, of the matrix $\mathbf{A}$, so that more efficient solvers can be implemented.

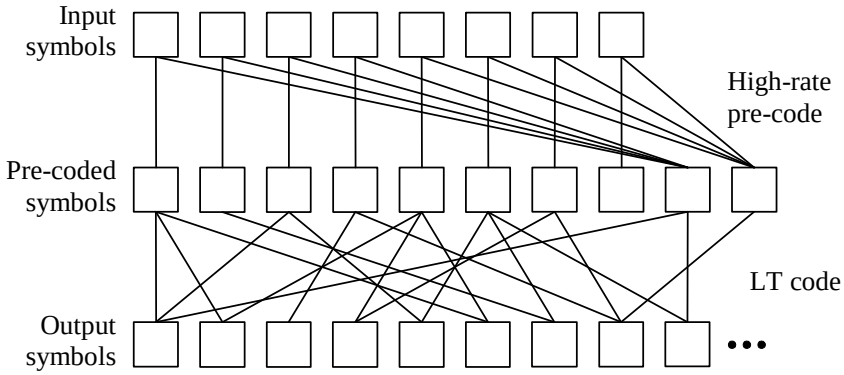


Figure 2.4: The representation of the Raptor encoder

2.5 Raptor Codes

Raptor codes [12] are the most recent practical realization of Fountain codes and are an extension of LT codes. They have two consecutive channel encoders, where the pre-code is a high rate FEC code and the outer-code is an LT code. The pre-code generates the pre-coded symbols from the input symbols and the LT code generates output symbols from these pre-coded symbols.

2.5.1 Encoder

The complexity of the LT codes is $O(k \ln(k))$ as explained in Section 2.4.2. The multiplicand $\ln(k)$ comes from the average degree $\bar{d}$. In order to make the complexity linear with k, Raptor codes use a constant average degree in the LT coding phase. The constant average degree reduces the performance of the LT code, but the high-rate pre-code compensates this reduction.

An example to the construction of Raptor encoded symbols are given in Figure 2.4. Initially, the high-rate pre-code generates $k/(1-\tilde{e})$ pre-coded symbols which are not transmitted but used in an intermediate step to generate the transmitted output symbols. Then, LT coding with constant average degree is

applied to the pre-coded symbols. The details of the chosen pre-code and degree distribution are given in [12].

2.5.2 Decoder

The Raptor decoder begins to operate when slightly more than k output symbols arrive. After the LT decoder, using the balls and bins example in Section 4.2.1, the average number of source packets that remain unconnected in the graph can be calculated as $e^{-\bar{d}} = e^{-3} \cong 0.05$, when $\bar{d} = 3$. Thus, when $k\,(1+\varepsilon)$ output symbols are received, 5% of the pre-coded symbols will remain undecoded. The pre-code is a perfect FEC code that can decode all input symbols when erasure rate is $\tilde{e}$ (the number of generated pre-codes was $k/\,(1-\tilde{e})$). When $\tilde{e}$ is chosen as 0.05 all of the input symbols can be decoded by the pre-code.

Instead of the simple pre-code we used in Figure 2.4, more sophisticated codes, such as LDPC codes [5], are used. Moreover, two stages of pre-codes are also defined such as the fully-specified Raptor FEC scheme defined in [16].

2.6 Video Coding and Streaming

2.6.1 History of Video Coding Standards

Video coding corresponds to the efficient representation, compression and transmission of image sequences. Studies on digital video processing have been virtually initiated by the studies on digital image processing in 1960s. Current video coding techniques partially use the image compression techniques.

First practically applicable video coding standard appeared in the early 90s named H.261 in 1990 as an ITU recommendation, which is described in [17]. The

video codec aims at facilitating videoconferencing and videophone over the integrated digital services network (ISDN). The bit rate of the video signal, together with audio, can be varied from 64 kbps to 1.92 Mbps, for two image formats CIF (360x180) and QCIF (180x90). With the introduction of efficient compression elements, such as inter-frame coding, motion compensation, discrete cosine transform (DCT), maximum delay of 150ms, ease of hardware implementation etc., H.261 standard is considered as the basis of latter video codecs.

In 1992, MPEG-1 was approved as an ISO video and audio coding standard whose details are given in [18]. The codec is proposed to progressively compress VHS quality video down to 1.5 Mbps for Video Compact Disc (VCD) storage. The codec does not standardize the compression mode, instead it specifies the coded bit stream and the decoder. In order to allow random access, periodic intra frames that are self decodable are introduced. Compared to H.261, the delay requirement of MPEG-1 is relaxed to 1 sec, because it does not aim bidirectional interactivity, instead unidirectional interactivity.

MPEG-2 [35] is a generic audio-video codec designed for two purposes; first, broadcast of TV signals at high bit rates, second, media storage on the Digital Versatile Discs (DVD). The video coding part, which is also known as H.262, is jointly developed by ISO and ITU-T teams, and the first standard appeared on 1994. Compared to MPEG-1, MPEG-2 supports a wider range of bit rates and resolutions; moreover it supports efficient tools for interlaced video. One of its most important specialities is the utilization of two different containers for the encoded video; first one is the transport stream that defines a communication protocol for data multiplexing and loss recovery for unreliable transmission, whereas second one is the program stream that is designed for storage in reliable media, such as DVDs.

Later, in 1996, another video compression technique H.263 is proposed targeting visual telephony, such as video conference similar to H.261, which is described

in [36]. The compressed data is aimed to be used on low bit rate networks such as ISDN and wireless networks. H.263 has a better performance than its prior H.261 with little additional complexity. An advanced version of H.263 is proposed in 1998 as H.263+, described in [37], with additional features, one of the most important being the introduction of scalability. Three modes of scalability are proposed; temporal, SNR and spatial scalability, which aim to improve the delivery of video information in error-prone and lossy packet networks.

The next codec, developed jointly by ITU-VCEG and MPEG teams, and completed in 2003, is the H.264 standard which is described in [24]. The codec has the largest area of application among all previous codecs, so that it addresses all video telephony, storage, broadcast and streaming applications. The codec has the most advanced compression techniques and most flexible structure together with network friendly representation of the compressed data, so that compared to MPEG-2 the bit rate is reduced up to by half for high resolutions. H.264/AVC has two fundamental components; the Video Coding Layer (VCL) and the Network Abstraction Layer (NAL). The VCL is designed to efficiently compress the video, and the NAL is designed to format the compressed video and insert appropriate header information for transport layers or storage media. The scalable extension of the codec, which is described in [38], is completed in 2007. Currently, H.264/AVC is considered to be the state-of-the-art video coding system.

In order to further improve the visual experience, H.264 video codec is being extended to include multi-view video, pioneered by the 3D Audio Visual (3DAV) group under MPEG committee [39], [40]. There are individual studies on the extension of H.264 to multi-view video, such as [41], [42] and [43]. In our work we also focused on a subset of multi-view video coding; stereoscopic video coding, and utilized the codec in [43].

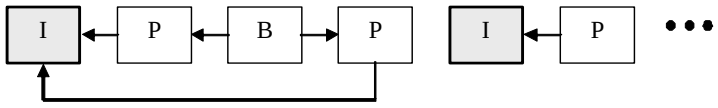


Figure 2.5: An example reference structure in video coding

2.6.2 Video Compression Principles

Current video compression techniques achieve significant bit rate savings by combining still image compression and motion compensation. The motivation behind the use of image and video compression is the huge amount of raw data that emanates after the video capture. For example a raw video of resolution 1280x720 at 30 fps requires nearly 650Mbps which is unattainable in current technology for commercial use. The most recent compression standard H.264 can reduce the bit rate down to 10 Mbps with a reasonable quality.

A generic referencing structure of a video codec is given in Figure 2.5. The frames denoted with I are intra coded frames, the other ones, P and B, are inter coded frames. Intra-coding refers to self-decodable frames that are still image coded. Inter coding refers to compressing the frame of interest dependent on the previous or subsequent frames. When a frame is compressed referring only to previous frames it is denoted with the letter P, and when it is coded bi-directionally referring to both previous and subsequent frames it is denoted with the letter B.

In the following sections we briefly describe the two different modes of video coding, namely intra and inter coding.

Intra Coding

Intra coding uses the principles of still image compression techniques such as the ones used in JPEG [19]. It starts with RGB yo YUV conversion, a simple linear

conversion, which removes the dependency in color representation of RGB. After the YUV conversion, 2-D DCT is applied on block-by-block basis (generally 8x8 or 16x16) in order to reduce the correlation among the image pixels. After DCT and quantization, zig-zag scan and entropy coding are used to apply the final compression steps to the chosen frame.

Intra coded frames are introduced on a regular basis in order to allow fast synchronization and prevention of the propagation of losses. The period of intra frames are subject to optimization for different types of environments. There is a significant tradeoff between bit rate, quality and error robustness on the period of intra frames. Since the entire frames between two intra-frames are eventually dependent on them, intra-frames are generally given the highest priority during lossy transmission.

Inter Coding

In the current block based video coding systems, inter coded frames are the main factor of bandwidth saving. Inter coding refers to the compression of frames by removing the temporal redundancy, whereas intra-coding removes the spatial redundancy. The simplest way of removing the temporal redundancy is by just sending the residual (the difference) between the current and the previous frame, however more compression can be achieved by means of estimating and compensating the motion between the frames. Block based motion estimation and compensation is considered to be the most practical and efficient technique for this purpose.

In Figure 2.6, block based motion estimation is presented. The aim of block based motion estimation is to find a motion vector that relates best matching block from the n^{th} frame to the block of interest in the $(n+1)^{th}$ frame. The best matching vector is searched in a limited search region to decrease the complexity

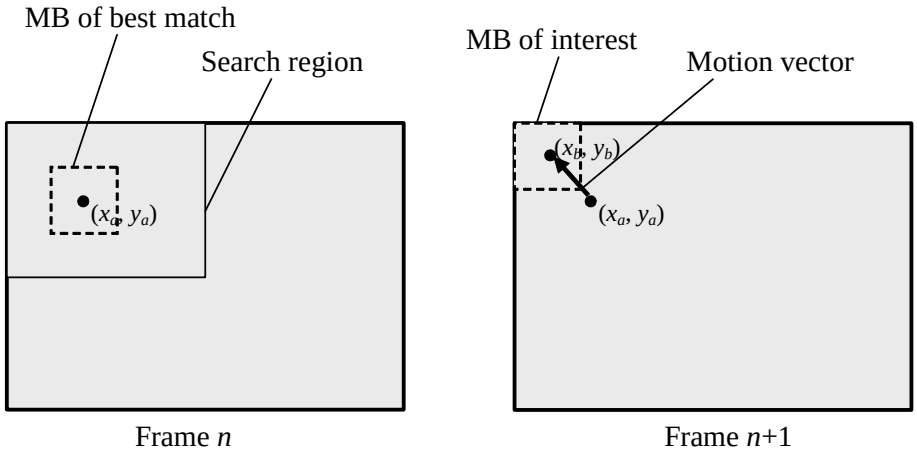


Figure 2.6: Block-based motion estimation in inter-frames

of the method. In Figure 2.6, an artificial motion vector is demonstrated that heads from the point (x_a, y_a) to (x_b, y_b). This process is repeated for each block in the current frame. After all motion vectors are determined, a new motion compensated frame from the $(n+1)^{th}$ frame is formed using the best matching blocks of the n^{th} frame. A residual frame is constructed by subtracting the motion compensated frame from the n^{th} frame, and the residual is still image coded. Then, the still image coded residual together with the motion vectors are sent or stored as the compressed data of frame $n+1$.

2.6.3 Video Streaming Principles

The term streaming appeared in the multimedia area following the possibility of on-demand or real-time video over the Internet. It refers to the display of the multimedia content at the end-user while it is being transmitted from the server.

In the lossy channels, video streaming is a complicated task where one has to deal with many problems, such as latency, packet loss detection and recovery, and bit rate budget-quality tradeoff, etc.. One of the most widely used streaming protocols is Hypertext Transfer Protocol (HTTP) over TCP [44], which is

a good choice to avoid firewall issues. Another protocol similar to HTTP is Real-Time Streaming Protocol (RTSP) [45] which is being adopted to many systems. Real-time Transport Protocol (RTP) [46] is another streaming protocol which is essentially a packet format that adds a timestamp, a sequence number, a contributing source identifier, and a payload type and format on top of an ordinary TCP or User Datagram Protocol (UDP) packet. In these protocols, there are transmission rate control, congestion control, error control and packet loss recovery, due to the properties of TCP. However this may cause latency and inefficiency in bandwidth utilization.

RTP over UDP is another alternative to these systems where packets are sent once and there is no means of an integrated transmission rate control or error control. In these systems transmission rate and error controls can be performed by the application layer which brings flexibility. The best candidate for application layer error control in these systems is FEC, where few extra packets are inserted to recover any lost packets. Thus, RTP over UDP provides flexibility, lower latency and bandwidth utilization, which is well-suited for video streaming.

2.7 Multi-view Video Coding and Streaming

Multi-view video coding is an approach for the efficient compression and transmission of image sequences captured from more than one cameras. Due to the increase in the number of captured views, eventually, the raw bit rate increases linearly with the number of cameras. This fact necessitates the exploitation of spatial dependencies among the views, besides the exploitation of temporal dependencies among frames of one view. Standardization of multiview video is under process as a study in ISO/IEC [39], [40]. Other than the standardization effort, there are individual studies on multi-view coding [41], [42] and [43].

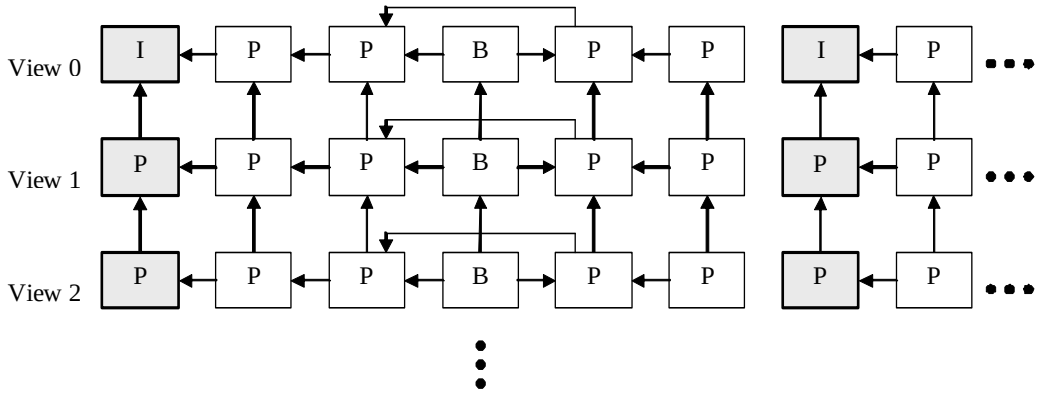


Figure 2.7: An example reference structure in multi-view video coding

In Figure 2.7, a referencing structure for multi-view coding is presented. Many different referencing structures can be constructed according to the need of application. Generally, one of the views is coded independent of the others (View 0 in Figure 2.7) in order to provide compatibility to monoscopic decoders. Other views are coded dependent on their prior. In such a scheme, if one tries to group the frames according to their importance, the easiest way is to use the referencing structure. The priority of the frames diminishes from top to bottom. Providing different protection to different frames can increase the quality of the decoded multi-view video significantly.

Streaming of multi-view coded video is another research topic. In [47], a streaming system that uses standard LAN connection is proposed. In [48], the multi-view coded video is transmitted over RTP/UDP. Another study is on the utilization of DCCP for streaming [49]. In [50], stereoscopic video is layered using data partitioning, but an FEC method specific to stereoscopic video is not used. The video data is segmented into three parts, however only the part containing motion and disparity compensated error residuals is protected with different channel code rates to observe changes experimentally, and simply, the significance of UEP over EEP is demonstrated.

2.8 Error Resilient Video Streaming with Fountain Codes

Fountain codes for error resilient video streaming have gained a significant interest in recent years. This is mainly caused by the extreme increase in the transmission bit rates due to widespread deployment of video streaming systems and consequent reliability issues. Fountain codes promise reliability with low complexity for this large amount of data.

A scalable video streaming system which uses Raptor codes using multiple media servers is presented in [51]. Different servers with different channel characteristics stream same scalable video content independent of the client. The authors considered delivery of two layers; base layer and enhancement layer. They proposed two schemes: one with optimal rate allocation with complete knowledge of network condition, and the other one with an heuristic approach. A unicast video streaming method with rateless codes is proposed in [52]. Contrary to the idea of rateless codes, an acknowledgement message is sent from the receiver for every source block. Thus, redundant output symbols arrive to the receiver before the acknowledgement arrives to the sender. The authors derive strategies to minimize the overhead under lossy transmission. A multi-layered video coding scheme with unequal error protection with rateless codes are proposed in [53]. A layer-aware FEC (L-FEC) system which applies different protection to the defined dependent layers by a combined FEC scheme is proposed. FEC is applied according to the dependency of the layers, so that the FEC of current layer is created using all layers up to current layer. Fountain codes also attracted interest for video transmission in mobile and ad hoc networks. In [54], the authors used Scalable Video Coding (SVC) for rate adaptation at the peers of a mobile network. They also used application layer FEC (AL-FEC) for encoding the source blocks separately as in [51]. The simultaneous

utilization of SVC and AL-FEC allows distributed reception of video between the peers. Another work about rateless codes in mobile networks is presented in [55]. The authors proposed a new UEP structure for Raptor codes where they apply different code rates in the pre-coding stage to the video content with different priorities. The authors proposed the scheme for scalable video delivery in mobile networks. In [56], authors propose layered video streaming system, where the most important layers received highest protection. The proposed system for SVC is suitable for devices with limited computation capability. Network coding with video streaming, which targets the case of several servers and clients, is another novel research area. In [57], the authors proposed an efficient video transmission scheme by combining Raptor codes and network coding. Rateless codes are also used for energy-efficient video streaming studies, such as in [58].

2.9 Improving the State-of-the-Art Techniques

Fountain codes, especially Raptor codes, are the state-of-the-art error correction scheme for PEC. They appear in the most recent standards for multimedia broadcast, such as Digital Video Broadcast for Hand-held (DVB-H) [14] and 3^{rd} Generation Partnership Project (3GPP) [15]. In our work, we analyze and model the performance of LT and Raptor codes which can be used in these multimedia broadcast systems for ease of performance monitoring and optimization of quality.

Studies on the state-of-the-art multi-view video codecs are carried on under ISO/IEC [39], [40]. Integration of this system with an FEC scheme is one of the possible ways to improve the standard for handling the lossy transmission. Moreover, UEP techniques can also be applied since multi-view video has a high number of data groups with different loss sensitivities. In this sense, we study on the transmission of layered stereoscopic video with FEC.

Chapter 3

Error Correction in Stereoscopic Video Streaming

3.1 Motivation

Video streaming is a challenging problem when the transmission medium is lossy. Especially, when the dimension of the video increases the problem becomes more complicated. We aim at optimizing the visual quality of stereoscopic video under lossy transmission. In order to achieve this, we suggest a method for modeling the end-to-end rate distortion characteristics of video streaming. Specifically, we propose a system that models the RD curve of video encoder and performance of channel codec to jointly derive the optimal encoder bit rates and unequal error protection (UEP) rates specific to the layered stereoscopic video streaming.

In Section 3.2, we present common error resiliency tools and demonstrate that in order to achieve best quality, FEC is required for lossy transmission. In Section 3.3, we give an overview of the stereoscopic streaming system. In Section 3.4, we provide the calculation of the RD curve for the three layers of stereoscopic video and demonstrate the accuracy of the model by using it in video

encoder distortion minimization. In Section 3.5, we provide the performance modeling of Raptor codes that is further detailed in Section 4.5.4. In Section 3.6, we calculate the approximate distortion in video quality when units of video data are lost during transmission. Finally, in Section 3.7, we perform end-to-end distortion minimization to obtain optimal video encoder bit rates and UEP bit rates, and provide simulation results under various transmission scenarios.

3.2 Basics of Error Resiliency Techniques for Video Streaming

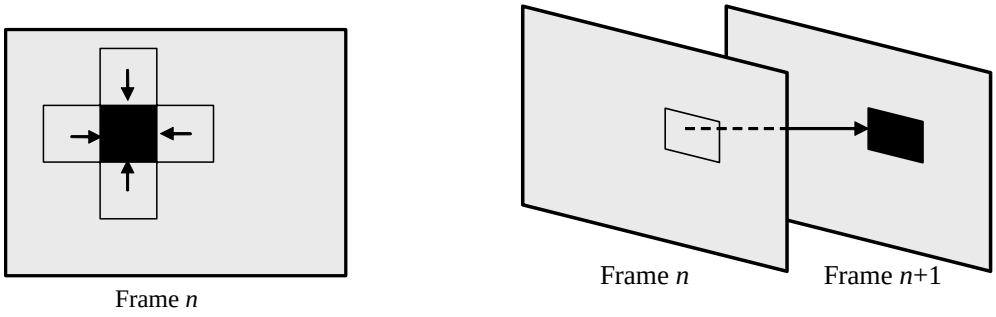
Error resiliency in video streaming refers to the techniques that aim at improving the quality of the video in case of packet losses during the transmission. In order to define the quality of the monoscopic video, generally peak signal to noise ratio (PSNR) measure, which is defined as

$$PSNR = 10 \log \left(\frac{\left(2^B - 1\right)^2}{MSE} \right), \tag{3.1}$$

is used, where B is the number of bits per pixel (generally 8), and MSE is mean squared error between the original and reconstructed T images of size m by n. The MSE is defined as

$$MSE = \frac{1}{T} \sum_{t=0}^{T-1} \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \left\| I_{orig}\left(i,j,t\right) - I_{rec}\left(i,j,t\right) \right\|^2, \tag{3.2}$$

where $I_{orig}\left(i,j,t\right)$ and $I_{rec}\left(i,j,t\right)$ are the pixel values at the location (i,j) of the t^{th} frame of original and reconstructed frames, respectively. This PSNR measure will be used in Sections 3.2.1 to 3.2.3, in order to demonstrate the effect of the error resiliency schemes in the case of monoscopic video. The PSNR measure that we use for the quality measurement of stereoscopic video is defined in (3.28).



(a) (b)

Figure 3.1: The representation of two basic error concealment techniques, (a) spatial, (b) temporal

3.2.1 Error Concealment

One of the most commonly used error resiliency techniques is error concealment. In this technique, the lost packet, namely some part of the image in the sequence, is replaced with an accurate representation from the neighboring received parts of the video. In Figure 3.1, two basic error concealment techniques are presented: spatial and temporal. The black block represents the lost part of the frame. In Figure 3.1(a), it is concealed by the average of neighboring blocks, whereas in Figure 3.1(b) concealed by the co-located block in the previous frame. There are many studies on advanced error concealment techniques such as described in [59], [60], [61].

The effect of error concealment on the quality of received video is significant. In Figure 3.2, the reconstructed video quality measures after %1, %3 and %5 losses with error concealment and the reconstructed video quality after %1 loss without error concealment are demonstrated with the reference software of H.264/AVC given in [62] for the 'Rena' video described in Section 3.7.2. Even at %1 loss, the quality of the video decreases significantly when error concealment is not used. The technique for error concealment is simply replacing the lost part of the video with the previous frame, namely temporal concealment. Error

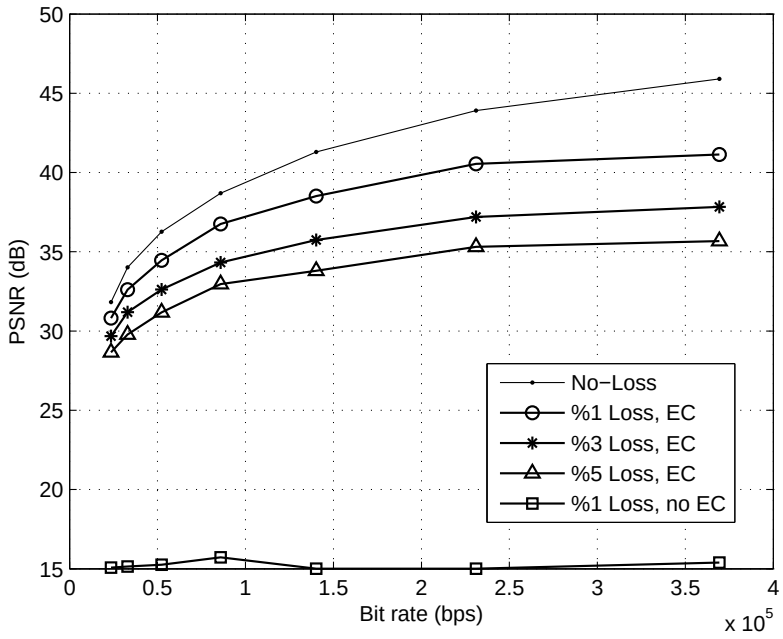


Figure 3.2: The results on the effect of error concealment on video quality

concealment is generally independent of the encoder, however for better gains the video has to be encoded in small sized slices (NAL units).

3.2.2 Flexible Macroblock Ordering

Flexible macroblock ordering (FMO) is another error resiliency tool that enables the division of image sequences into different regions called slice groups, which may consist of several slices (NAL units). Each slice group is encoded independently and this yields more efficient error concealment by exploiting spatial redundancy. When the slice groups are chosen so that no neighboring blocks remain in the same group, a lost slice can be more accurately concealed using the neighboring slices. There are many studies on different FMO schemes, such as the ones described in [21], [20], [63].

In Figure 3.3, we give the results when FMO is used on top of error concealment. The video is encoded and decoded with the reference software of H.264/AVC in [62] for the 'Rena' video described in Section 3.7.2. The chosen

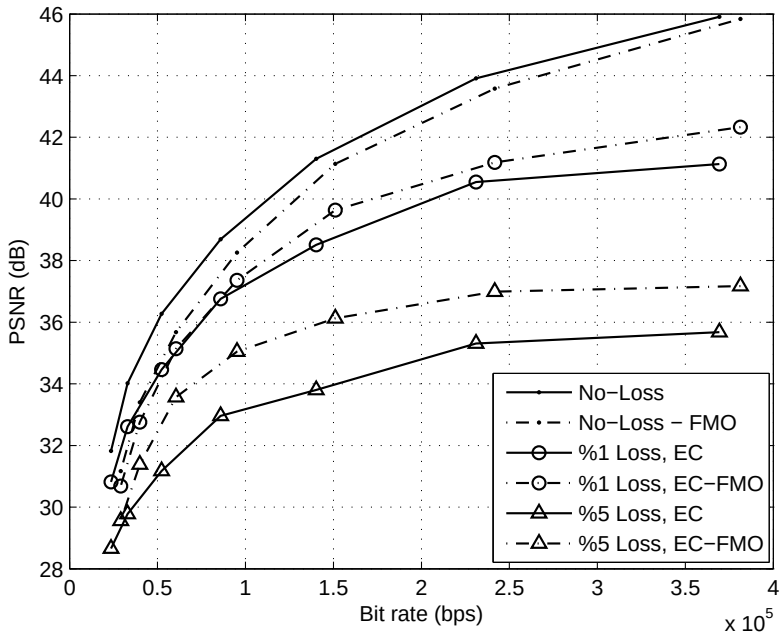


Figure 3.3: The results on the effect of FMO on video quality

mode of FMO is type 1, which corresponds to the checker-board pattern. This type is chosen due to simplicity and ease of use. Other types can yield better performance. The results of EC-FMO and only EC are provided in the same figure. When no loss occurs, the video quality with FMO is slightly less than that of the original. This is caused by the decrease in dependency and extra overhead when more than one slice groups are formed. In the cases of lossy transmission, when FMO is utilized, the quality of the reconstructed video increases especially for high loss rates.

3.2.3 Forward Error Correction

Forward error correction (FEC) is one of the most efficient and powerful techniques for error recovery. It corresponds to insertion of extra data to help the recovery of lost data. Assume that RS coding [4] is used as the FEC scheme and assume that the original input data length is k symbols and $n-k$ extra symbols are inserted to obtain a whole data stream of n symbols. At the decoder side,

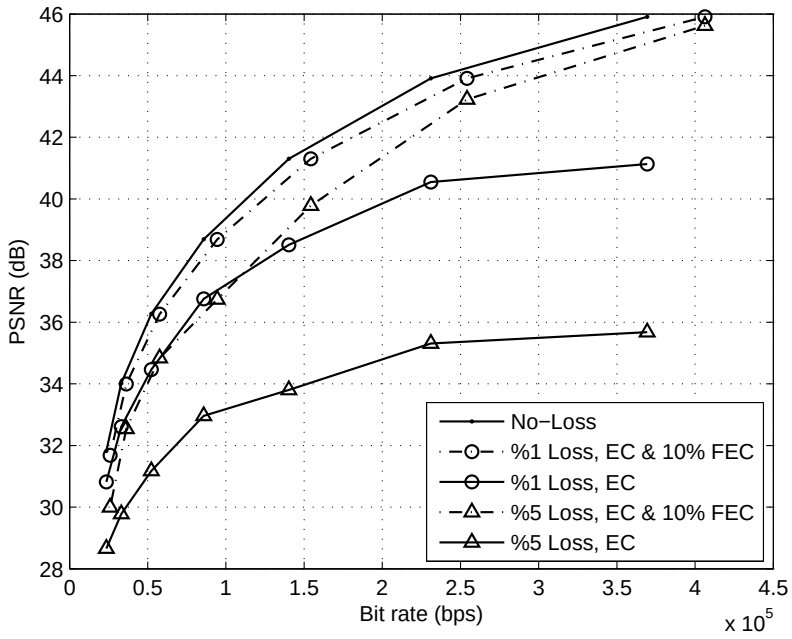


Figure 3.4: The results on the effect of FEC on video quality

the arrival of any k symbols out of n symbols ensure the recovery of the whole original symbols. Actually, this is the theoretical limit in an erasure channel that a maximum distance separable (MDS) code can achieve. RS codes operate at this limit, however they have complexity of $O(n^2)$. Raptor codes approach this limit with a lower complexity as described in Section 2.5. Video streaming with FEC codes is studied on various papers such as in [64], [65], [66], [67], [68], [69].

Raptor codes are well-suited to the video streaming application through packet networks, such as Internet. In case of video streaming, original symbols that are given as input to FEC are the slices, namely NAL units, which are composed of a fixed number of bytes. We present the effect of an ideal FEC scheme, such as RS codes, in Figure 3.4. In the simulations 10% extra FEC packets are sent together with the original packets for 1% and 5% losses. The effect of FEC on video quality is significant, especially for high bit rates where the number of packets is large.

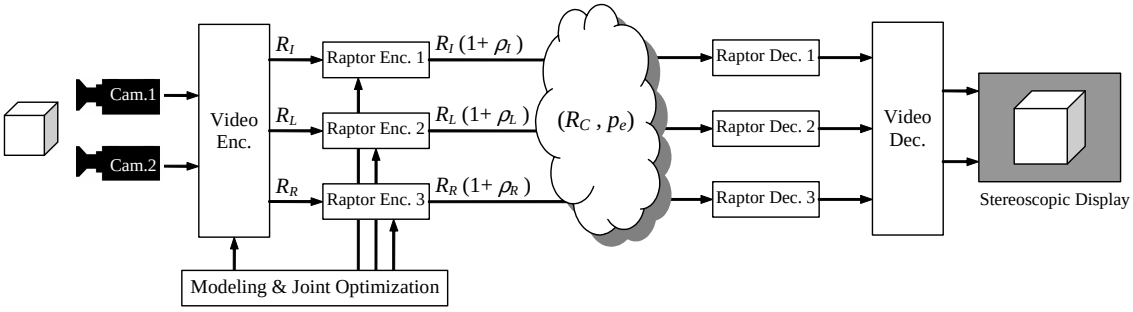


Figure 3.5: Overview of the stereoscopic streaming system

FEC schemes can be extended to provide unequal error protection to different parts of video data. After the partitioning of the video, different protection is applied to each priority, where the highest priority receives the highest error protection.

3.3 Overview of the Video Streaming System

An overview of our proposed stereoscopic streaming system is presented in Figure 3.5. Initially, the scene of interest has to be captured with two cameras to obtain the raw stereoscopic video data. The video capture process is not in the scope of our work, thus we use publicly available raw video sequences. We encode the raw stereoscopic video data with an H.264 based multi-view video encoder. We use the codec in stereoscopic mode and generate three layers which are denoted with the symbols I, L and R. I-frames are the intra-coded frames of the left view, L and R-frames are the inter-coded frames of the left view and right view. The video encoder can encode each layer with different quantization parameters, thus with different bit rates R_I, R_L and R_R. Due to lossy compression, the encoding process causes a distortion of D_e in the video quality. After the stereoscopic encoder, we apply FEC to each layer separately where we use Raptor codes as the FEC scheme. The channel of interest in our system is a packet erasure channel of loss rate p_e and the available bandwidth of the channel is R_C.

We apply different protection rates ρ_I, ρ_L and ρ_R to each layer, because they contribute differently to the video quality. After the lossy transmission, some of the packets are lost and Raptor decoder operates to recover the losses. However, some packets still may not be recovered and the loss of these packets causes a distortion of D_{loss} in the video quality. In this system, our goal is to obtain the optimal values of encoder bit rates R_I, R_L and R_R and protection rates ρ_I, ρ_L and ρ_R by minimizing the total distortion $D_{tot} \triangleq (D_e + D_{loss})$. In order to calculate the total distortion, we simply summed encoder distortion and transmission distortion. However, any other metric can be used, and the proposed system can still operate. In the following sections, we describe the methods for distortion minimization step by step.

3.4 Modeling the Rate-Distortion Curve of Stereoscopic Video

3.4.1 Layers of the Stereoscopic Video

We use the stereoscopic video codec presented in [43], because it has low complexity and simple decoding procedure. However, any multi-view video codec can be used for the system we propose. The referencing structure of the codec in [43] is given in Figure 3.6 where GOP size is set to 4. Let $\mathbf{I}_L, \mathbf{P}_L$ and $\mathbf{P}_R$ denote the set of I-frames of left view, P-frames of left views and P-frames of right views respectively. The set of frames can be written in open form as $\mathbf{I}_L = \{\mathrm{I}_{L1}, \mathrm{I}_{L5}, ...\}$, $\mathbf{P}_L = \{\mathrm{P}_{L2}, \mathrm{P}_{L3}, ...\}$, $\mathbf{P}_R = \{\mathrm{P}_{R1}, \mathrm{P}_{R2}, ...\}$, where L and R indicate the frames of left and right video.

Although this coding scheme is not layered, frames are not equal in importance. We can classify the frames according to their contribution to the overall

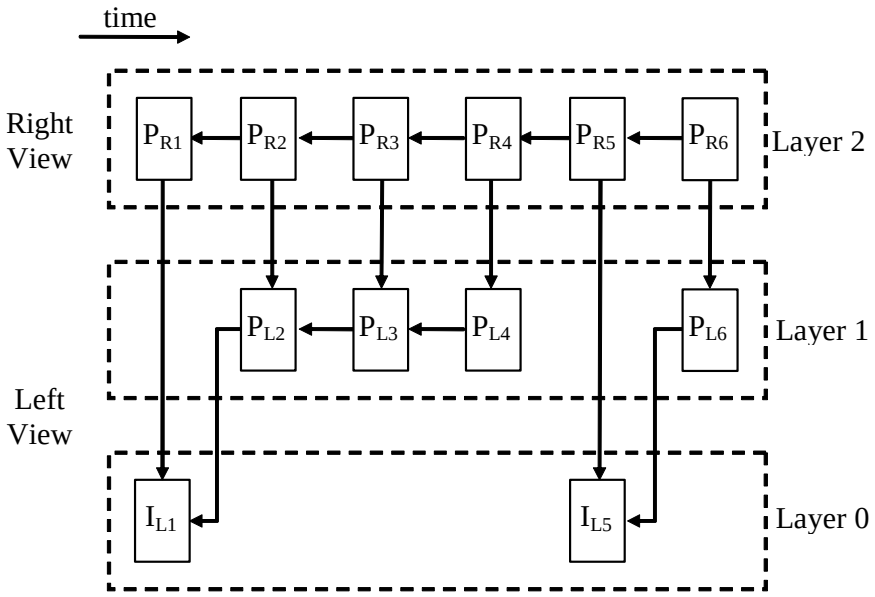


Figure 3.6: The layers of stereoscopic video and reference structure

quality and use them as layers of the video. Since losing an I-frame causes large distortions due to motion / disparity compensation and error propagation, I-frames should be protected the most. Among P-frames, left frames are more important since they are referred by both left and right frames. According to this prioritization of the frames, we form three layers as shown in Figure 3.6. Layers can be coded with different quality (bit rate) by using either spatial scaling [70] or quantization. In this work, we use quantization parameter to adjust the quality of different layers.

3.4.2 RD Curve Models for the Layers of Stereoscopic Video

RD curve is one of the widely used tools to adjust the quality of the video for a given bit rate in the encoder. The RD curves used in this work are specific to the utilized stereoscopic video encoder and do not yield the information theoretic bounds that are obtained with the rate-distortion theory. In [71], a simple

analytical RD curve model that can accurately approximate a wide range of monoscopic video sequences is presented. The model in [71] has the form

$$D_e(R) = \frac{\theta}{R - R_0} + D_0 \ , \tag{3.3}$$

where $D_e(R)$ is the mean-squared error (MSE) at the video encoder output at the encoding rate of R bits/sec. There are 3 parameters to be solved. These are θ, R_0 and D_0. The parameters R_0 and D_0 do not correspond to any rate or distortion values and they are not initial values. At least three samples of the RD curve are required to solve for the parameters.

In our work, we extended (3.3) for the layers of stereoscopic video and handled the interdependency among the layers. The primary layer is Layer 0 (I-frame) which consists of intra frames and it does not depend on any previous frame. Thus, the distortion of Layer 0 only depends on the encoder bit rate of Layer 0. We model its RD curve using the same framework as in (3.3) and set the model as

$$D_e^I(R_I) = \frac{\theta_I}{R_I - R_{0I}} + D_{0I} \ , \tag{3.4}$$

where $D_e^I(R_I)$ is the MSE coming from Layer 0 when Layer 0 is allocated a rate of R_I bits/sec. The model parameters, which have to be solved, are θ_I, R_{0I} and D_{0I}.

The next layer is Layer 1 whose frames are coded dependent on previous frames of Layer 1 and Layer 0. Thus, the distortion of Layer 1 depends on the encoder bit rates of Layer 1 and Layer 0. We modify the model in (3.3) to handle this dependency as

$$D_e^L(R_L, R_I) = \frac{\theta_L}{R_L + c_1 R_I - R_{0L}} + D_{0L} \ , \tag{3.5}$$

where $D_e^L(R_L, R_I)$ is the MSE coming from Layer 1 when Layer 1 and Layer 0 are allocated the rates of R_L and R_I bits/sec, respectively. The model parameters are θ_L, c_1, R_{0L} and D_{0L} which also have to be solved. The term $c_1 R_I$ in the denominator is inserted to handle the dependency of the distortion of Layer 1 to Layer 0 where the encoder bit rate of Layer 0 is weighted with the parameter c_1.

The final layer is Layer 2 whose frames are coded dependent on previous frames of Layers 2, 1 and 0. Thus, the encoder distortion of Layer 2 depends on the encoder bit rates of all layers. We modify the model in (3.3) to handle this dependency as

$$D_e^R(R_R, R_L, R_I) = \frac{\theta_R}{R_R + c_2 R_I + c_3 R_L - R_{0R}} + D_{0R} \ , \tag{3.6}$$

where $D_e^R(R_R, R_L, R_I)$ is the MSE coming from Layer 2 when Layers 2, 1 and 0 are allocated the rates of R_R, R_L and R_I bits/sec, respectively. The model parameters are θ_R, c_2, c_3, R_{0R} and D_{0R}, which also must be solved. The terms $c_2 R_I$ and $c_3 R_L$ in the denominator are inserted to handle the dependency of Layer 2 to Layer 0 and Layer 1, where the encoder bit rate of Layer 0 and Layer 1 are weighted with parameters c_2 and c_3.

3.4.3 Results

In order to construct the RD curve models of stereoscopic videos, i.e., to obtain the model parameters, we used curve fitting tools. In our work, we used the stereoscopic videos 'Rena' and 'Soccer' explained in Section 3.7 and obtained the RD curve models of these videos for the analytical models in (3.4) to (3.6). We

Table 3.1: Encoder RD curve parameters for the 'Rena' and 'Soccer' videos

'Rena' Video	Layer 0			θ_I	R_{0I}	D_{0I}
				1.605e+011	6050	-289860
	Layer 1		c_1	θ_L	R_{0L}	D_{0L}
			0.616	3.483e+013	51858	6142922
	Layer 2	c_2	c_3	θ_R	R_{0R}	D_{0R}
		0.308	0.086	4.535e+013	50000	4056654
'Soccer' Video	Layer 0			θ_I	R_{0I}	D_{0I}
				2.978e+011	10249	120330
	Layer 1		c_1	θ_L	R_{0L}	D_{0L}
			0.456	1.513e+014	-23018	2209000
	Layer 2	c_2	c_3	θ_R	R_{0R}	D_{0R}
		0.333	0.235	1.496e+014	19482	6003200

used a general purpose non-linear curve fitting tool which uses the Levenberg-Marquardt method with line search [72]. Before the curve fitting operation we obtained many RD curve samples of the video by sweeping the quantization parameters of each layer from low to high quality. We obtained more RD samples than required in order to be able to observe the curve fitting performance. Then, we chose some of the RD samples and inserted into the curve fitting tool. The resulting analytical model parameters of the curve fit process are given in Table 3.1 for the chosen videos. The parameters are in accordance with the properties of the videos. 'Rena' has a static background with moving objects and 'Soccer' has camera motion. Since the 'Soccer' video has camera motion, while encoding a right frame, correlation with the current left frame can be more than the previous right frame. This shows why the c_3 parameter of Layer 2 of the 'Soccer' video is high when compared with the results of the 'Rena' video.

In Figures 3.7 to 3.12, we present the results of analytical modeling of the RD curves. In Figures 3.7 and 3.8, we give the results for Layer 0 where the analytical models are constructed using the model in (3.4) with the corresponding parameters from Table 3.1. The RD samples correspond to the actual RD values obtained from the video encoder before the curve fitting process. Later, the

results for Layer 1 are presented in Figures 3.9 and 3.10 and those of Layer 2 are presented in Figures 3.11 and 3.12. In the figures for Layers 1 and 2, we present two cross-sections of the RD curves. The cross sections are obtained by fixing the encoder bit rates of the layers other than the corresponding layer of interest. The average difference between analytical models and RD samples for the 'Rena' video are 3.62%, 7.60% and 9.19% for Layers 0, 1 and 2 respectively, and those of the 'Soccer' video are 1.00%, 5.87% and 8.89%. Thus, for both of the videos, which have different characteristics, satisfactory results are achieved where the analytical model approximates the RD samples accurately.

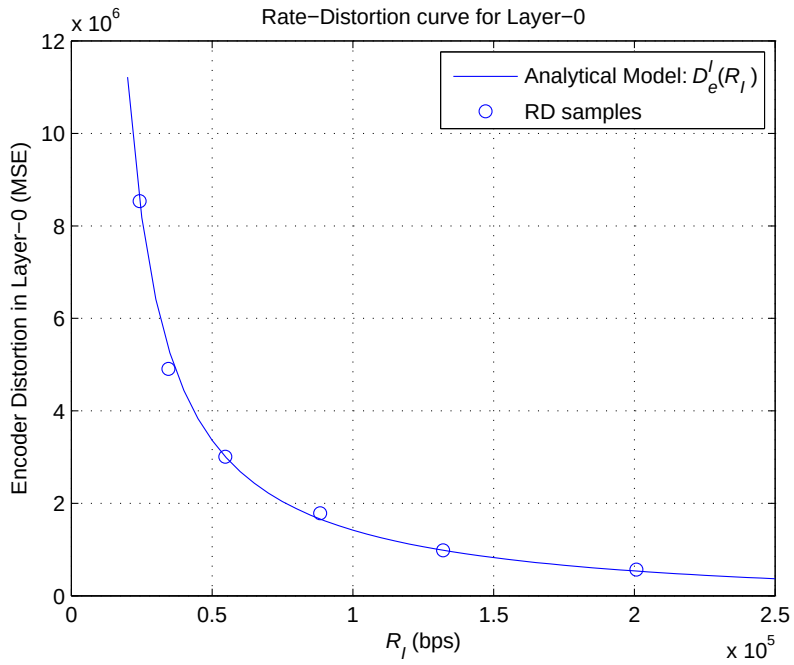


Figure 3.7: The RD curve for Layer 0 of the 'Rena' video

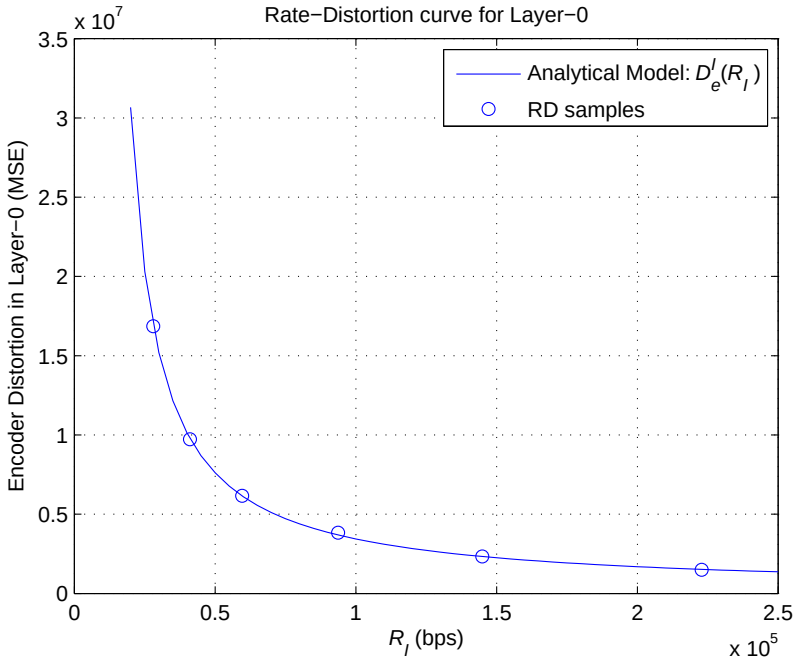


Figure 3.8: The RD curve for Layer 0 of the 'Soccer' video

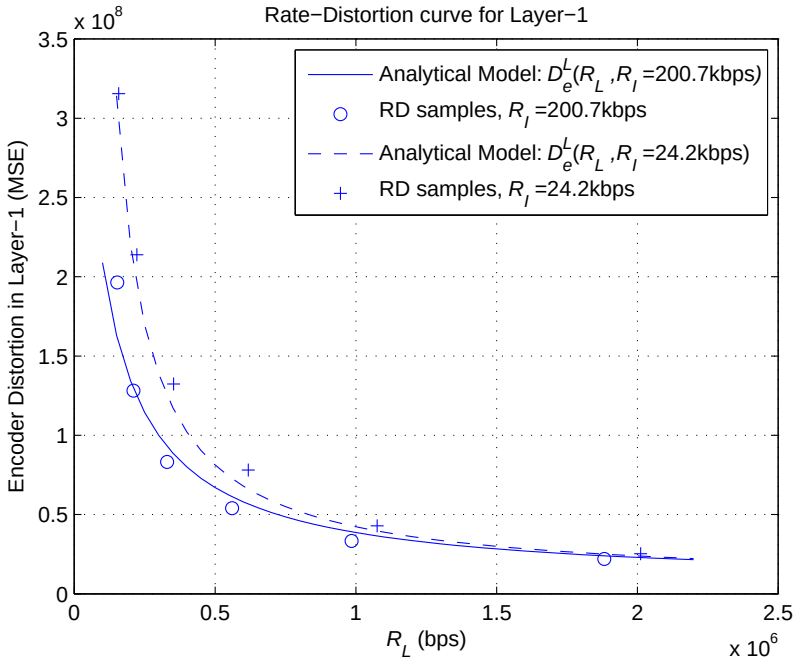


Figure 3.9: The RD curve for Layer 1 of the 'Rena' video

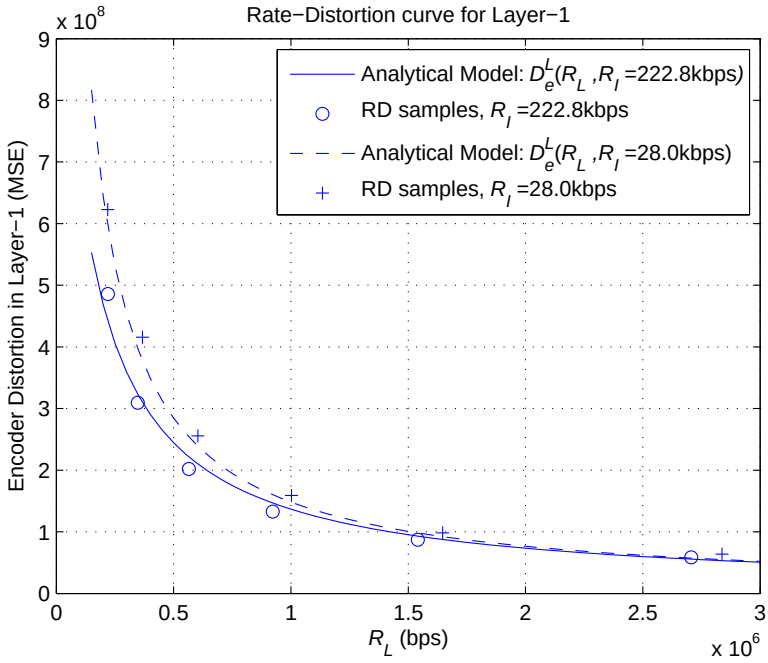


Figure 3.10: The RD curve for Layer 1 of the 'Soccer' video

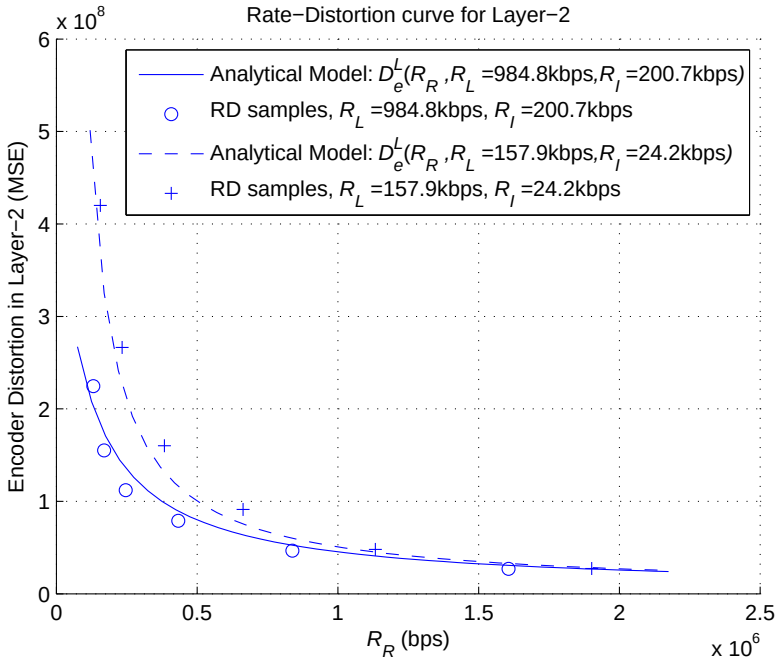


Figure 3.11: The RD curve for Layer 2 of the 'Rena' video

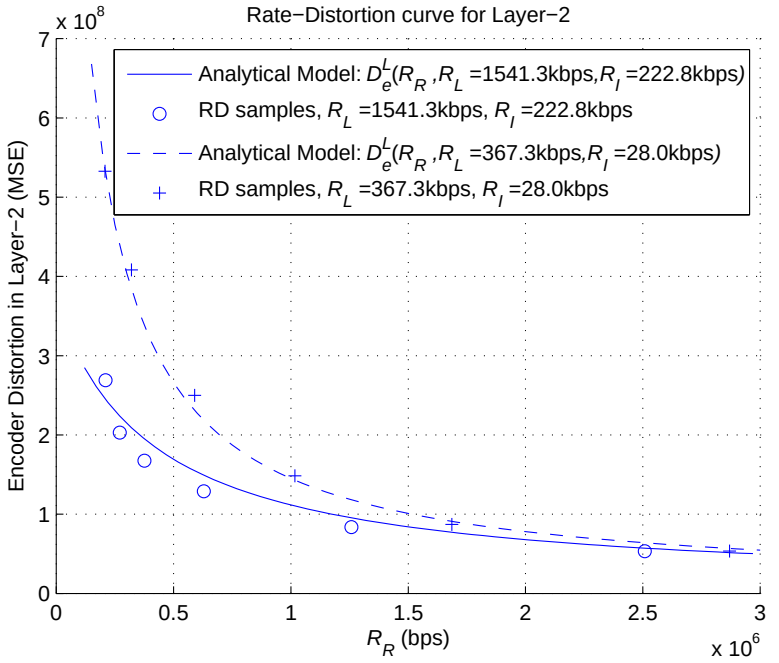


Figure 3.12: The RD curve for Layer 2 of the 'Soccer' video

3.4.4 Optimization on Encoder RD curve

In this part we use the previously derived RD curve models to obtain the optimal values of bit rates that minimize the distortion at the encoder output. This method does not relate to the end-to-end distortion minimization, but it can be used to obtain best quality compression for a total bit rate. The results will also demonstrate the accuracy of the RD models. The total distortion model D_e^{ILR} can be written as

$$D_e^{ILR}(R_R, R_L, R_I) = D_e^I(R_I) + D_e^L(R_L, R_I) + D_e^R(R_R, R_L, R_I). \tag{3.7}$$

Using the analytical model of the RD curve, the optimal encoding rates for each of the layers can be calculated for a constant transmission bandwidth. The optimization is defined as

$$\begin{aligned} &\min_{(R_I, R_L, R_R)} D_e^{ILR}(R_R, R_L, R_I) \\ &\text{such that } \; R_I + R_L + R_R = R_C \,. \end{aligned} \tag{3.8}$$

In (3.8), R_C denotes the channel bandwidth. The solution of this optimization can be calculated with the Lagrange multiplier method as

$$\begin{aligned} &L(\lambda) = D_e^{ILR}(R_R, R_L, R_I) + \lambda(R_I + R_L + R_R - R_C) \\ &\frac{\partial L(\lambda)}{\partial R_I} = 0 \Rightarrow R_I = R_{I0} + \sqrt{\frac{\theta_I}{\lambda(1-c_1-c_2+c_1c_3)}} \\ &\frac{\partial L(\lambda)}{\partial R_L} = 0 \Rightarrow R_L = R_{L0} - c_1 R_I + \sqrt{\frac{\theta_L}{\lambda(1-c_3)}} \\ &\frac{\partial L(\lambda)}{\partial R_R} = 0 \Rightarrow R_R = R_{R0} - c_2 R_I - c_3 R_L + \sqrt{\frac{\theta_R}{\lambda}} \\ &\frac{\partial L(\lambda)}{\partial \lambda} = 0 \Rightarrow R_I + R_L + R_R - R_C = 0 \\ &\Rightarrow \lambda = \left(\frac{\sqrt{\theta_I(1-c_1-c_2+c_1c_3)} + \sqrt{\theta_L(1-c_3)} + \sqrt{\theta_R}}{R_C - (1-c_1-c_2+c_1c_3)R_{I0} - (1-c_3)R_{L0} - R_{R0}} \right)^2, \end{aligned} \tag{3.9}$$

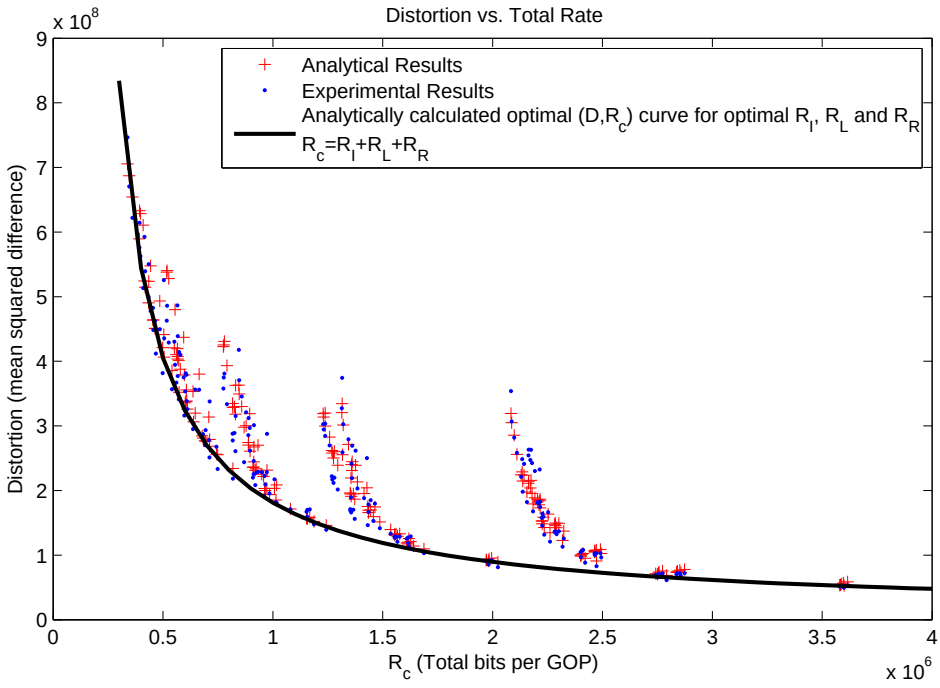


Figure 3.13: The RD curve for 3 layers

where θ_I, θ_L, θ_R, R_{I0}, R_{L0}, R_{R0}, c_1, c_2, c_3 were previously determined in Section 3.4.2. Thus, R_C remains as the only free variable. Let $D_e^{ILR}(R_C)$ be the solution of the minimization in (3.8). In Figure 3.13, the dots represent the experimental RD samples with different bit rates of layers that satisfy $R_I + R_L + R_R = R_C$, whereas the plus signs represent the analytically calculated RD samples for the same bit rates. The minimum distortion $D_e^{ILR}(R_C)$, which accurately approximates the convex hull of the RD samples of the stereo encoder, is represented with the black line.

3.5 Modeling the Performance of Raptor Codes

An abstraction of the performance of Raptor codes is required for optimization purposes in large scale systems. In Section 4.5.4, we define the modeling of systematic Raptor codes defined in [16]. The utilized model is

$$F(r,k,\rho) = \begin{cases} k - \frac{r}{1+\rho} & r < k \\ k \cdot 0.68 \cdot \frac{\rho}{1+\rho} \cdot (0.545)^{(r-k)} & r \geq k \end{cases} . \quad (3.10)$$

In (3.10), $F(k,r,\rho)$ is the analytical model of the average number of undecoded input symbols where k is the number of input symbols, r is the number of received output symbols, and $\rho = (n-k)/k$ is the parity ratio, where n is the number of output symbols generated in the encoder. Since the modeling results are given in Section 4.5.4, we do not present results on modeling in this section. The model given in (3.10) are used to obtain the optimal values of the video bit-rate and protection bit-rate.

3.6 Modeling the Error Propagation in Video

Video packets may get lost during the transmission and this loss propagates to the subsequent frames. In this section, we estimate the average distortion caused by the loss of a packet including the effect of its propagation.

3.6.1 Lossy Transmission

The channel of interest in our work is PEC as mentioned previously. During the transmission of stereoscopic video layers from PEC, NAL units are lost with probability p_e. In the remaining part of our work, for simplicity, X will represent the layer denotations I, L and R. As explained in in Section 3.4.1, we have three layers of video with source bit rate R_X which are Raptor encoded separately with inserted parity rate ρ_X. Let $N_i^X = R^X/N_{bits}$ be the number of input symbols where N_{bits} is the number of bits per NAL packet. Thus, $N_i^X(1+\rho_X)$ output symbols are created and transmitted for each layer. After lossy transmission, the number of received output symbols in Raptor decoder can be calculated as

$$N_r^X = N_i^X \left(1 + \rho_X\right) \left(1 - p_e\right), \tag{3.11}$$

where we use the average loss probability for simplified modeling purposes only. The experimental results in Section 3.7 reflect the actual distortions over lossy channels where a single packet is lost with probability p_e.

3.6.2 Reconstruction of Input Symbols in Raptor Decoder

After receiving N_r^X output symbols Raptor decoder operates to solve for the input symbols. We use the model of the performance curve of Raptor codes to obtain the average number of undecoded input symbols using (3.10). The average number of undecoded input symbols (the residual number of lost NAL units) can be calculated as

$$N_u^X = F\left(N_i^X, N_r^X, \rho_X\right). \tag{3.12}$$

3.6.3 Propagation of Lost NAL units in Stereoscopic Video Decoder

Due to the recursive structure of the video codec, the distortion of a NAL unit loss not only causes distortion in the corresponding frame but it also propagates to subsequent frames in the video. Initially, since each NAL unit contains a specific number of macroblocks (MBs), we estimate the distortion in a frame when a single MB is lost. The distortion is calculated after error concealment techniques, explained in Section 3.2.1, are applied for the lost MB. Then, we

calculate the average propagated distortion of a single MB and, consequently, a NAL unit.

In [71], a model for distortion propagation is proposed where the propagated error energy (distortion) at frame t after a loss at frame 0 is given as

$$\sigma_u^2(t) = \frac{\sigma_{u0}^2}{1+\gamma t}\,, \tag{3.13}$$

where σ_{u0}^2 is the average distortion per lost unit, and γ is the leakage factor which describes the efficiency of the loop filtering in the decoder to remove the introduced error ($0 < \gamma < 1$). We assume $\gamma \approx 0$ which results in worst case propagation where the distortion propagates equally to all subsequent frames ($\sigma_u^2(t) = \sigma_{u0}^2$). In the following sections, we calculate the propagated NAL unit loss distortion for each layer separately where we set MBs as the video unit.

NAL unit loss from Layer 0

The average distortion of spatial error concealment when a lost MB is concealed by the average of its neighboring MBs can be calculated as

$$\sigma_{I0}^2 = \frac{1}{N_{MB}^I} \sum_{k \in S_{MB}} \left[\sum_{x,y \in MB_k} \left(\mathrm{I}_\mathrm{I}(x,y,0) - \sum_{x',y' \in MB_k'} \mathrm{I}_\mathrm{I}(x',y',0)/N_k' \right)^2 \right], \tag{3.14}$$

where S_{MB}, MB_i, $S_{\mathrm{MB},i}$, N_i' and N_{MB}^I represent the set of macroblocks, the i^{th} macroblock, the set of i^{th} MB's neighbors, the number of neighbors of i^{th} MB and the number of MBs of Layer 0, respectively. $\mathrm{I}_\mathrm{I}(x,y,0)$ denotes the pixel in position (x,y) of the intra frame of Layer 0. Layer 0 consists of a single intra frame, thus only spatial error concealment can be used due to intra coding.

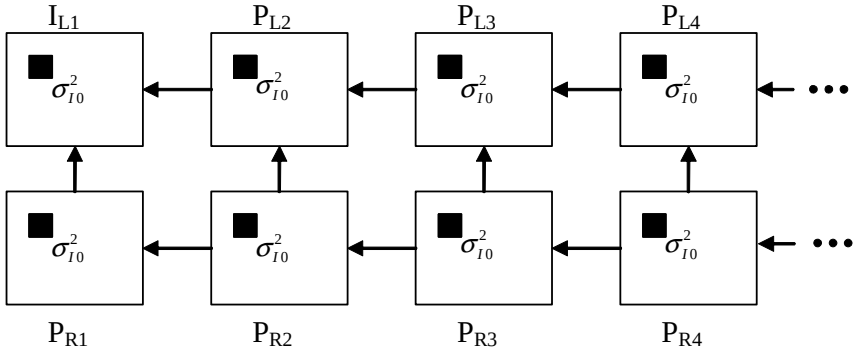


Figure 3.14: The propagation of a MB loss from I-frame

In Figure 3.14, the propagation of an MB loss in an I-frame is demonstrated. The black box in the frame $\mathrm{I_{L1}}$ represents a possible loss in the I-frame. The loss causes a distortion of σ_{I0}^2 as calculated in (3.14) for the frame $\mathrm{I_{L1}}$. The loss propagates to all subsequent frames with equal distortion on the average since both L-frames and R-frames refer initially to the I-frame. If we denote the GOP size as T, then the average of total propagated loss distortion when an MB is lost from Layer 0 can be calculated as

$$D^I_{\mathrm{MB}prop} = 2\mathrm{T}\sigma^2_{I0}\ . \tag{3.15}$$

In order to calculate the average distortion of losing a NAL unit from Layer 0 ($D^I_{\mathrm{NAL}loss}$), we have to calculate the average number of MBs in a NAL unit. Let N^I_{MB} denote the number of MBs in Layer 0. Then, $D^I_{\mathrm{NAL}loss}$ can be calculated as

$$D^I_{\mathrm{NAL}loss} = \left(\frac{N^I_{\mathrm{MB}}}{N^I_i}\right) \cdot D^I_{\mathrm{MB}prop}\ . \tag{3.16}$$

NAL unit Loss from Layer 1

The average distortion of temporal error concealment when a lost NAL unit is concealed from the previous frame of Layer 1 can be calculated as

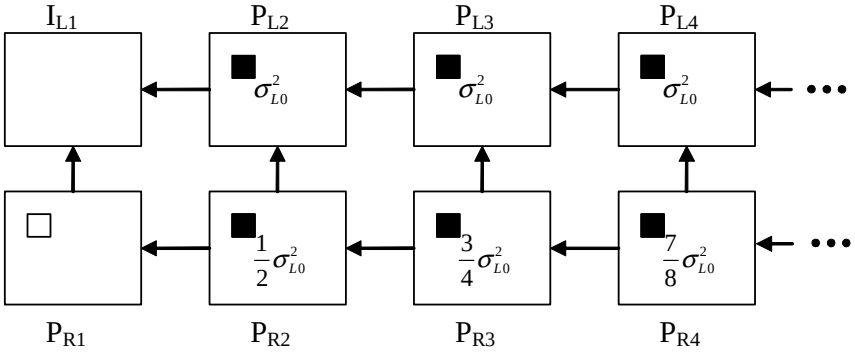


Figure 3.15: The propagation of a MB loss from L-frame

$$\sigma_{L0}^2 = \frac{\frac{1}{\mathrm{T}-1}\sum_{i=1}^{\mathrm{T}-1}\sum_{x,y}\left[\mathrm{I_L}\left(x,y,i\right)-\mathrm{I_L}\left(x,y,i-1\right)\right]^2}{N_{\mathrm{MB}}^{L}}\ , \tag{3.17}$$

where N_{MB}^{L} and T represent the number of MBs of Layer 1 and GOP size, respectively. $\mathrm{I_L}(x,y,i)$ denotes the pixel in position (x,y) of i^{th} frame of Layer 1. Layer 1 consists of predicted frames of left view. In our stereoscopic codec, we used temporal error concealment for Layer 1 as shown in Section 3.4.1.

In Figure 3.15, the propagation of an MB loss in an L-frame is demonstrated. The black box in the frame $\mathrm{P_{L2}}$ represents a possible loss in the L-frame. The loss causes a distortion of σ_{L0}^2 as calculated in (3.17) for the frame $\mathrm{P_{L2}}$. The loss propagates to all subsequent L-frames with equal distortion since each L-frame refers to the previous L-frame. Let m denote the frame index of loss in a GOP, then the average propagated loss to L-frames can be calculated as

$$\frac{1}{\mathrm{T}-1}\sum_{m=1}^{\mathrm{T}-1}\left(\mathrm{T}-m\right)\sigma_{L0}^2\ . \tag{3.18}$$

The MB loss also propagates to R-frames. However, R-frames not only refer to current L-frames but also previous R-frames. Due to this fact, the distortion in $\mathrm{P_{R2}}$ can be calculated as $\sigma_{L0}^2/2$ using the previous undistorted MB (white box in $\mathrm{P_{R1}}$). In the frame $\mathrm{P_{R3}}$ the propagated distortion can be calculated as

$(\sigma_{L0}^2/2 + \sigma_{L0}^2)/2 = \frac{3}{4}\sigma_{L0}^2$. In the subsequent frames the propagated distortion is calculated similarly as shown in Figure 3.15. The average of total propagated distortion in an R-frame caused by the loss of an L-frame MB can be calculated as

$$\frac{1}{\mathrm{T}-1}\sum_{m=1}^{\mathrm{T}-1}\sum_{n=1}^{\mathrm{T}-m}\left(\left(1-\frac{1}{2^n}\right)\sigma_{L0}^2\right). \tag{3.19}$$

Thus, the average of total propagated distortion when an MB is lost from Layer 1 can be calculated as

$$D_{\mathrm{MB}prop}^{L} = \frac{1}{\mathrm{T}-1}\sum_{m=0}^{\mathrm{T}-2}\sum_{n=0}^{m}\left(\left(2-\frac{1}{2^{n+1}}\right)\sigma_{L0}^2\right). \tag{3.20}$$

In order to calculate the average distortion of losing a NAL unit from Layer 1 ($D_{\mathrm{NAL}loss}^{L}$), we have to calculate the average number of MBs in a NAL unit. Let N_{MB}^{L} denote the number of MBs in Layer 1. Then, $D_{\mathrm{NAL}loss}^{L}$ can be calculated as

$$D_{\mathrm{NAL}loss}^{L} = \left(\frac{N_{\mathrm{MB}}^{L}}{N_i^{L}}\right)\cdot D_{\mathrm{MB}prop}^{L}\,. \tag{3.21}$$

NAL unit Loss from Layer 2

The average distortion of temporal error concealment when a lost NAL unit is concealed from the frames of Layers 2 and 1 can be calculated as

$$\sigma_{R0}^2 = \frac{\sum_{x,y}\left[\mathrm{I_L}\left(x,y,0\right)-\mathrm{I_R}\left(x,y,0\right)\right]^2}{\left(\mathrm{T}-1\right)N_{\mathrm{MB}}^{R}} + \frac{\sum_{i=1}^{\mathrm{T}-1}\sum_{x,y}\left[\left(\frac{\mathrm{I_R}(x,y,i-1)+\mathrm{I_L}(x,y,i)}{2}\right)-\mathrm{I_R}\left(x,y,i\right)\right]^2}{\left(\mathrm{T}-1\right)N_{\mathrm{MB}}^{R}}\,, \tag{3.22}$$

where N_{MB}^{R} and T represent the number of MBs of Layer 2 and GOP size, respectively. $\mathrm{I_R}(x,y,i)$ denotes the pixel in position (x,y) of i^{th} frame of Layer 2.

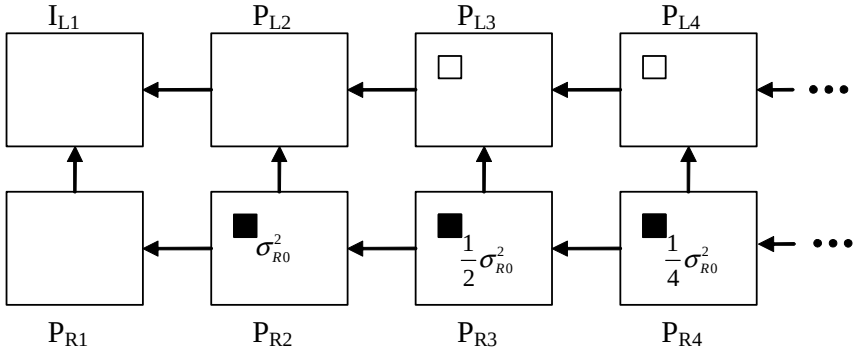


Figure 3.16: The propagation of a MB loss from R-frame

Layer 2 consists of predicted frames of right view. In our stereoscopic codec, we used temporal error concealment for Layer 2 where the frames are referred to previous Layer 2 and current Layer 1 frames, as described in Section 3.4.1.

In Figure 3.16, the propagation of an MB loss in an R-frame is demonstrated. The black box in the frame $\mathrm{P_{R2}}$ represents a possible loss in the R-frame. The loss in an R-frame propagates only to the subsequent R-frames. A loss in the frame $\mathrm{P_{R2}}$ creates a distortion of σ^2_{R0} as calculated in (3.22). In frame $\mathrm{P_{R3}}$, the propagation distortion can be calculated as $\sigma^2_{R0}/2$ using the undistorted MB in the L-frame (white box in $\mathrm{P_{L3}}$). In each of the following R-frames the propagated distortion is the half of the previous R-frame. Thus, the average of total propagated distortion when an MB is lost from Layer 2 can be calculated as

$$D^R_{\mathrm{MB}prop} = \sum_{m=0}^{\mathrm{T}-1} \frac{1}{\mathrm{T}} \sum_{n=0}^{m} \left(\frac{1}{2^n} \sigma^2_{R0} \right). \tag{3.23}$$

In order to calculate the average distortion of losing a NAL unit from Layer 2 ($D^R_{\mathrm{NAL}loss}$), we have to calculate the average number of MBs in a NAL unit. Let N^R_{MB} denote the number of MBs in Layer 2. Then, $D^R_{\mathrm{NAL}loss}$ can be calculated as

$$D^R_{\mathrm{NAL}loss} = \left(\frac{N^R_{\mathrm{MB}}}{N^R_i} \right) \cdot D^R_{\mathrm{MB}prop} \,. \tag{3.24}$$

3.6.4 Calculation of Residual Loss Distortion

In this part, we calculate the average transmission distortion after Raptor decoder and stereoscopic video decoder. Let

$$D_{loss}^{X}\left(R_X, \rho_X, p_e\right) = N_u\left(N_i^X, N_r^X, \rho_X\right) \cdot D_{\mathrm{NAL}loss}^{X} \tag{3.25}$$

denote the residual transmission distortion. In (3.25), we calculate D_{loss}^{X} by multiplying the number of undecoded input symbols with the average distortion of losing a NAL unit. We use the assumption that the NAL unit losses are uncorrelated which is met for low number of losses after the Raptor decoder. Thus, the accuracy of the model may reduce for high loss rates.

3.7 Distortion Minimization and Results

In the previous sections, we described how to model each part of an end-to-end stereoscopic streaming system. In this section, we use these models to derive optimal values of the encoder bit rates and protection rates by minimizing the end-to-end distortion. We present the minimization as

$$\begin{aligned} &\min_{(R_I, R_L, R_R, \rho_I, \rho_L, \rho_R)} D_{tot} \\ &\text{such that } \left(1+\rho_I\right) R_I + \left(1+\rho_L\right) R_L + \left(1+\rho_R\right) R_R = R_C \ , \end{aligned} \tag{3.26}$$

The minimization aims at obtaining the optimal encoder bit rates R_I, R_L and R_R, and optimal parity ratios ρ_I, ρ_L and ρ_R for given p_e and R_C. The constraint ensures that the final bit rate satisfies a total transmission bandwidth of R_C including both the encoder bit rates and protection data bit rates. In (3.27), we present the calculation of D_{tot} where $D_e^I\left(.\right)$, $D_e^L\left(.\right)$ and $D_e^R\left(.\right)$ are the encoder

distortions defined in (3.4), (3.5) and (3.6), and $D^I_{loss}(.)$, $D^L_{loss}(.)$ and $D^R_{loss}(.)$ are the residual loss distortions defined in (3.25).

Total distortion in left and right frames is weighted to handle the objective stereoscopic video quality as stated in [73], so that

$$\begin{aligned} D_{tot} = \tfrac{1}{3}\left[D^R_e(R_R, R_L, R_I) + D^R_{loss}(R_R, \rho_R, p_e)\right] \\ +\tfrac{2}{3}\left[D^I_e(R_I) + D^L_e(R_L, R_I) + D^I_{loss}(R_I, \rho_I, p_e) + D^L_{loss}(R_L, \rho_L, p_e)\right]. \end{aligned} \tag{3.27}$$

The weighting parameters in [73] are found by Least Squares Fitting of the subjective results with the distortion values. In [73], there are three parameters used for coding, number of layers, quantization parameter for left view and temporal scaling. In our codec, we only use quantization parameter for adjusting the bit rates. Although both codecs are not the same, they are both extensions of H.264 JM and JSVM softwares. So the distortions become similar if we consider only the case where quantization parameter is used to adjust the bit rates. Also, subjective results for our codec with temporal and spatial scaling can be found in [74], where we have similar results as given in [73].

We did not determine whether the optimization in 3.26 is convex, since it is easily processed with the tool that we used. In order to check the convexity, the Hessian matrix of D_{tot} with respect to the parameters R_I, R_L, R_R, ρ_I, ρ_L and ρ_R should be calculated. If the Hessian is positive semi-definite, then the optimization is convex.

3.7.1 Results on the Minimization of End-to-End Distortion

We solve the minimization in (3.26) by a general purpose minimization tool which uses sequential quadratic programming where the tool solves a quadratic programming at each iteration as described in [75]. In our work, we obtain the optimal encoder bit rates and parity ratios for $p_e \in \{0.03,\ 0.05,\ 0.1,\ 0.2\}$ and $R_C \in \{500,\ 750,\ 1000,\ 1500,\ 2000,\ 2500\ (\text{kbps})\}$ for 'Rena' video and $R_C \in \{1000,\ 1500,\ 2000,\ 2500,\ 3000,\ 3500\ (\text{kbps})\}$ for 'Soccer' video in order to demonstrate the performance of the proposed system. Thus we perform 24 optimizations per video using (3.26).

In Tables 3.2 and 3.3, the optimal encoder bit rates and protection rates for the proposed method are given for the 'Rena' and 'Soccer' stereoscopic videos for $p_e = 0.03, 0.05, 0.1, 0.2$. The encoder bit rates of the right view is lower than that of the left view; this is due to the unequal weighting in the total distortion expression in (3.27) and the higher priority of left view compared to right view. The protection rate of I-frame is the largest due to low bit rate and high distortion of losses.

3.7.2 Simulation Results

In this section, we evaluate the performance of the proposed stereoscopic video streaming system on lossy channels via simulations. We use two stereoscopic videos 'Rena' (Camera 38, 39) (640×480, first 30 frames) and 'Soccer' (720×480, first 30 frames) for performance evaluation. We encode the stereoscopic videos with the bit rates obtained by the minimization in (3.26) for given p_e and R_C, and NAL unit size is fixed to 150 bytes. The number of NAL units per layer, namely the number of input symbols for the Raptor code, can be calculated by dividing the given encoder bit rate to NAL unit size. The simulation results give

Table 3.2: The video encoder bit rates and Raptor encoder protection rates for the 'Rena' video

p_e	$R_C(Kbs)$	Encoder Bit Rates (Kbs)			Protection Rates Proposed (Optimal)		
		R_I	R_L	R_R	ρ_I	ρ_L	ρ_R
0.03	500	37.2	236.8	183.4	0.345	0.083	0.055
	750	56.8	367.8	270.8	0.261	0.069	0.054
	1000	76.5	499.7	358.8	0.213	0.061	0.051
	1500	115.8	764.6	535.5	0.162	0.052	0.047
	2000	155.2	1030.3	712.7	0.134	0.048	0.045
	2500	194.6	1296.4	890.1	0.116	0.045	0.043
0.05	500	36.1	230.9	179.4	0.384	0.109	0.081
	750	55.2	359.0	265.0	0.296	0.093	0.079
	1000	74.4	488.2	351.1	0.246	0.085	0.076
	1500	112.9	747.6	524.1	0.191	0.076	0.071
	2000	151.4	1007.8	697.6	0.162	0.071	0.068
	2500	190.0	1268.3	871.3	0.143	0.068	0.066
0.1	500	33.5	216.6	169.9	0.490	0.177	0.148
	750	51.6	337.9	250.8	0.389	0.159	0.144
	1000	69.7	460.1	332.3	0.332	0.149	0.140
	1500	106.0	705.7	496.1	0.270	0.138	0.134
	2000	142.5	952.0	660.3	0.237	0.132	0.130
	2500	178.9	1198.7	824.8	0.215	0.129	0.127
0.2	500	28.8	189.0	151.4	0.743	0.338	0.301
	750	44.7	296.3	223.0	0.612	0.313	0.294
	1000	60.8	404.7	295.3	0.538	0.300	0.288
	1500	93.0	622.6	440.6	0.457	0.285	0.280
	2000	125.4	841.3	586.4	0.414	0.278	0.275
	2500	157.8	1060.4	732.5	0.386	0.273	0.271

the average of 100 independent lossy transmission simulations for each p_e and R_C, where each packet is lost with a probability of p_e. Simulation results are based on the weighted PSNR measure which can be calculated as

$$PSNR_{weighted} = 10 \cdot \log_{10} \left(\frac{255^2}{D_{tot}} \right), \tag{3.28}$$

where the weighted MSE distortion D_{tot} is given in (3.27).

Table 3.3: The video encoder bit rates and Raptor encoder protection rates for the 'Soccer' video

p_e	$R_C(Kbs)$	Encoder Bit Rates (Kbs)			Protection Rates Proposed (Optimal)		
		R_I	R_L	R_R	ρ_I	ρ_L	ρ_R
0.03	1000	73.8	589.7	266.7	0.231	0.060	0.065
	1500	103.8	903.5	404.5	0.184	0.051	0.057
	2000	133.7	1217.9	542.8	0.155	0.047	0.052
	2500	163.6	1532.6	681.2	0.136	0.044	0.049
	3000	193.4	1847.7	819.7	0.123	0.042	0.046
	3500	223.3	2162.8	958.3	0.112	0.041	0.045
0.05	1000	72.2	576.2	260.6	0.263	0.084	0.091
	1500	101.5	883.4	395.6	0.214	0.074	0.081
	2000	130.7	1191.2	531.0	0.184	0.070	0.075
	2500	159.9	1499.5	666.5	0.164	0.067	0.072
	3000	189.2	1807.9	802.2	0.150	0.065	0.069
	3500	218.4	2116.6	937.9	0.139	0.063	0.068
0.1	1000	68.4	543.0	245.9	0.349	0.147	0.156
	1500	96.0	833.8	373.7	0.294	0.136	0.145
	2000	123.7	1125.3	501.9	0.260	0.130	0.138
	2500	151.3	1417.2	630.3	0.238	0.127	0.134
	3000	179.0	1709.3	758.7	0.222	0.125	0.131
	3500	206.6	2001.6	887.3	0.209	0.123	0.128
0.2	1000	61.5	477.4	217.1	0.552	0.298	0.312
	1500	86.0	735.6	330.6	0.484	0.284	0.295
	2000	110.5	994.4	444.4	0.442	0.276	0.287
	2500	135.1	1253.7	558.4	0.414	0.271	0.281
	3000	159.6	1513.2	672.5	0.393	0.268	0.277
	3500	184.2	1772.9	786.7	0.377	0.266	0.274

For channel protection, we use systematic Raptor codes based on their suitability for our case as explained in Section 2.5. We applied Raptor encoding to the source encoded video data using the protection rates obtained by the minimization in (3.26) for given p_e and R_C. The proposed optimal streaming scheme is compared with EEP, Protect-L, no-loss and no-protection cases. The protection rates of equal error protection (EEP) and Protect-L cases can be easily calculated from Tables 3.2 and 3.3. These protection rates are non-optimal and will be compared with the proposed optimal protection rates by simulations.

In order to construct the EEP case, the resulting bit rate of proposed protection is distributed to the layers so that each layer has the same protection ratio. Protect-L case is constructed similarly, using the results of [76], where the bit rate of protection is distributed to only layers of left view (Layer 1 and Layer 0) so that these layers have same protection ratio. The encoder bit rates for EEP and Protect-L are the same as the optimal streaming case, thus they use the optimal bit rates calculated by (3.26). The no-loss case represents the undistorted quality of the video when the stereoscopic video is encoded with all available channel bandwidth and, the bit rates of the layers are determined by the minimization in 3.8. The no-protection case represents the transmission of the video of no-loss case without any channel protection where only error concealment is used at the decoder.

We give the simulation results of stereoscopic video pair 'Rena' in Figures 3.17 to 3.20 and those of 'Soccer' in Figures 3.21 to 3.24. The noticeable observation is the large distortion in the no-protection case, even at low loss rates. This result clearly demonstrates the loss sensitivity of the stereoscopic video and points out the need for FEC utilization in stereoscopic video streaming. Another observation is that the performance of EEP and Protect-L cases depends on the chosen video. EEP is clearly better than Protect-L case for low bit rates for the 'Rena' video, however EEP is better for high bit rates for the 'Soccer' video. The proposed case is obviously better then the other cases. For low bit rates the difference is not clear but for high bit rates the difference is 1dB for $p_e = 0.10$ and nearly 2dB for $p_e = 0.20$. The gap between the results of the no-loss and the proposed case is caused by the reduction of the encoder bit rates of video where the remaining bit rate is used for channel protection. The results in Figures 3.17 to 3.24 demonstrate the necessity of FEC utilization and success of the proposed scheme. We handled the problem of end-to-end transmission with piecewise analysis and total distortion minimization. The proposed scheme can be easily extended to almost all types of layered multi-view codecs.

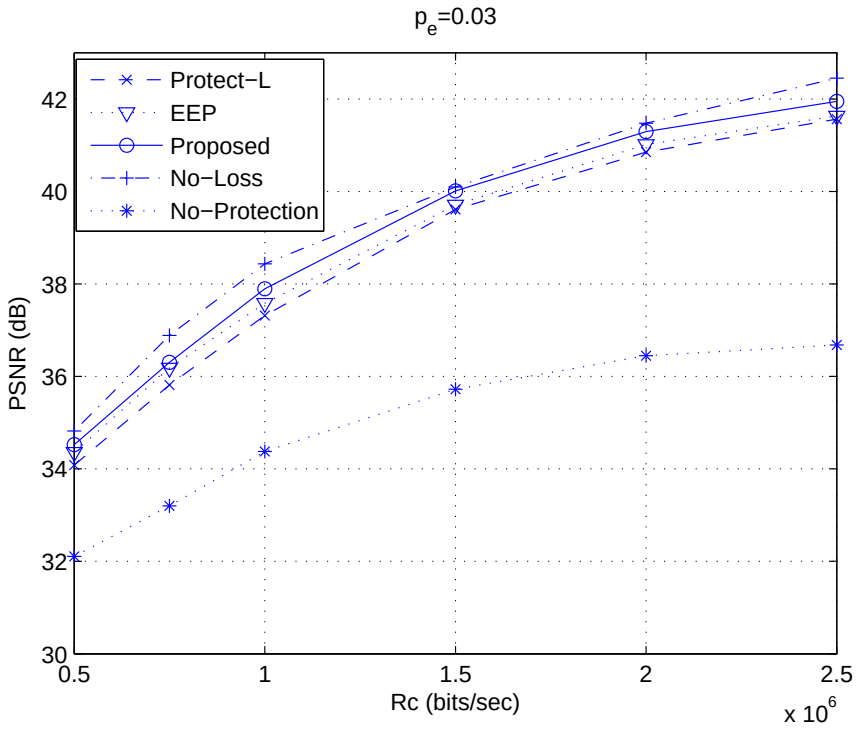


Figure 3.17: The results for $p_e = 0.03$ for the 'Rena' video

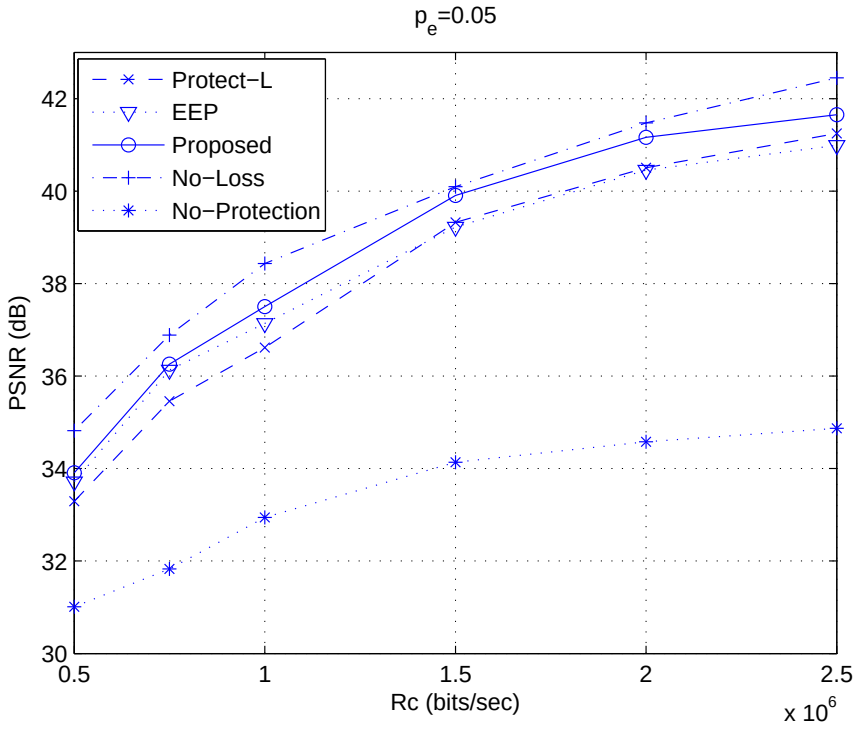


Figure 3.18: The results for $p_e = 0.05$ for the 'Rena' video

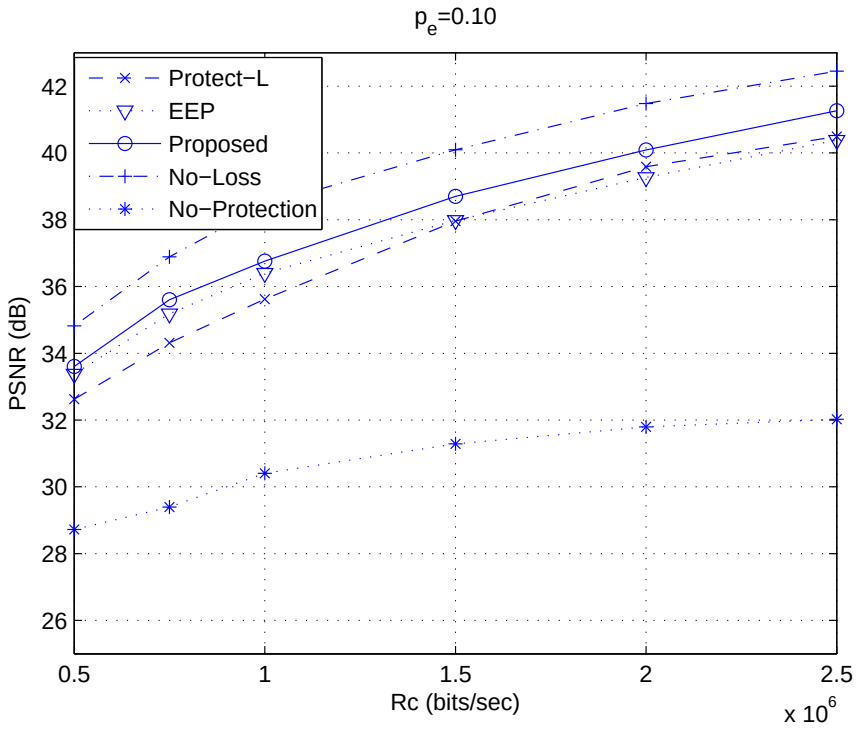


Figure 3.19: The results for $p_e = 0.10$ for the 'Rena' video

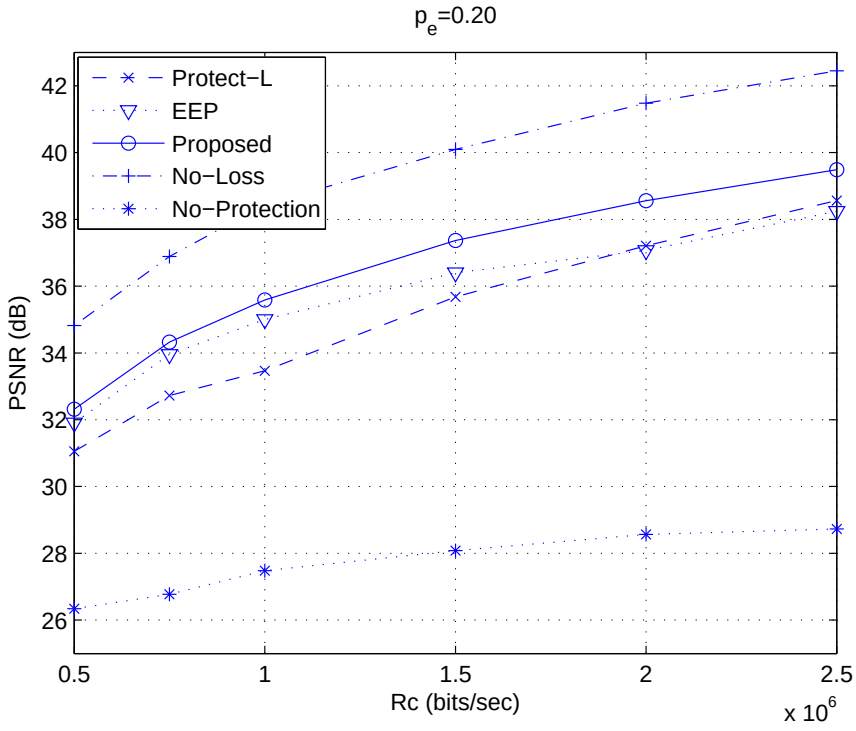


Figure 3.20: The results for $p_e = 0.20$ for the 'Rena' video

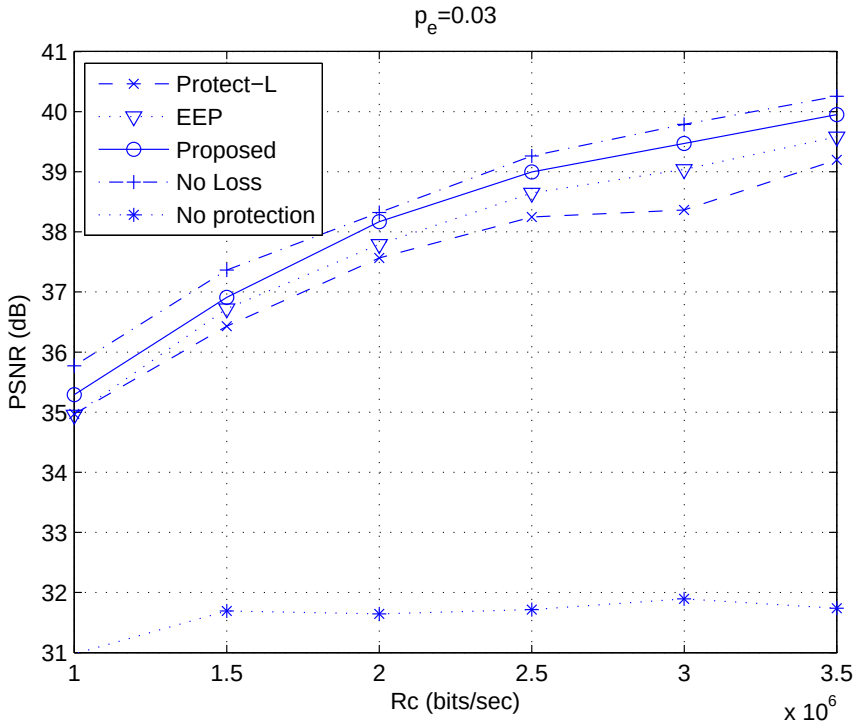


Figure 3.21: The results for $p_e = 0.03$ for the 'Soccer' video

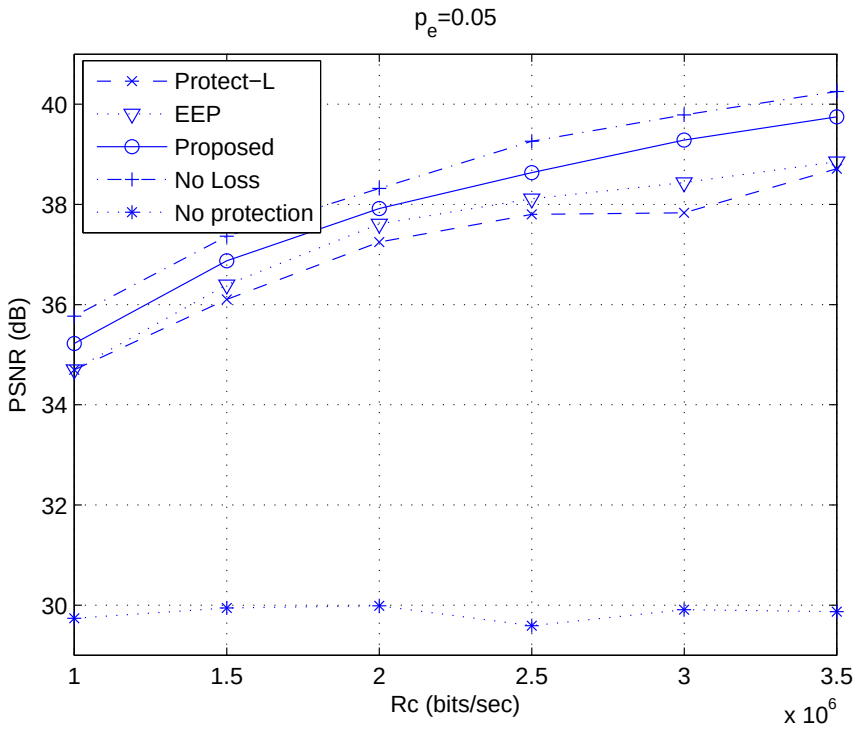


Figure 3.22: The results for $p_e = 0.05$ for the 'Soccer' video

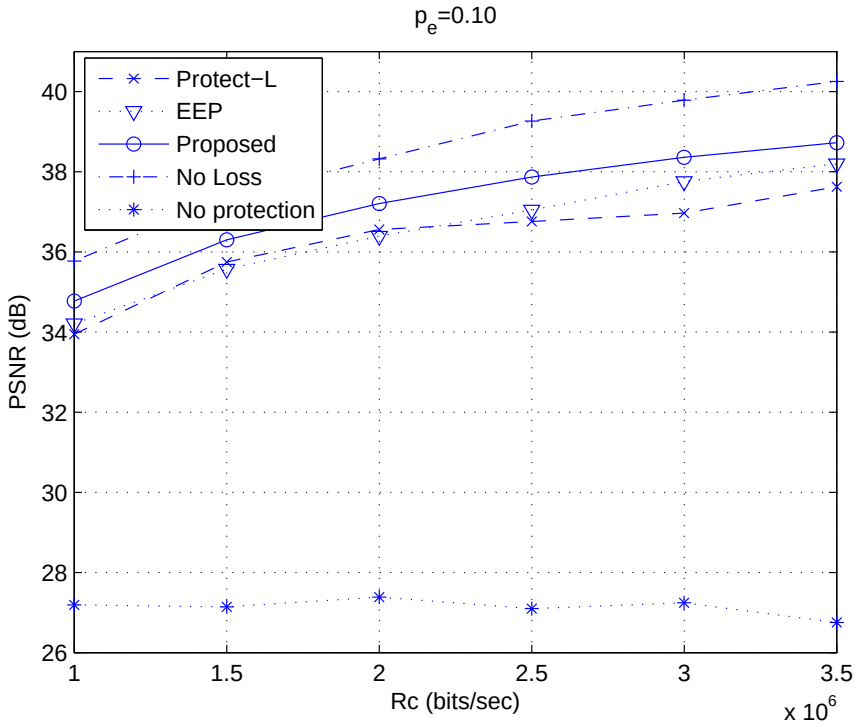


Figure 3.23: The results for $p_e = 0.10$ for the 'Soccer' video

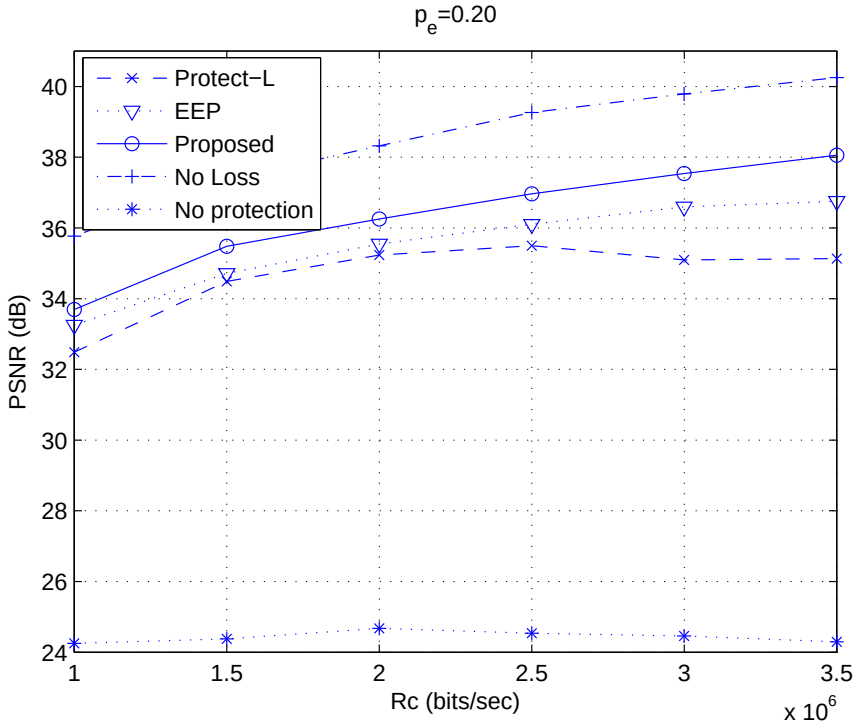


Figure 3.24: The results for $p_e = 0.20$ for the 'Soccer' video

Chapter 4

Analysis and Modeling of Fountain Codes

4.1 Motivation

The first practical realization of Fountain codes, namely LT codes were defined in [11]. That work defined the principles of LT codes and some bounds on the performance for asymptotic region. LT codes were shown to operate asymptotically optimal, namely when the number of input symbols increases significantly. The exact finite length analysis of LT codes that yielded significant high complexity of $\mathrm{O}\left(n^3 \log^2(n) \log\log(n)\right)$ were defined in [13]. This analysis was improved in [77] to yield a lower complexity with asymptotic Poisson approximation. Another study on the performance analysis of rateless codes was presented in [78] that aimed at the region where the number of received output symbols are less than the number of input symbols. In [79], exact analysis of LT codes with Markov chain approach are provided for number of input symbols smaller than 30. Following the introduction of Raptor codes [12], there has been various analysis and design approaches. In the original paper [12], performance analysis of

Raptor codes in the asymptotic region and finite length design for low number of input symbols were provided. In [80], pre-code only (PCO) Raptor codes were analyzed and a new definition of efficiency was defined for Fountain codes. The EXIT functions for LT and Raptor codes were defined in [81] as a part of the asymptotic analysis. The analysis of random linear Fountain codes combined with LT codes were provided in [82]. The previous studies on the performance analysis of Fountain codes used asymptotic approximations and most of them lacked comparisons with actual simulation results.

In Chapter 3, we used Raptor codes for end-to-end stereoscopic streaming. The complexity of such an end-to-end system is compelling to obtain simple analytical models for each part of the system. The heuristic model for Raptor codes was one of those we used in Chapter 3. In this chapter we analyze and model the performance of the LT codes and Raptor codes.

4.2 Design of Degree Distributions

4.2.1 Degree Distribution of LT Codes

The degree distribution of LT coding needs proper design in order to obtain an efficient coding scheme. The degree distribution has to be designed so that

- the entire input symbols have to be connected to at least one output symbol,
- there has to be degree-1 output symbols to initiate the decoding,
- the average degree has to be low to ensure low complexity encoding and decoding.

For ease of denotation of degree distributions and ease of operations, degree polynomial is defined. Let the the probability of degree i output symbols be Ω_i, then the degree polynomial is defined as

$$\Omega(x) = \sum_i \Omega_i x^i \ . \tag{4.1}$$

We explain two different process to describe the operation and degree design of the LT codes in the sequel.

Balls and Bins Process

A simple example about bins and balls is highly intuitive about the design of LT codes. Assume that there are k bins into which we throw balls. We want to calculate the number of balls that has to be thrown into the bins in order to guarantee that all bins have at least one ball with probability $1-\delta$. After throwing b balls, the probability that one bin is empty can be calculated as,

$$\left(1 - \frac{1}{k}\right)^b \simeq e^{-b/k} \ . \tag{4.2}$$

If $b = k$ balls are thrown the approximate fraction of bins that are empty is $1/e$. If $b = 4k$ balls are thrown the fraction is 1.8%. Thus, a large number of balls have to be thrown to fill all the bins. For any b, the expected number of empty bins can be calculated as,

$$k \cdot e^{-b/k}. \tag{4.3}$$

We want this number to be smaller than δ to ensure that all bins have at least one ball with probability $1-\delta$. Thus, b has to satisfy

$$b > k \ln \left(\frac{k}{\delta} \right). \tag{4.4}$$

In order to associate the above example with LT codes, assume the balls represent the edges and the bins represent the input symbols. In order to be a capacity achieving code, all input symbols (bins) need at least one edge (ball). Thus, the number of edges has to be greater than $k \ln (k)$ to ensure successful decoding which leads to an average degree of at least $\ln(k)$. In [11], it is shown that this bound can be achieved with the good design of degree distribution.

LT Process

In order to design the LT degree distributions, Luby defines the *LT process* as follows [11]:

- All input symbols are initially *uncovered*.
- In the first step, all output symbols with degree-1 are released to cover an input symbol.
- The set of covered input symbols that have not been processed is called the ripple.
- At each step, one input symbol in the ripple is processed, namely, the chosen input symbol from the ripple is removed from all output symbols that have it as a neighbor and the newly emerging output symbols with degree-1 are released to cover their neighbor.
- The process is complete when the ripple becomes empty. The process fails if there are uncovered input symbols when the ripple becomes empty.

The LT process at step t is illustrated in Figure 4.1. The figure illustrates the release of the output symbol y at step t. In order to be released at step t, y

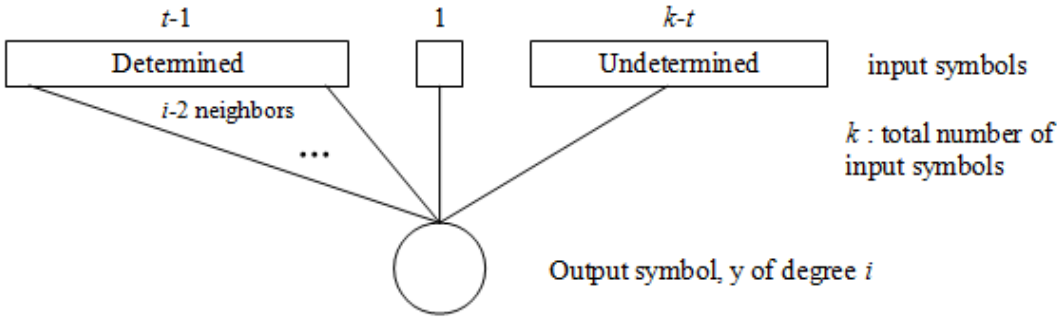


Figure 4.1: LT process at step t

needs to have its $i-2$ neighbors previously determined. Since the current step is t there has to be $t-1$ determined input symbols (Smaller values than $t-1$ mean decoding failure and larger values are not possible since 1 input symbol is released at each step). Then one of the two remaining neighbors of y has to be processed at step $t-1$ and the remaining neighbor has to be in the undetermined input symbols set. The number of undetermined input symbols at step t is calculated by $k-(t-1)-1=k-t$. Let

$$q(i,t)=\Pr\{\text{y is released at step } t|\text{degree(y)} =i\}=\frac{\binom{t-1}{i-2}\binom{1}{1}\binom{k-t}{1}}{\binom{k}{i}} \tag{4.5}$$

represent the probability that y is released at step t when it has a degree i. In (4.5), $q(i,t)$ is obtained by calculating the probability that $i-2$ of the neighbors of the output symbols of y are among the first $t-1$ input symbols determined, one neighbor is determined in step t, and the remaining neighbor is among the $k-t$ undetermined input symbols. If for all i, Ω_i is the probability distribution that an output symbol has degree i, since the events are disjoint, the overall release probability is

$$r(t)=\Pr\{\text{y is released at step } t\}=\sum_i \Omega_i \frac{\binom{t-1}{i-2}\binom{1}{1}\binom{k-t}{1}}{\binom{k}{i}}\ . \tag{4.6}$$

Ideal Soliton Distribution

The idea behind the design of Ideal Soliton distribution is that only one output symbol is expected to be released at each step of the decoding. When n output symbols arrive the number of released output symbols at step t can be expressed as $n \cdot r\,(t)$. In order to release one output symbol at each step we require

$$n \cdot r\,(t) = 1\ . \tag{4.7}$$

For sufficiently large k, $r(t)$ can be approximated as

$$\frac{1}{k}\left(1 - \frac{t}{k}\right)\sum_{i} \Omega_i i\,(i-1)\,\frac{(t-1)^{i-2}}{k^{i-2}}\ . \tag{4.8}$$

For an asymptotically optimal code we require $n = k$. Approximating $(t-1) \sim t$ and substituting x for $\frac{t}{k}$, (4.7) can be modified as

$$(1-x)\,\Omega''\,(x) = 1\ . \tag{4.9}$$

Solving for $\Omega(x)$ in (4.9) we obtain

$$\Omega(x) = c_0 + c_1 x + \sum_{i \geq 2} \frac{x^i}{i(i-1)}\ . \tag{4.10}$$

The Ideal Soliton distribution based on this result is given in [11] as

$$\begin{aligned} &\Omega_1 = 1/k \\ &\Omega_i = 1/i(i-1), i = 2, ..., k\ . \end{aligned} \tag{4.11}$$

The average degree of an output symbol with the Ideal Soliton distribution is $\ln(k)$. Thus, when k output symbols are received the number of edges becomes

Table 4.1: The Ideal and Robust Soliton distributions for $k = 500$ ($\delta = 0.01$, $c = 0.1$ for Robust Soliton)

i	1	2	3	4	5	6	7	8	...	21	...
Ω_i	0,002	0,5	0,1666	0,0833	0,05	0,0333	0,0238	0,0178	...	0,0023	...
μ_i	0,0324	0,3379	0,1178	0,0615	0,0384	0,0266	0,0198	0,0154	...	0,2445	...

$k \ln(k)$, which ensures covering all of the input symbols. According to balls and bins process, $k \ln(k/\delta)$ balls (edges) are required to succeed with probability $1 - \delta$. Thus, the encoding and decoding complexities will be $O(k \ln(k))$. The Ideal Soliton distribution Ω_i is given Table 4.1 for $k = 500$.

Ideal Soliton distribution behaves poorly due to the assumption that expected number of released output symbols is 1. However, the expected value of 1 is highly vulnerable to even a small variance. Thus, in any step of decoder the probability that there is no output symbol with degree is high.

Robust Soliton Distribution

The Robust Soliton distribution is proposed to remove the practical weakness of the ideal soliton distribution by increasing the expected number of degree-1 output symbols in each step. The robust soliton distribution is designed to ensure that the expected number of output symbols released at step t is

$$R = c \ln(k/\delta)\sqrt{\delta} \tag{4.12}$$

for some suitable constant $c > 0$. The parameter δ is used in a bound on probability that the decoding fails.

The Robust Soliton distribution is calculated in the following way. First we define

$$\tau_i = \begin{cases} R/ik & \text{for } i = 1, ..., k/R - 1 \\ R\ln(R/\delta)/k & \text{for } i = k/R \\ 0 & \text{for } i = k/R + 1, ..., k \end{cases} . \tag{4.13}$$

Then, we add the Ideal Soliton distribution Ω_i to τ_i and normalize with β to obtain the robust soliton distribution μ_i as

$$\begin{aligned} \beta &= \sum_{i=1}^{k} (\Omega_i + \tau_i) \\ \mu_i &= (\Omega_i + \tau_i)/\beta \ \text{ for } \ i = 1, ..., k \, . \end{aligned} \tag{4.14}$$

The Robust Soliton distribution μ_i is given in Table 4.1 for $k = 500$, $c = 0.1$, $\delta = 0.01$. There are two significant differences from the Ideal Soliton distribution; first, an increase in the probability of degree-1 output symbols, second, the spike at a high degree (i=21 in Table 4.1). The increase in the probability of a degree-1 symbol ensures that the decoding can start with a reasonable number of received output symbols. The spike at a high degree ensures that all of the input symbols are connected to an output symbol. The average degree of an output symbol can be calculated as $\ln(k/\delta)$. Thus, the complexity of encoder and decoder is $O(k \ln(k/\delta))$.

4.2.2 Degree Distribution of Raptor Codes

The degree distribution of LT codes is slightly modified, so that the distribution has a maximum degree of D, and degree one output symbols have an appropriate weight. The asymptotic degree distribution is defined as [12]

$$\Omega_D(x) = \frac{1}{\mu + 1} \left(\mu x + \sum_{i=2}^{D} \frac{x^i}{(i-1)\, i} + \frac{x^{D+1}}{D} \right) , \tag{4.15}$$

Table 4.2: The degree distribution of Raptor codes for different values of k

k	8192	65536	120000
η_1	0.009766	0.007969	0.004807
η_2	0.459042	0.493570	0.496472
η_3	0.210964	0.166220	0.166912
η_4	0.113392	0.072646	0.073374
η_5		0.082558	0.082206
η_8		0.056058	0.057471
η_9		0.037229	0.035951
η_{10}	0.111342		
η_{11}	0.079863		
η_{18}			0.001167
η_{19}		0.055590	0.054305
η_{40}	0.015627		
η_{65}		0.025023	0.018235
η_{66}		0.003135	0.009100
$\eta'(1)$	4.63	5.87	5.83

where $D := \lceil 4(1+\varepsilon)/\varepsilon \rceil$ and $\mu = (\epsilon/2) + (\epsilon/2)^2$ for some ε larger than zero. This degree distribution has the constant average degree $1 + H(D)/(1+\mu)$, where $H(D)$ is the harmonic sum up to D. The degree distribution is shown to be sufficient to recover at least $(1-\delta)k$ input symbols via BP decoding when $(1+\epsilon/2)k + 1$ output symbols are received, where $\delta = (\epsilon/4)/(1+\epsilon)$. The remaining δ constant fraction of the input symbols are recovered by the pre-code. The degree distribution given in (4.15) is optimal in the asymptotic region. In [12], optimal degree distributions for finite block lengths are also derived. Let $\eta(x)$ represent the degree polynomial of optimal distribution. Then, $\eta(x)$ is calculated by minimizing the average degree $\eta'(1)$ over a set of constraints, so that

$$\min_{\eta_d, d=1:k} \eta'(1) \qquad (4.16)$$
$$\eta'(x) \geq \frac{-\ln\left(1-x-c\sqrt{\frac{1-x}{k}}\right)}{1+\epsilon} \quad .$$

In Table 4.2, some results on the minimization in (4.16) are given for different values of k.

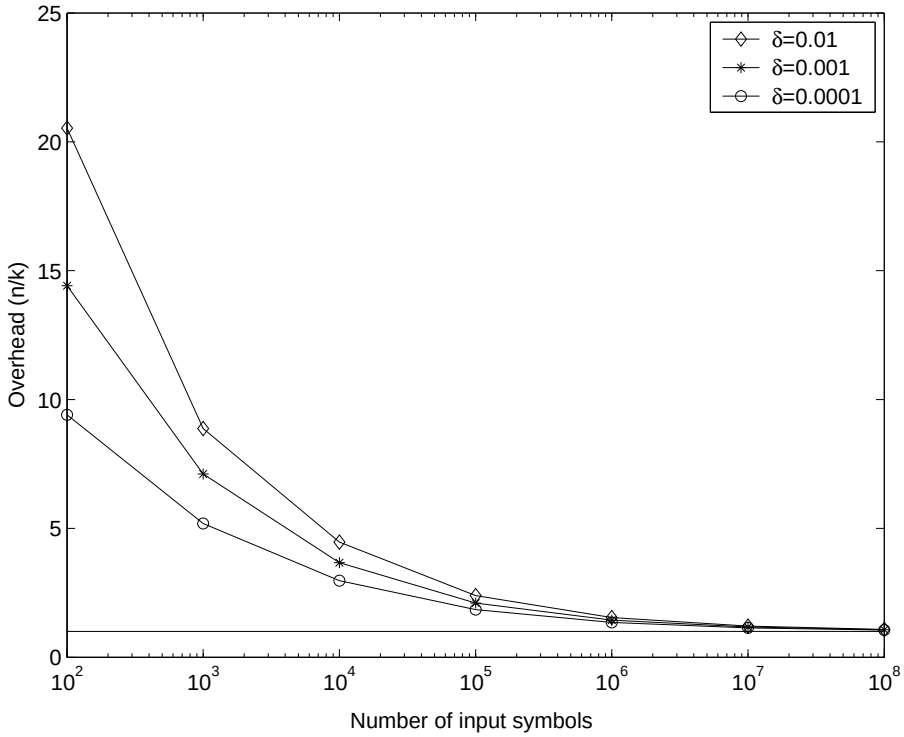


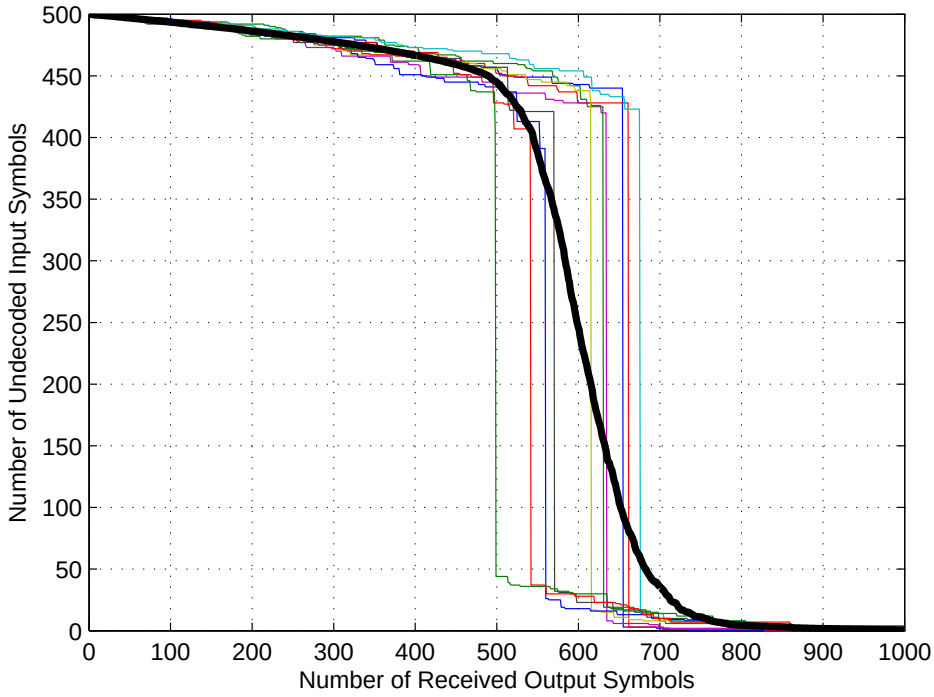
Figure 4.2: The overhead of LT codes

4.3 Performance of LT Codes

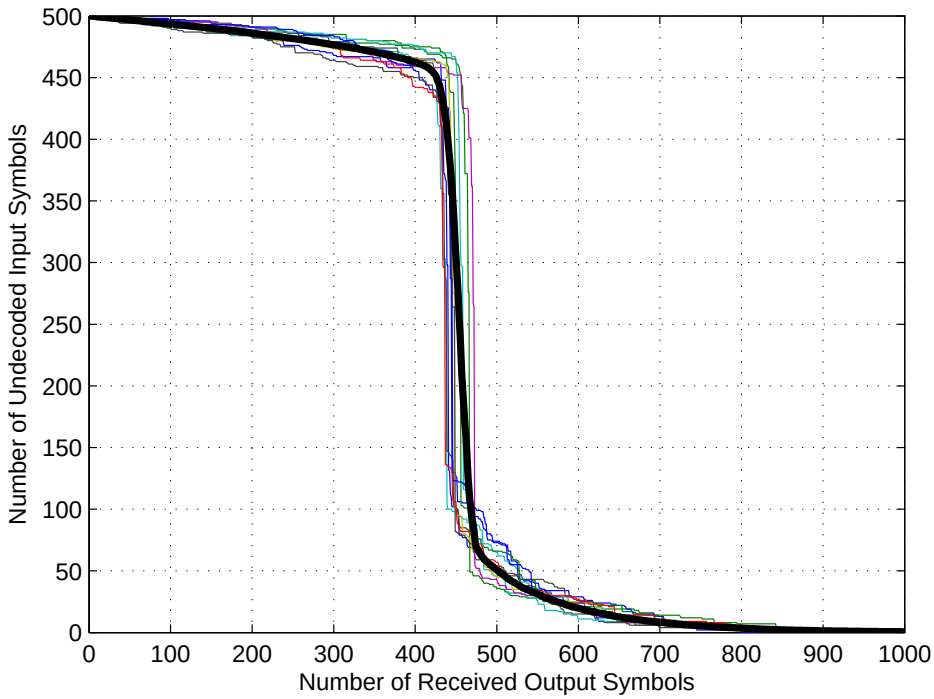
The performance of Fountain codes is measured by the curve that yields the average number of undetermined (or determined) input symbols versus the number of received output symbols. If we denote k and n as the number input symbols and output symbols respectively, then the ratio $(n-k)/k$ can be called fractional overhead. When Robust Soliton distribution is used, the required number of output symbols for LT codes in order to succeed in decoding with probability $1-\delta$ is given in [11] as

$$n = k + \alpha \cdot \ln\left(\frac{\alpha}{\delta}\right) + \sum_{i=1}^{k/\alpha-1} \left(\frac{\alpha}{i}\right), \tag{4.17}$$

where $\alpha = c \cdot \sqrt{k} \cdot \log(k/\delta)$ for some c. Figure 4.2 shows this ratio as a function of k for $c = 1$ and $\delta = 0.01, 0.001, 0.0001$.



(a)



(b)

Figure 4.3: The performance curve of LT codes with Robust Soliton distribution, $k = 500$, $c = 0.7$ and $\delta = 0.001$, (a) LT BP decoder, (b) LT ML decoder

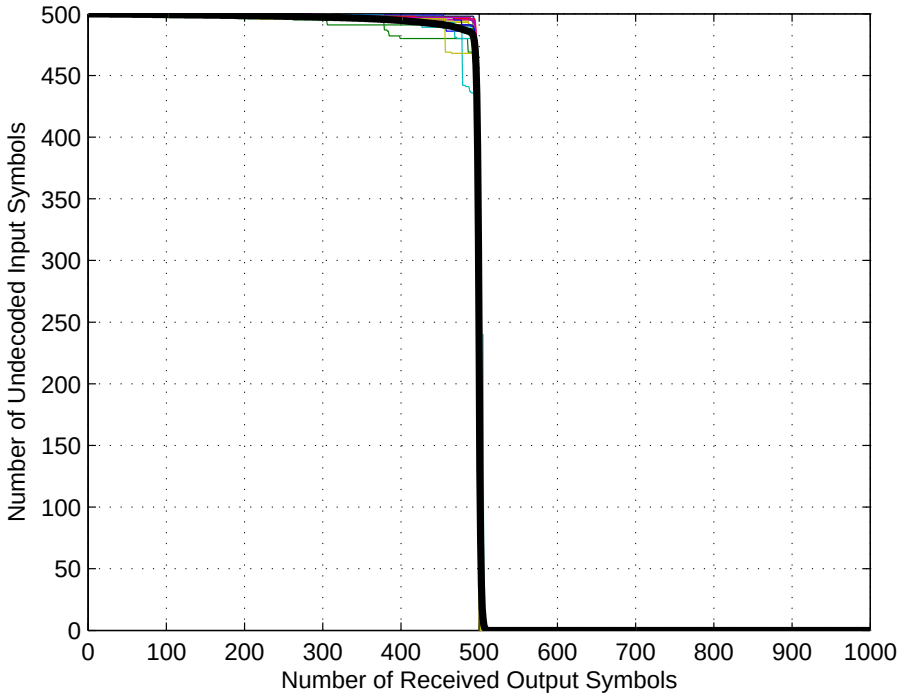
The overhead versus the number of input symbols is significant unless the number of input symbols k is on the order of 10^7. Thus, LT coding as proposed by Luby is asymptotically optimal but yields excessive overhead in the non-asymptotic regime.

We also give practical simulation results of an LT code with BP decoder in Figure 4.3(a). We use Robust Soliton distribution with $k = 500$, $c = 0.7$ and $\delta = 0.001$. The thin curves show the exact results of several encoding and decoding simulations. The bold black line represents the average of 100 simulations. In Section 2.4.2, the ML decoder of the LT coding is described. The performance of LT ML decoder is given in Figure 4.3(b), similar to the BP decoder case.

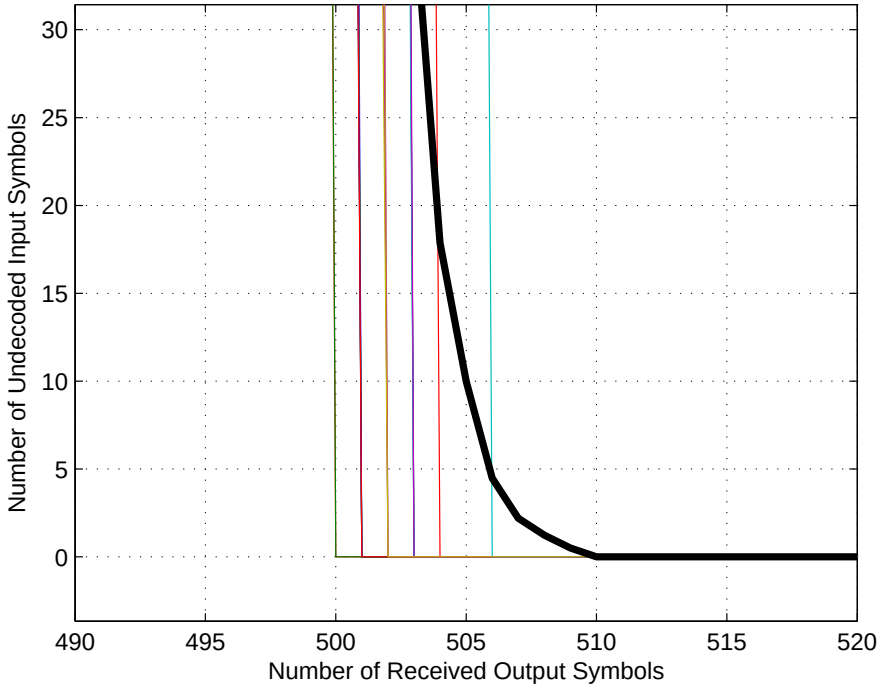
4.4 Performance of Raptor Codes

As mentioned in previous sections Raptor codes are more advanced than their prior LT codes. The performance of Raptor can be observed in Figure 4.4(a), where we provide the average of 100 simulations together with several exact simulation results. The steep performance when 500 output symbols arrive is zoomed in Figure 4.4(b).

In Figure 4.5 we provide the comparison of the average performance of LT and Raptor codes. When the number of received output symbols is smaller than the number of input symbols, Raptor codes can decode lower number of input symbols. However, in the opposite side, Raptor codes perform significantly better than the LT codes due to the two staged encoding.



(a)



(b)

Figure 4.4: The performance curve of Raptor Codes with the Raptor distribution in Table 4.2, $k = 500$, (a) Whole Performance, (b) Performance zoomed around 500 received output symbols

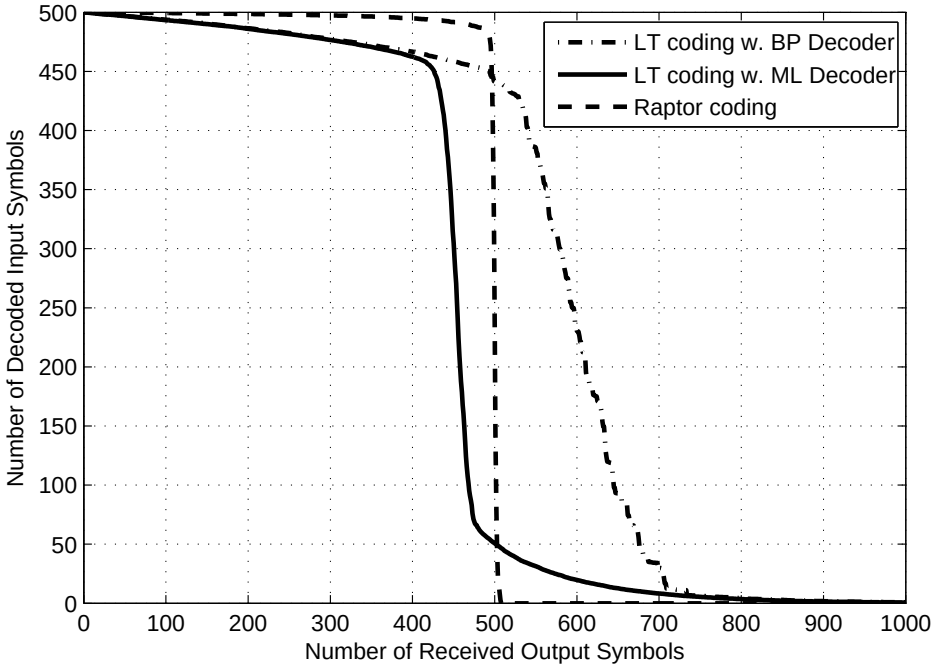


Figure 4.5: The comparison of the performance of LT codes with ML and BP decoder and Raptor codes

4.5 General Method for the Analysis of Fountain Codes

The analysis of Fountain codes is an iterative process. In this section, we try to obtain the analytical model of the LT codes and Raptor codes.

4.5.1 The Analysis of LT BP Decoder

The LT BP decoder operates according to the algorithm given in Section 2.4.2 whose progression can be tracked via the number of edges. Here, we only model the progression of the number of edges throughout the decoding process when a constant number of output symbols has arrived. This method will be used as an intermediate step in the next section to model the performance curve of the LT BP decoder. In [34] a method utilizing difference equations method is presented

for the analysis of the elimination of the edges in the bipartite graphs. Let $R_i(t)$ and $L_i(t)$ represent the expected number of edges with degree i connected to a right node and left node at the t^{th} step of decoder, respectively.

A single step of this decoding process is given in Figure 2.3. At each step a node of degree 1 on the right is chosen, if it exists, and the corresponding node on the left and all of its adjacent edges are removed from the graph. We start by analyzing the behavior of $L_i(t)$. Initially, as described in Section 2.4.2, a degree-1 output symbol is chosen and its corresponding edge is removed from the graph. If the chosen degree-1 edge on the right is connected to a degree-i edge on the left then the number of edges with degree-i on the left will decrease by i. Let $\mathrm{P}_L(i,t)$ denote the probability that the chosen edge is connected to a left node with degree-i at step t. Then, the number of left nodes with degree-i at the next step can be calculated as

$$L_i(t+1) = L_i(t) - i\mathrm{P}_L(i,t)\ ,\ i = 1,2,...,k\ \ , \tag{4.18}$$

where

$$\mathrm{P}_L(i,t) = \frac{L_i(t)}{E(t)}\ ,\ i = 1,2,...,k \tag{4.19}$$

$$E(t) = \sum_i R_i(t) = \sum_i L_i(t)\ . \tag{4.20}$$

Let $A(t) = \sum_i i\mathrm{P}_L(i,t)$ denote the number of removed edges from left nodes. The number of removed edges is used in the calculation of $R_i(t)$, since those are also removed from right nodes. However, one edge was already removed from right nodes when a degree-1 output symbol was chosen. Thus, on the average $A(t)-1$ edges will be removed from right nodes. If one of these edges is connected to a degree-i node on the right then the number of edges with degree-i on the right will decrease by i. Similarly the number of edges with degree-i will increase

by i if the removed edge is connected to a degree-(i+1) node. Let $\mathrm{P}_R(i,t)$ denote the probability that the chosen edge is connected to a right node with degree-i at step t. Then, the number of left nodes with degree-i at the next step can be calculated as

$$R_i(t+1) = \begin{cases} R_1(t) + [\mathrm{P}_R(2,t) - \mathrm{P}_R(1,t)](A(t)-1) - 1 & i = 1 \\ R_i(t) + [\mathrm{P}_R(i+1,t)\,i - \mathrm{P}_R(i,t)\,i](A(t)-1) & 1 < i < k-1 \\ R_k(t) - \mathrm{P}_R(k,t)\,k\,(A(t)-1) & i = k \end{cases}, \tag{4.21}$$

where

$$\mathrm{P}_R(i,t) = \frac{R_i(t)}{E(t)}, i = 1,..,k\ . \tag{4.22}$$

The recursive equation for $i = 1$ is different from the others. Because an edge from a degree-1 node on the right was removed at the beginning of a step. Hence, for $i = 1$, 1 has to be subtracted for the recursive equation. Among all nodes, the progression of $R_1(t)$ is the most important. Because it has to stay larger than 0 for the decoding process to succeed. Otherwise, the decoding fails, as stated in Section 2.4.2.

Evaluation of the Iterative Analysis

Before initiating the iterations, $R_i(0)$ and $L_i(0)$ have to be initialized for $1 \leq\ i \leq\ k$, and the whole bipartite graph has to be constructed. In order to achieve these, the edge degree distribution on both sides has to be calculated. We start with the calulation of edge degree distribution on right. Assume right node degree distribution is defined with the degree polynomial $\Gamma(x) = \sum_i \Gamma_i x^i$. Then, the edge degree distribution polynomial on right can be easily calculated

as $\rho(x) = \Gamma'(x)/\Gamma'(1)$. The edge degree distribution on left needs more effort. First we have to calculate the node degree distribution on left, and then the edge degree distribution. An edge from a right node is connected to left node according to uniform distribution. The probability that a node on left is the neighbor of a node on right is $\Gamma'(1)/k$, where $\Gamma'(1)$ is the average degree of a right node. Let $\Lambda(x)$ denote the left node degree distribution polynomial, then it can be calculated as

$$\Lambda(x) = \left(\left(1 - \frac{\Gamma'(1)}{k}\right) + \frac{\Gamma'(1)}{k}x\right)^n \tag{4.23}$$

for n received output symbols. The corresponding edge degree distribution on left can be calculated as $\lambda(x) = \Lambda'(x)/\Lambda'(1)$. The total number of edges can be calculated as $E = n\Gamma'(1) = k\Lambda'(1)$. Finally, the initial conditions can be defined as

$$R_i\,(0) = E\rho_i\ ,\ L_i\,(0) = E\lambda_i\ ,\ \text{for } i = 1, .., k\ . \tag{4.24}$$

The iteration results for $R_1(t), R_2(t)$ and $R_3(t)$ are shown in Figure 4.6 for $k = 500$ and $n = 2000$, where the analytical result is compared with the average of 100 simulations. The vertical axis is the average number of edges, and the horizontal axis is the iteration step. As seen from the results the analytical model perfectly tracks the progression of the number of edges of a LT BP decoder.

4.5.2 Modeling the Performance Curve of LT BP Decoder

The analysis in Section 4.5.1 cannot yield the performance curve of the LT BP decoder alone. Because the analysis only tracks the number of edges on left and right nodes for a given number of output symbols. In this section, using the previous analysis of LT BP decoder, we try to obtain the average number of

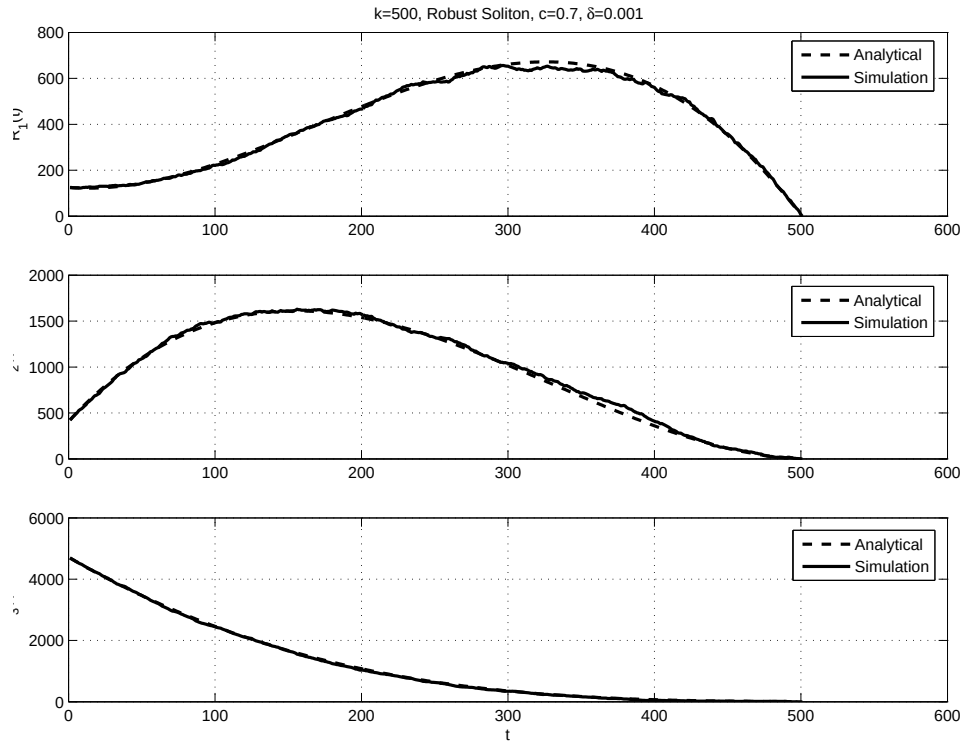


Figure 4.6: The comparison of the analytical and simulation results on LT BP decoder

decoded input symbols versus the number of received output symbols, namely the performance curve.

We use the notation $\tilde{R}_i(n)$ and $\tilde{L}_i(n)$ to denote the number of edges remaining after n^{th} output symbol arrival. Thus, initially $\tilde{L}_i(0) = 0$ and $\tilde{R}_i(0) = 0$ for $i = 1, ..., k$. The iterations start with the update of $\tilde{R}_i(n)$ as

$$\tilde{R}_i\left(n\right) = \tilde{R}_i\left(n-1\right) + i\Gamma_{i,n}\ ,\ i = 1, 2, ..., k\ , \tag{4.25}$$

where $\Gamma_{i,n}$ is the effective degree of an arriving right node at the n^{th} output symbol arrival. The degree of an arriving output symbol changes, since it may contain previously determined input symbols. The effective degree distribution coefficients can be calculated as

$$\Gamma_{i,n} = \sum_{j=i}^{k} \Gamma_{i,0} \binom{j}{j-i} \left(P_{dec}\left(n\right)\right)^{j-i} \left(1 - P_{dec}\left(n\right)\right)^{i} , \tag{4.26}$$

where $i = 1, ..., k$, and $P_{dec}(n)$ is the probability of decoding of an input symbol. After algebraic manipulations, using the degree coefficients given in (4.26), we can obtain the degree polynomial of nodes on the right at the n^{th} output symbol arrival as

$$\Gamma(x, n) = \Gamma\left(P_{dec}(n) + x\left(1 - P_{dec}(n)\right)\right) . \tag{4.27}$$

The iterations for the number of edges on left nodes are calculated with effective degree similar to that of right nodes. When the n^{th} output symbol arrives, $\Gamma'(1, n)$ edges will be connected to left nodes on the average. The degree-i on the left nodes is effected by two events. First, if an edge is connected to a degree-i node, then the number of degree-i nodes decrease by i. Second, if an edge is connected to a degree-$(i-1)$ node, then the number of degree-i nodes increase by i. Thus, a new arriving node on the right effects the number of edges on the left as

$$\tilde{L}_i\left(n\right) = \tilde{L}_i\left(n-1\right) + \left[\left(\frac{\tilde{L}_{i-1}\left(n-1\right)}{i-1} - \frac{\tilde{L}_i\left(n-1\right)}{i}\right)\frac{i}{k}\right]\Gamma'(1, n) \tag{4.28}$$

for $i = 1, ..., k$. After every arrival of an output symbol the BP decoder modeled in Section 4.5.1 has to run until $R_1(n)$ becomes less than 1. We set the initial conditions as

$$R_i\left(0\right) = \tilde{R}_i\left(n\right) \ , \ L_i\left(0\right) = \tilde{L}_i\left(n\right) \ , \text{ for } i = 1, .., k . \tag{4.29}$$

Assuming $R_1(t)$ becomes less than 1 just after step T we obtain $\tilde{L}_i(n)$ and $\tilde{R}_i(n)$ as

$$\tilde{R}_i(n) = R_i(T) \ , \ \tilde{L}_i(n) = L_i(T) \ , \text{ for } i = 1, .., k \ . \tag{4.30}$$

The average number of eliminated equations, $\Delta(n)$, after the arrival of n^{th} output symbol can be calculated as

$$\Delta(n) = \sum_i R_i(0) - \sum_i R_i(T) \ . \tag{4.31}$$

The average number of eliminated equations includes redundant equations which are dependent on other equations. Let $P_{red}(n)$ denote the probability of an equation being redundant. Then, the number of eliminated equations that are not redundant after the arrival of n^{th} output symbol, $\Delta_1(n)$, can be defined as

$$\Delta_1(n) = \sum_{i=0}^{\lfloor \Delta(n) \rfloor} \left[\binom{\lfloor \Delta(n) \rfloor}{i} (\Delta(n) - i) \left(P_{red}(n)\right)^i \times \left(1 - P_{red}(n)\right)^{(\lfloor \Delta(n) \rfloor - i)} \right] . \tag{4.32}$$

After the elimination of redundant equations, the equations are simply the number of decoded input symbols at n^{th} output symbol. However, these input symbols include previously decoded input symbols. The number of decoded input symbols that are not previously decoded after the arrival of n^{th} output symbol, $\Delta_2(n)$, can be calculated as

$$\Delta_2(n) = \sum_{i=0}^{\lfloor \Delta_1(n) \rfloor} \left[\binom{\lfloor \Delta_1(n) \rfloor}{i} (\Delta_1(n) - i) \left(P_{dec}(n)\right)^i \times \left(1 - P_{dec}(n)\right)^{(\lfloor \Delta_1(n) \rfloor - i)} \right] . \tag{4.33}$$

The number of decoded input symbols after the n^{th} symbols arrival, $N_{dec}(n)$, can be updated as

$$N_{dec}\,(n) = N_{dec}\,(n-1) + \Delta_2(n)\ . \tag{4.34}$$

The probability of decoding of an input symbol can be updated for the next step as

$$P_{dec}\,(n+1) = N_{dec}\,(n)\,/k\ . \tag{4.35}$$

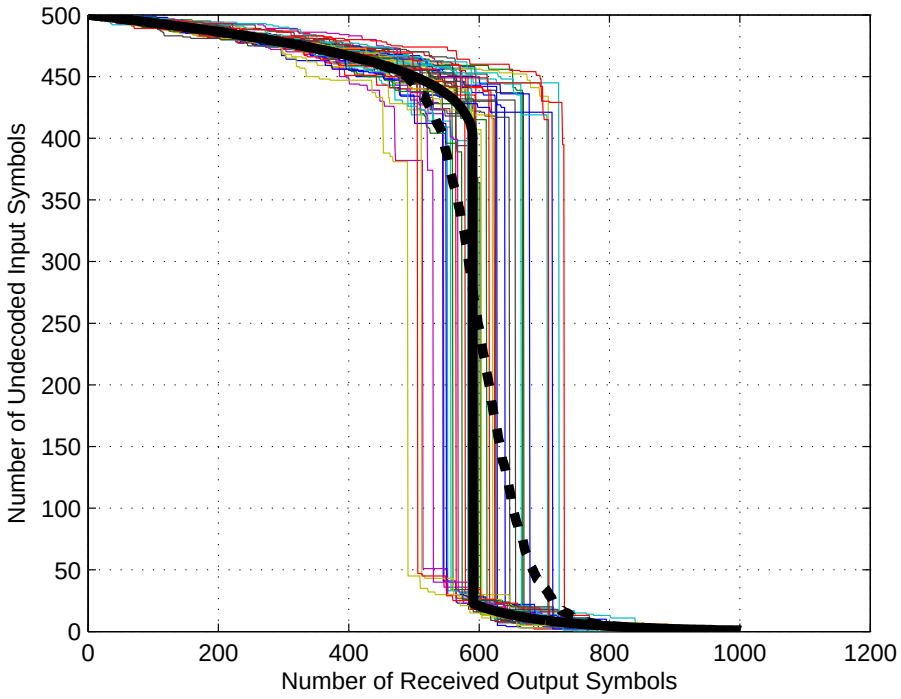
The probability of a redundant equation can be updated for the next step as

$$P_{red}\,(n+1) = \frac{n - \sum_i \tilde{R}_i(n)/i - N_{dec}(n)}{n}\ . \tag{4.36}$$

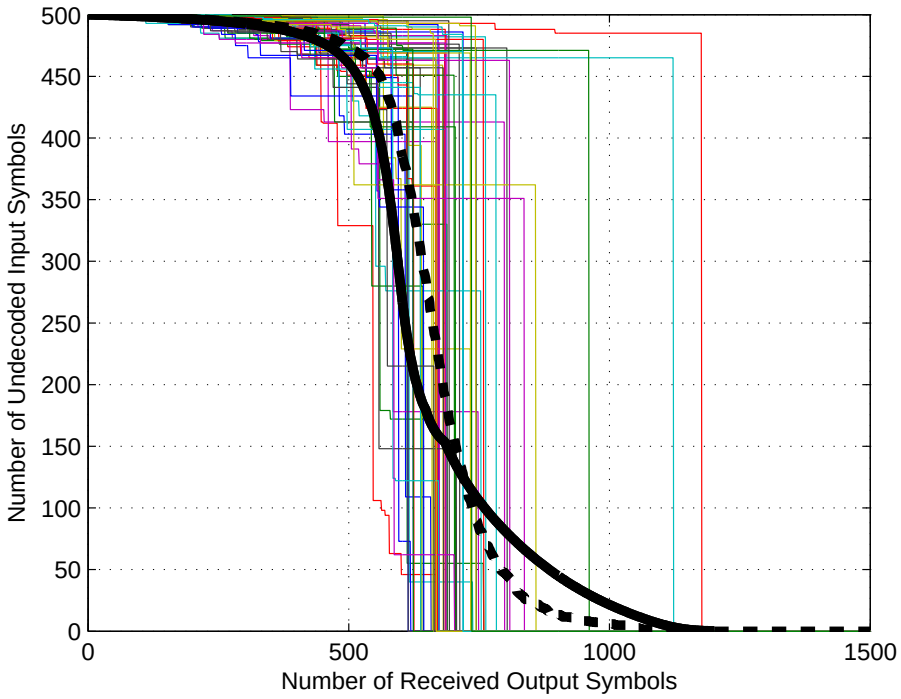
In (4.36), in the numerator, the term $\sum_i \tilde{R}_i(n)/i$ is the number of non-eliminated equations, the term $N_{dec}(n)$ is the number of decoded input symbols. Dividing the number of redundant equations by the number of all received equations we obtain the probability of an equation being redundant.

Evaluation of the Modeling

The steps (4.25) through (4.36) are executed iteratively starting from $n = 0$, the initialization step. At the initial step, we set $R_i(0) = 0$ and $L_i(0) = 0$ for $i = 1, ..., k$, $N_{dec}(0) = 0$, $P_{red}(0) = 0$, and $P_{dec}(0) = 0$. In Figures 4.7(a) and 4.7(b) we present two different results on the modeling with different parameters. In Figure 4.7(a), the modeling result does not exactly match the average of the simulations. However, the model approximates the behavior of a single simulation, and the initial and final parts of the model accurately approximates the average of simulations. On the other hand, in Figure 4.7(a), the modeling is satisfactory for the stated parameters.



(a)



(b)

Figure 4.7: Modeling the performance of LT BP decoder. Black bold solid line: model, black bold dashed line: the average of simulations. (a) $c = 0.7$, $\delta = 0.001$, (b) $c = 0.02$, $\delta = 0.001$

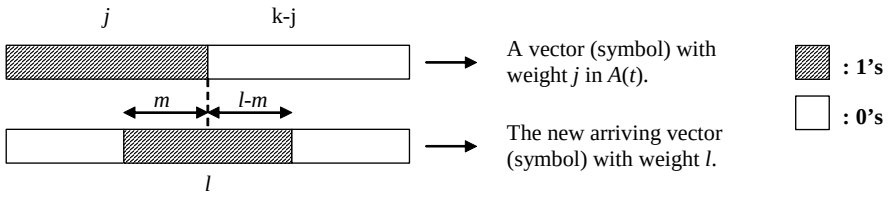


Figure 4.8: The representation of two vectors corresponding to two output symbols with degrees j and l

4.5.3 Modeling the Performance Curve of LT ML Decoder

The ML decoder is the optimal decoder for the LT codes. As shown in Section 2.4, the output symbols actually denote equations generated from the input symbols. Thus, ML decoding for LT codes is solving a set of linear equations. Although there are many different algorithms, Gaussian elimination is the most common. In order to analyze the performance of the ML decoder, we track the linear combinations of the rows of the output symbols' generator matrix, $\mathbf{A}_i$, in GF(2).

In order to track the number of linear combinations we use an iterative analysis. Let $w(i,n)$ represent the expected number of different vectors with weight i after calculating all linear combinations of the generating vectors of the n output symbols. We need to find $w(i,n+1)$ in terms of $w(.,n)$. When a new output symbol arrives, all possible linear combinations and the weight of these combinations has to be calculated. Assume a symbol with weight l has arrived; the below vector in Figure 4.8. Without loss of generality we pick a vector with weight j from $\mathbf{A}$; the above vector in Figure 4.8. Assume that 1's coincide at m locations as shown in the figure. Then we obtain a new vector with weight $i=(j-m)+(l-m)$. The probability of m 1's coinciding with j 1's and $(l-m)$ 0's coinciding with $(k-j)$ 0's can be represented with hyper-geometric distribution as

$$h\left(m,l,j,k\right)=\frac{\binom{j}{m}\binom{k-j}{l-m}}{\binom{k}{l}}\ , \tag{4.37}$$

where $m=(j+l-i)/2$, m integer. Summing over all vectors with degree-j, the increase in the number of linear combinations with combined weight-i, when the $(n+1)^{th}$ output symbol has degree-l, can be calculated as

$$\sum_{j=0}^{k} w\left(j,n\right) h\left(\frac{j+l-i}{2},l,j,k\right)\ . \tag{4.38}$$

Then, summing over all possible values of degree-l, the final iteration can be calculated as

$$w\left(i,n+1\right)=w\left(i,n\right)+\sum_{l=1}^{k}\Gamma_l\sum_{j=0}^{k} w\left(j,n\right) h\left(\frac{j+l-i}{2},l,j,k\right) \tag{4.39}$$

for $i=1,...,k$, $(j+l-i)/2$ integer, and Γ_l $(1\leq l\leq k)$ is the degree distribution of the output symbols. The average number of weight-1 vectors after linear combinations gives the number of determined input symbols that can be obtained with $w(1,n)$. However, the term $w(1,n)$ is not enough to obtain the expected number of determined input symbols. Because it includes linear combinations with vectors of weight-0 that is given by $w(0,n)$. Hence, the expected number of decoded input symbols when n output symbols arrive can be calculated as $w(1,n)/w(0,n)$.

Evaluation of the Modeling

The result on the analytical method for ML decoder is given Figure 4.9 and compared with the simulation results. The degree distribution is the Robust Soliton distribution with parameters $k=500$, $c=0.7$ and $\delta=0.001$. The black

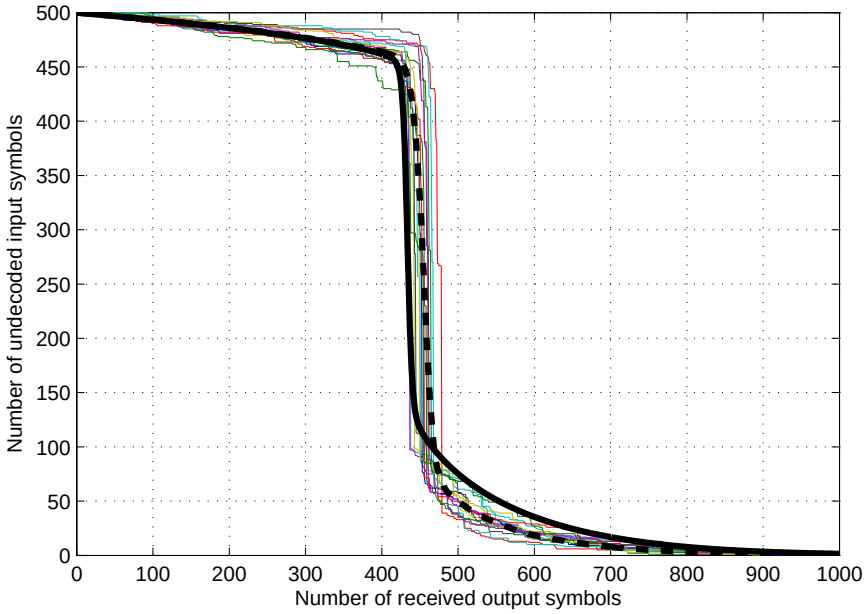


Figure 4.9: The comparison of the analytical model and simulation results on the LT ML decoder

bold solid line is the model and the black bold dashed line is the average of simulations. There is a small difference between the curves that is caused by a slight abuse on the expected value operator, where we assumed distribution over division property. Though, the analytical result approximates the LT simulation results accurately.

4.5.4 Modeling the Performance Curve of Raptor Decoder

We investigate the performance of both non-systematic and systematic Raptor codes for the algorithm described in [16]. For both of the cases, the modeling of the performance of Raptor codes is based on observations on the simulation results rather than theoretical iterations.

First, we focus on the non-systematic case. As easily observed from Figure 4.4(b), the number of undecoded input symbols decays exponentially when

the number of received symbols exceeds the number of input symbols. Based on this observation the performance of the Raptor code for this exponential decay can be written as

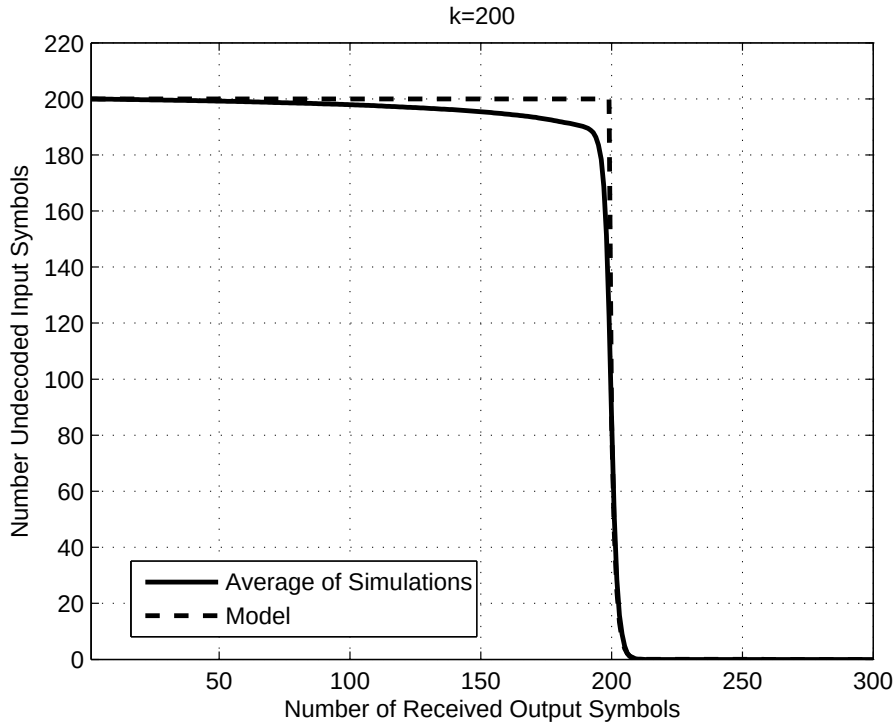
$$N_{undec}(r,k) = k \cdot \beta \cdot \alpha^{r-k} \, , r \geq k \, , \tag{4.40}$$

where $N_{undec}(r,k)$ is the number of undecoded input symbols when $r \geq k$ output symbols are received. The parameters β and α can be found simply by line of best fit to several simulations in the log scale plot. In [83], this method was shown to yield accurate approximate results. When $\beta = 0.42$ and $\alpha = 0.54$ the model yields an accurate approximation. Thus, the end-to-end model of the Non-systematic Raptor code can be written as

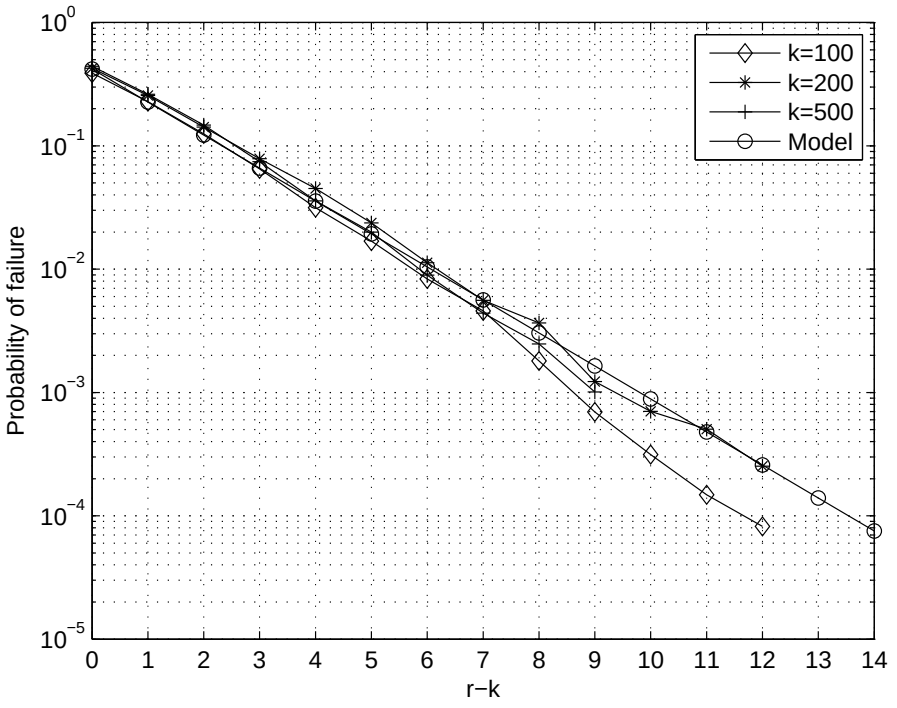
$$N_{undec}(r,k) = \begin{cases} k & r < k \\ k \cdot 0.42 \cdot (0.54)^{r-k} & r \geq k \end{cases} . \tag{4.41}$$

We present the results on the modeling for $k = 200$ using (4.41) in Figure 4.10(a). We also provide the normalized performance in log-scale for $r \geq k$ in Figure 4.10(b), where the vertical axis is the probability of failure, and the horizontal axis is the code overhead; $r - k$ for $r \geq k$.

The modeling of the performance of the systematic Raptor codes is a similar process to that of non-systematic Raptor codes. In order to form the model, we investigate the performance curve in two separate regions; first in the region with the number of received symbols less than the number of input symbols and, second, in the remaining region. In the first region of the model, we assume that the Raptor decoder cannot decode any lost symbols other than the received systematic symbols. Whereas, in the second region, an exponential decrease in the number of undecoded symbols is assumed. The performance in the second region can be modeled as



(a)



(b)

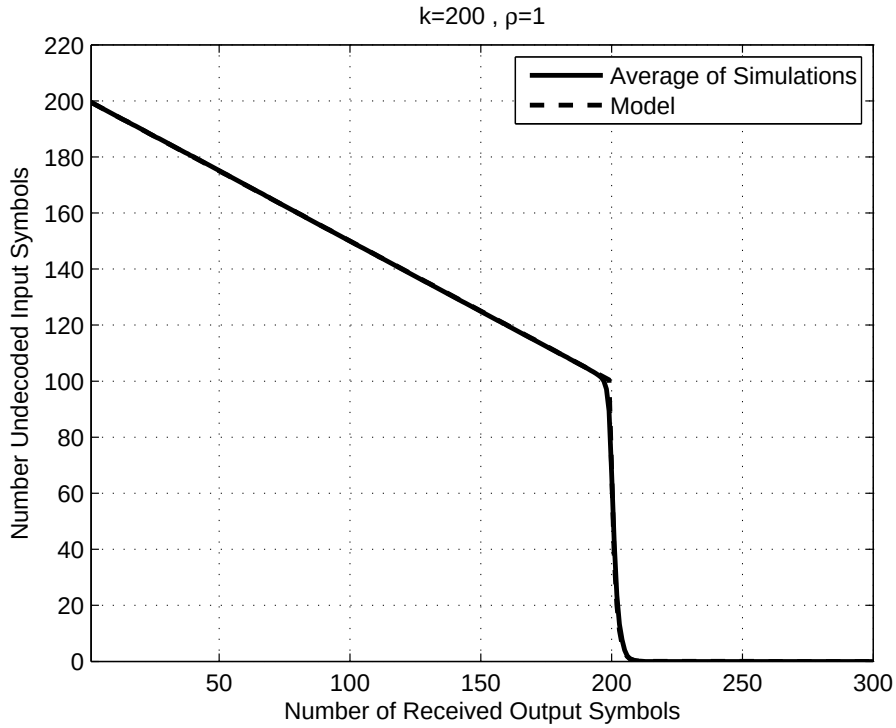
Figure 4.10: The results on modeling the performance of non-systematic Raptor codes (a) whole performance for $k = 200$ (b) normalized log-scale plot for $r \geq k$ for $k = 100$, $k = 200$ and $k = 500$

$$N_{undec}(r,k,\rho) = k \cdot \beta' \frac{\rho}{1+\rho} \cdot \alpha'^{r-k} \, , r \geq k \, , \tag{4.42}$$

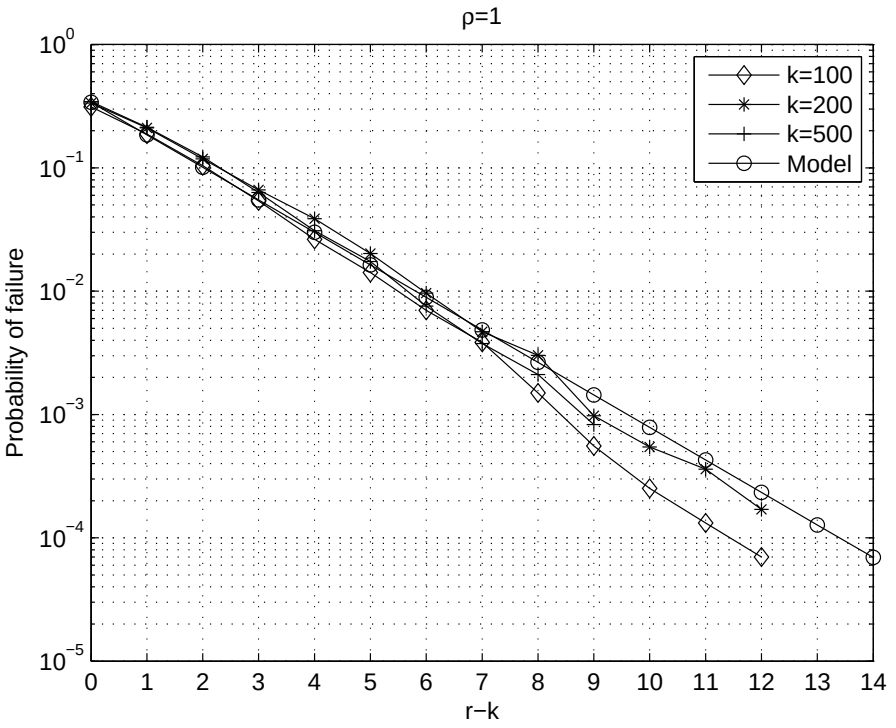
where $\rho = (n-k)/k$ denotes the ratio of parity symbols to the number of systematic symbols (input symbols). When $\beta' = 0.68$ and $\alpha' = 0.545$ accurate approximation is achieved, and the model is obtained as

$$N_{undec}(r,k,\rho) = \begin{cases} k - \frac{r}{1+\rho} & r < k \\ k \cdot 0.68 \cdot \frac{\rho}{1+\rho} \cdot (0.545)^{r-k} & r \geq k \end{cases} . \tag{4.43}$$

The results on the modeling of systematic Raptor codes are given in Figures 4.11 and 4.12. The modeling results for both systematic and non-systematic Raptor codes are quite accurate. They can be used in bit rate optimizations for joint source-channel coding systems as we used in Chapter 3.

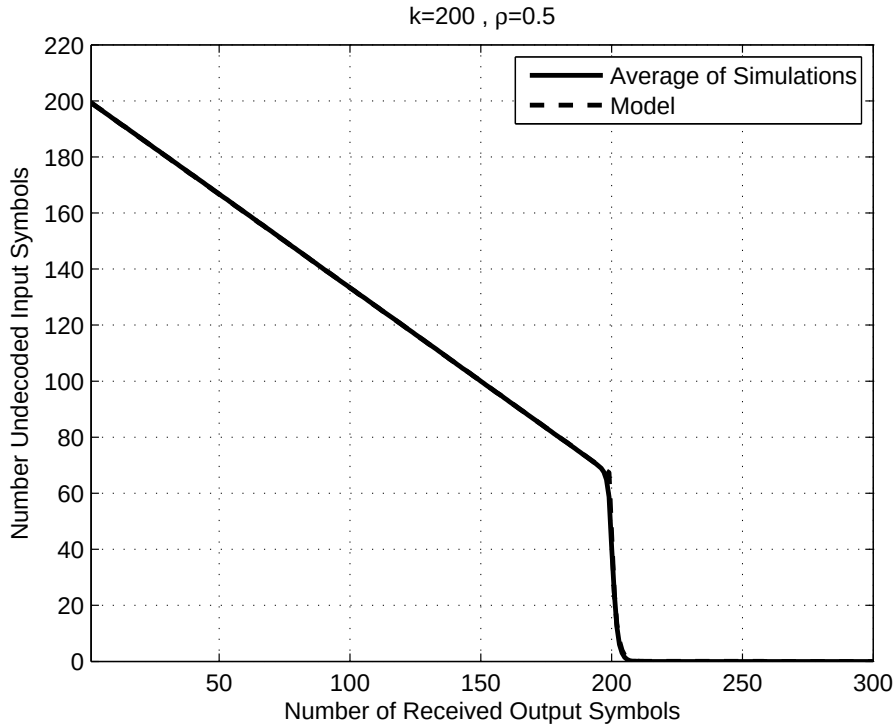


(a)

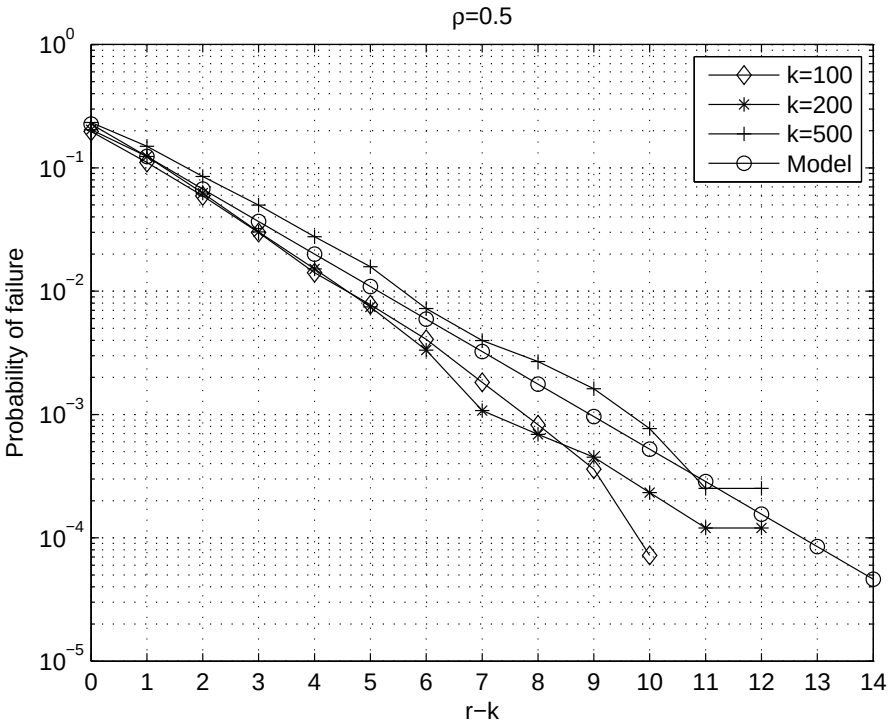


(b)

Figure 4.11: The results on modeling the performance of systematic Raptor codes for $\rho = 1.0$ (a) whole performance for $k = 200$ (b) normalized log-scale plot for $r \geq k$ for $k = 100$, $k = 200$ and $k = 500$



(a)



(b)

Figure 4.12: The results on modeling the performance of systematic Raptor codes for $\rho = 0.5$ (a) whole performance for $k = 200$ (b) normalized log-scale plot for $r \geq k$ for $k = 100$, $k = 200$ and $k = 500$

Chapter 5

Conclusions

This thesis proposes an error-resilient stereoscopic video streaming system and places focus on the analysis and modeling of the performance of Fountain codes. We demonstrate the benefits of FEC for video streaming by lossy transmission simulations. The results clearly shows the significance of quality increase with the use of FEC compared to other error resiliency tools. Motivated by this result, we use the recently proposed Raptor codes for optimal stereoscopic video streaming. Specifically, we propose an optimal streaming system for lossy channels based on three layers of stereoscopic video where each layer receives different error protection. The most important aspects are, respectively, basic partitioning of the video according to unequal importance, obtaining rate-distortion (RD) properties of these partitions, analysis of the utilized FEC scheme for UEP, and end-to-end distortion minimization. The end-to-end distortion minimization and the calculation of the corresponding encoder bit rates and protection rates are performed once for each video separately.

The partitions of video, called layers, are simply chosen according to the referencing structure. The selection of layers according to the referencing structure complicates the modeling of their RD curves due to the inherent inter-dependencies. For this purpose, we use a previously proposed RD model of monoscopic single layer video and extend it for dependent layers. The detailed results show the accuracy of RD models which can be separately used to obtain optimal stereoscopic video quality for a given bit rate in lossless scenarios. The RD modeling necessitates the use of a reliable curve fitting tool in order to determine the model parameters, because the curve fitting process is dependent on the algorithm and initial conditions. The aim in calculating the RD curves is to obtain the distortion in video quality caused by the video compression. After the calculation of RD properties of the layers we focus on the error correction scheme, namely Raptor codes. The performance model of Raptor codes that gives the residual number of unrecovered packets, plays a crucial role in the end-to-end distortion minimization, since each residual lost packet creates a distortion in the video decoder deducing the other source of distortion, namely the transmission distortion. Furthermore, we calculate the average transmission distortion caused by the loss of a single packet including all possible propagations to subsequent frames. The propagation models for stereoscopic video are heuristically extended from those of monoscopic video. We make an assumption, which is valid for low number of residual loss rates, that the total distortion of a single packet is independent of other packet losses. The distortion minimization is a non-linear optimization problem with a single constraint. When the models are obtained for a specific video, the distortion minimization can be instantly performed for any channel bandwidth and loss rate. Fortunately, the only time consuming part of this system becomes the modeling phase for a video, which has to be performed only once. The results on the proposed system are presented including comparisons with non-optimal schemes. The results demonstrate up to 2dB increase in performance when the allocated channel bit rate increases.

The distortion metric that we used is the mean squared error. Simply, we add up the encoder distortion and transmission distortion to calculate the end-to-end distortion. In case of utilizing any other metric to calculate the distortion, the proposed minimization can still yield optimal results.

Another aim in this thesis is the analysis and modeling of the performance of Fountain codes, which have become a promising candidate for the next generation communication systems. In this thesis we present detailed analysis of the performance of two types of Fountain codes, LT and Raptor codes. We describe the iterative decoding of LT codes, the belief propagation (BP) decoder, together with the ML decoder. Then, we demonstrate the performance of LT codes with BP and ML decoders. Due to the random nature of LT codes, the decoder yields different results for different sets of encoded symbols. Thus, we defined performance curve of Fountain codes as the average number of undecoded input symbols versus the number of received output symbols. Investigating the performance results, we can clearly state that LT coding is not an efficient FEC scheme for time limited applications, such as video streaming, owing to the excessive overhead for low number of output symbols. However, they can bring efficiency to the communication systems with high loss rates when they are used instead of ARQ type schemes.

The analysis and modeling of LT codes is iterative resembling their decoding process. In this thesis, we studied towards obtaining mathematical models for the performances of BP and ML decoders. The randomness of LT codes, especially with BP decoder, prevents accurate modeling. Fortunately, taking advantage of the ease of tracking the average number of edges in the decoding, the model approximates the behavior of LT BP decoder. The model for ML decoder tracks the number of linear combinations of the output symbols and achieves accurate results. We expected the model of ML decoder to be better so that it would exactly give the average performance. The difference is mainly caused by an

assumption that the expectation operation is distributable on division operation. Nevertheless, the results of the modeling of ML decoder are more accurate than that of BP decoder; this is mainly caused by the reduction in the randomness. Owing to the inefficiency of LT codes, Raptor codes are proposed. They eliminate the tail effect in the performance of LT codes by inserting a pre-coding stage, in return, complicating their analysis and modeling. We propose heuristic modeling by inspection and, indeed, obtain quite accurate models with explicit functions. Within the FP6 project 3DTV, we have implemented Raptor codes for multi-view video streamer and achieved excellent results. The encoding and decoding algorithm for Raptor codes is also very well defined in the standards and has low complexity. Raptor codes are a promising candidate for next generation communication systems.

The proposed stereoscopic video streaming system is practical and promises optimal results for lossy channels. This system can be used as a base for layered multi-view video streaming system with Raptor codes. The proposed methods are easily applicable to any layered multi-view video streaming system. The analysis of Fountain codes in this thesis provides valuable insight to understand their structure. The mathematical models of LT and Raptor codes are useful for optimal bit rate and channel rate allocation in communication systems.

Bibliography

[1] C. Shannon, "A mathematical theory of communication (parts I and II)," *Bell System Technical Journal*, vol. 27, no. Part I, pp. 379–423, 1948.

[2] R. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[3] R. Bose and D. Ray-Chaudhuri, "Further results on error correcting binary group codes," *Information and Control*, vol. 3, pp. 279–290, 1960.

[4] L. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, pp. 300–304, June 2001.

[5] R. Gallager, "Low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[6] A. Morello and V. Mignone, "DVB-S2: the second generation standard for satellite broad-band services," *Proceedings of the IEEE*, vol. 94, no. 1, pp. 210–227, 2006.

[7] F. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[8] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, 2001.

[9] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proceedings of IEEE International Conference on Communications*, vol. 2, (Geneva, Italy), pp. 1064–1070, 1993.

[10] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, pp. 56–67, 1998.

[11] M. Luby, "LT codes," in *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, (Vancouver, British Columbia, Canada), pp. 271–280, 2002.

[12] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, pp. 2551–2567, June 2006.

[13] R. Karp, M. Luby, and A. Shokrollahi, "Finite length analysis of LT codes," in *Proceedings of International Symposium on Information Theory*, (Chicago, Illinois, USA), p. 39, 2004.

[14] "Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Content delivery protocols," Tech. Rep. ETSI TS 102 472 V1.2.1, European Broadcasting Union, 2006.

[15] "Multimedia Broadcast/Multicast Service (MBMS); protocols and codecs," Tech. Rep. TS 26.346 V7.7.0, 3rd Generation Partnership Project (3GPP), 2008.

[16] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer, "Raptor forward error correction scheme for object delivery," *Internet Draft*, http://www.ietf.org/internet-drafts/draft-ietf-rmt-bb-fec-raptor-object-09.txt, June 2007.

[17] M. Liou, "Overview of the p× 64 kbit/s video coding standard," *Communications of the ACM*, vol. 34, no. 4, pp. 59–63, 1991.

[18] D. Le Gall, "MPEG: a video compression standard for multimedia applications," *Communications of the ACM*, vol. 34, no. 4, pp. 46–58, 1991.

[19] G. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, 1992.

[20] P. Lambert, W. De Neve, Y. Dhondt, and R. Van de Walle, "Flexible macroblock ordering in H. 264/AVC," *Journal of Visual Communication and Image Representation*, vol. 17, no. 2, pp. 358–375, 2006.

[21] A. Sio-Kei Im, "Unequal error protection with the H. 264 flexible macroblock ordering," *Proceedings of SPIE*, vol. 5960, p. 596032, 2005.

[22] R. Talluri, I. Moccagatta, Y. Nag, and G. Cheung, "Error concealment by data partitioning," *Signal Processing: Image Communication*, vol. 14, no. 6-8, pp. 505–518, 1999.

[23] A. Eleftheriadis and P. Batra, "Optimal data partitioning of MPEG-2 coded video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 10, pp. 1195–1209, 2004.

[24] T. Wiegand, G. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H. 264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[25] A. Smolic, P. Merkle, K. Muller, C. Fehn, P. Kauff, and T. Wiegand, "Compression of multi-view video and associated data," in *Three-Dimensional*

Television: Capture, Transmission, and Display (eds. Levent Onural and Haldun M. Ozaktas), pp. 313–350, 2007.

[26] A. Smolic, K. Mueller, P. Merkle, C. Fehn, P. Kauff, P. Eisert, and T. Wiegand, "3D Video and Free Viewpoint Video–Technologies, Applications and MPEG Standards," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, (Toronto, Canada), pp. 2161–2164, 2006.

[27] M. Luby, "Information additive code generator and decoder for communication systems," *United States Patent*, Patent Number: US 6,307,487, February 5, 1999.

[28] M. Luby, "Information additive code generator and decoder for communication systems," *United States Patent*, Patent Number: US 6,373,406, January 8, 2001.

[29] M. Luby, "Information additive code generator and decoder for communication systems," *United States Patent*, Patent Number: US 6,614,366, February 14, 2002.

[30] M. Luby, "Information additive code generator and decoder for communication systems," *United States Patent*, Patent Number: US 7,057,534, June 19, 2003.

[31] M. Luby, "Information additive code generator and decoder for communication systems," *United States Patent*, Patent Number: US 7,233,264, September 13, 2005.

[32] A. Shokrollahi, S. Lassen, and M. Luby, "Multi-stage code generator and decoder for communication systems," *United States Patent*, Patent Number: US 7,068,729, December 21, 2001.

[33] A. Shokrollahi, "Raptor codes," Tech. Rep. DF2003-06-001, Digital Fountain, Inc., June 2003.

[34] M. Luby, M. Mitzenmacher, M. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proceedings of the 29th annual ACM symposium on Theory of computing*, (Texas, USA), pp. 150–159, 1997.

[35] B. Haskell, A. Puri, and A. Netravali, *Digital video: An introduction to MPEG-2*. Kluwer Academic Publishers, 1997.

[36] K. Rijkse, "H. 263: video coding for low-bit-rate communication," *IEEE Communications Magazine*, vol. 34, no. 12, pp. 42–45, 1996.

[37] G. Cote, B. Erol, M. Gallant, and F. Kossentini, "H. 263+: video coding at low bit rates," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 7, pp. 849–866, 1998.

[38] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable H. 264/MPEG4-AVC extension," in *Proceedings of International Conference on Image Processing (ICIP)*, (Atlanta, USA), pp. 8–11, 2006.

[39] Y. Chen, P. Pandit, and S. Yea, "WD 3 Reference software for MVC," *JVT AC207*, JVT of ISO/IEC MPEG & ITU-T VCEG, Busan, October 2008.

[40] S. Shimizu, A. Vetro, and Y. Chen, "Conformance testing for MVC," *JVT AC206*, JVT of ISO/IEC MPEG & ITU-T VCEG, Busan, October 2008.

[41] A. Akbari, N. Canagarajah, D. Redmill, D. Bull, and D. Agrafiotis, "A novel H. 264/AVC based multi-view video coding scheme," in *Proceedings of 3DTV Conference*, (Kos, Greece), pp. 1–4, 2007.

[42] G. Li and Y. He, "A novel multi-view video coding scheme based on H. 264," in *Fourth International Conference on Information, Communications and Signal Processing*, vol. 1, (Singapore), pp. 493–497, 2003.

[43] C. Bilen, A. Aksay, and G. Bozdagi Akar, "A multi-view video codec based on H.264," in *Proceedings of IEEE Conference on Image Processing (ICIP)*, (Atlanta, USA), pp. 541–544, 2006.

[44] R. Fielding, "RFC2616: Hypertext Transfer Protocol – HTTP/1.1," *Internet Request For Comments (RFC)*, 1999.

[45] H. Schulzrinne, A. Rao, and R. Lanphier, "RFC2326: Real Time Streaming Protocol (RTSP)," *Internet Request For Comments (RFC)*, 1998.

[46] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RFC3550: RTP: A transport protocol for real-time applications," *Internet Request For Comments (RFC)*, 2003.

[47] J. Lou, H. Cai, and J. Li, "A real-time interactive multi-view video system," in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, (Singapore), pp. 161–170, 2005.

[48] S. Pehlivan, A. Aksay, C. Bilen, G. Akar, and R. Civanlar, "End-to-end stereoscopic video streaming system," in *Proceddings of IEEE International Conference on Multimedia and Expo.(ICME)*, (Toronto, Canada), pp. 2169–2172, 2006.

[49] N. Ozbek, B. Gorkemli, A. Tekalp, and T. Tunali, "Adaptive streaming of scalable stereoscopic video over DCCP," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, vol. 6, (Texas, USA), pp. 489–492, 2007.

[50] P. Yip, J. Malcolm, A. Fernando, K. Loo, and H. Arachchi, "Joint source and channel coding for H.264 compliant stereoscopic video transmission," in *Proceedigs of Canadian Conference on Electrical and Computer Engineering*, (Saskatoon, Canada), pp. 188–191, May 2005.

[51] J. Wagner, J. Chakareski, and P. Frossard, "Streaming of scalable video from multiple servers using rateless codes," in *Proceedings of IEEE International*

Conference on Multimedia and Expo (ICME), (Toronto, Canada), pp. 1501–1504, 2006.

[52] S. Ahmad, R. Hamzaoui, and M. Al-Akaidi, "Robust live unicast video streaming with rateless codes," in *Proceedings of the 16th International Packet Video Workshop*, (Lausanne, Switzerland), pp. 78–84, 2007.

[53] C. Hellge, T. Schierl, and T. Wiegand, "Multidimensional layered forward error correction using rateless codes," in *Proceedings of IEEE International Conference on Communications*, (Beijing, China), pp. 480–484, May 2008.

[54] T. Schierl, S. Johansen, C. Hellge, T. Stockhammer, and T. Wiegand, "Distributed rate-distortion optimization for rateless coded scalable video in mobile ad hoc networks," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, (Texas, USA), pp. 497–500, October 2007.

[55] U. Kozat and S. Ramprashad, "Unequal error protection rateless codes for scalable information delivery in mobile networks," in *Proceedings of 26th IEEE International Conference on Computer Communications (INFOCOM)*, pp. 2316–232, May 2007.

[56] S. Chang, K. Yang, and J. Wang, "Unequal-protected LT code for layered video streaming," in *Proceedings of IEEE International Conference on Communications (ICC 2008)*, pp. 500–504, May 2008.

[57] N. Thomos and P. Frossard, "Collaborative video streaming with raptor network coding," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME 2008)*, pp. 497–500, April 2008.

[58] R. Razavi, M. Fleury, and M. Ghanbari, "Block-based rateless coding for energy-efficient video streaming over bluetooth," in *Proceedings of IEEE Symposium on Computers and Communications (ISCC 2008)*, pp. 1–6, July 2008.

[59] S. Aign and K. Fazel, "Temporal and spatial error concealment techniques for hierarchical MPEG-2 video codec," in *Proceedings of IEEE International Conference on Communications (ICC)*, vol. 3, (Seattle, USA), pp. 1778–1783, June 1995.

[60] B. Wah, X. Su, and D. Lin, "A survey of error-concealment schemes for real-time audio and videotransmissions over the Internet," in *Proceedings of International Symposium on Multimedia Software Engineering*, pp. 17–24, 2000.

[61] W. Kung, C. Kim, and C. Kuo, "Spatial and temporal error concealment techniques for video transmission over noisy channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 7, pp. 789–802, 2006.

[62] "H.264 Reference Software," *http://iphome.hhi.de/suehring/tml/*.

[63] S. Argyropoulos, A. Tan, N. Thomos, E. Arikan, and M. Strintzis, "Robust transmission of multi-view video streams using flexible macroblock ordering and systematic LT codes," in *Proceedings of 3DTV Conference*, (Kos, Greece), pp. 1–4, 2007.

[64] J. Bolot and T. Turletti, "Adaptive error control for packet video in the Internet," in *Proceedings of International Conference on Internet Protocols*, (Ohio, USA), pp. 25–28, 1996.

[65] J. Kim, R. Mersereau, and Y. Altunbasak, "Error-resilient image and video transmission over the Internet using unequal error protection," *IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 121–131, 2003.

[66] W. Tan and A. Zakhor, "Video multicast using layered FEC and scalable compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 373–386, 2001.

[67] U. Horn, K. Stuhlmueller, M. Link, and B. Girod, "Robust Internet video transmission based on scalable coding and unequal error protection," *Signal Processing: Image Communication*, vol. 15, no. 1, pp. 77–94, 1999.

[68] J. Kim, R. Mersereau, and Y. Altunbasak, "Distributed video streaming using multiple description coding and unequal error protection," *IEEE Transactions on Image Processing*, vol. 14, no. 7, pp. 849–861, 2005.

[69] Y. Yuan, B. Cockburn, T. Sikora, and M. Mandal, "Efficient allocation of packet-level forward error correction in video streaming over the Internet," *Journal of Electronic Imaging*, vol. 16, p. 023012, 2007.

[70] V. Varsa, M. Hannuksela, and Y. Wang, "Non-normative error concealment algorithms," *ITU-T VCEG*, vol. 62, 2001.

[71] K. Stuhlmuller, N. Farber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 1012–1032, 2000.

[72] J. Mor, "The Levenberg-Marquardt algorithm: Implementation and theory," *Lecture Notes in Mathematics*, vol. 630, pp. 105–116, 1977.

[73] N. Ozbek, A. Tekalp, and E. Tunali, "Rate allocation between views in scalable stereo video coding using an objective stereo video quality measure," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (Hawaii, USA), pp. 1045–1048, 2007.

[74] A. Aksay, C. Bilen, E. Kurutepe, T. Ozcelebi, G. Bozdagi Akar, M. R. Civanlar, A. M. Tekalp , "Temporal and spatial scaling for stereoscopic video compression," in *Proceedings of IEEE European Signal Processing Conference (EUSIPCO)*, (Florence, Italy), September 2006.

[75] P. Gill, W. Murray, and M. Wright, "Practical optimization," in *London: Academic Press*, 1981.

[76] A. Serdar Tan, A. Aksay, G. B. Akar, and E. Arikan, "Error resilient layered steroscopic video streaming," in *Proceedings of 3DTV Conference*, (Kos, Greece), pp. 1–4, 2007.

[77] E. Maneva and A. Shokrollahi, "New model for rigorous analysis of LT-codes," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, (Seattle, USA), pp. 2677–2679, 2006.

[78] S. Sanghavi and M. LIDS, "Intermediate performance of rateless codes," in *Proceedings of Information Theory Workshop (ITW)*, (Tahoe City, California, USA), pp. 478–482, 2007.

[79] E. Hyytia, T. Tirronen, and J. Virtamo, "Optimal degree distribution for LT codes with small message length," *26th IEEE International Conference on Computer Communications (INFOCOM)*, pp. 2576–2580, 2007.

[80] A. Tarable and S. Benedetto, "Analysis of PCO raptor codes and turbo-fountain codes on noiseless channels," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, (Nice, France), pp. 416–420, 2007.

[81] P. Pakzad and A. Shokrollahi, "EXIT functions for LT and raptor codes, and asymptotic ranks of random matrices," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, (Nice, France), pp. 411–415, 2007.

[82] T. Tirronen and J. Virtamo, "Performance analysis of divided random linear fountain," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM)*, (Washington, USA), pp. 520–526, 2007.

[83] M. Luby, M. Watson, T. Gasiba, and T. Stockhammer, "Mobile data broadcasting over MBMS tradeoffs in forward error correction," in *Proceedings of the 5th International Conference on Mobile and Ubiquitous Multimedia*, (Stanford, CA, USA), 2006.