UNIVERSITY OF CALIFORNIA, SAN DIEGO

SAN DIEGO STATE UNIVERSITY

Cross-Layer Prioritized Video Transmission: Adaptive Packetization, FEC Protection and Scheduling Methods

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Engineering Science (Electrical and Computer Engineering)

by

Kashyap Kodanda Ram Kambhatla

2014

The dissertation of Kashyap Kodanda Ram Kambhatla is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Co-Chair

Chair

University of California, San Diego
San Diego State University

2014

DEDICATION

To my parents,
Hema and Narayana Prasad Kambhatla,
my wife Seethal, sisters Divya and Sindhu,
and my in-laws Dhanalaxmi and Rama Murti Paluri.

TABLE OF CONTENTS

Signature Page iii

Dedication iv

Table of Contents v

List of Figures viii

List of Tables xi

Acknowledgements xii

Vita xiv

Abstract of the Dissertation xvi

Chapter 1 Introduction 1
- 1.1 Background on Payload Adaptation 5
- 1.2 Thesis outline 8

Chapter 2 Video Quality Enhancement through Prioritized Packet Fragmentation 11
- 2.1 Introduction 11
- 2.2 Proposed Cross-Layer Fragmentation Scheme 12
  - 2.2.1 H.264 Slice and Video Packet Formation 12
  - 2.2.2 Slice Priority 13
  - 2.2.3 Video Packet Fragmentation 14
  - 2.2.4 Problem Formulation for Determining Optimal Fragment Sizes 15
  - 2.2.5 Branch and Bound (BnB) Optimization using Interval Arithmetic Analysis 22
- 2.3 Priority-agnostic vs. Priority-aware Fragmentation 25
- 2.4 Experimental Results and Discussion 27
  - 2.4.1 Simulation Setup 27
  - 2.4.2 Priority-Agnostic Fragmentation 29
  - 2.4.3 Priority-Aware Fragmentation 31
- 2.5 Acknowledgement 42

Chapter 3 Cross-Layer Prioritized Video Packetization and Error Protection 43
3.1 Introduction 43
3.2 Proposed Cross-Layer Approach 44
3.2.1 CMSE Computation/Prediction of H.264 Video Slices 45
3.2.2 H.264 Video Packet Formation 45
3.3 Expected Video Distortion Minimization 46
3.3.1 Packet Formation (PF) Block 46
3.3.2 Distortion Minimization with Optimal RCPC Code Rate Allocation (OCRA) Block 48
3.3.3 Determination of $\boldsymbol{\lambda}$ 51
3.3.4 Discarding Packets 52
3.4 Problem Formulation of other Error Protection Schemes . . 52
3.5 Performance of **DP-UEP** 53
3.5.1 Simulation Setup 53
3.5.2 Performance Comparison 54
3.6 Frame-Level **DP-UEP** Scheme 58
3.7 Frame-Level **DP-UEP** using Prediction 60
3.7.1 Generalized Linear Modeling Approach for Estimating w^l_{ovh} 61
3.7.2 PSNR and VQM Performance 64
3.8 Acknowledgement 67

Chapter 4 Scheduling Scalable Video Streaming over Time-Varying Wireless Links 68
4.1 Introduction 68
4.2 Related Work 71
4.3 System Model 75
4.3.1 Scalable Video Coding 75
4.3.2 Video Streaming System 77
4.3.3 Wireless Channel 79
4.4 Problem Formulation 81
4.4.1 EDF-based Scheme 81
4.4.2 CMSE-based Scheme 81
4.4.3 Proposed Scheme 84
4.5 Experimental Results and Discussion 87
4.5.1 Simulation Setup 87
4.5.2 Evaluation of Average Goodput and Percentage of Expired Whole Frames 89
4.5.3 Evaluation of Expired NAL Units 93
4.5.4 Evaluation of Video Quality 95
4.6 Acknowledgement 100

Chapter 5 Conclusion 101
5.1 Future work 103

Bibliography 105

LIST OF FIGURES

Figure 2.1: Cross-layer prioritized packet fragmentation approach for RTP, UDP, IP based transmission. 13
Figure 2.2: Illustration of video packet fragmentation with and without slice fragmentation. 15
Figure 2.3: Expected goodput G vs. fragment size y at (a) R =960 Kbps and different p_b, and (b) $p_b = 5 \times 10^{-5}$ and different R. 19
Figure 2.4: Slice data in Kbytes discarded per second at channel BER of 5×10^{-5}. 20
Figure 2.5: (a) Expected weighted goodput, and (b) slices discarded for Foreman encoded at R=960 Kbps and x=150 bytes. 23
Figure 2.6: (a) Expected goodput, and (b) slice discard comparisons between priority-agnostic and priority-aware fragmentation. 26
Figure 2.7: (a) Expected weighted goodput G_W, and (b) average PSNR at R= 960 Kbps, R_{CH}= 1 Mbps, and $p_b = 10^{-5}$. 27
Figure 2.8: Average PSNR and VQM achieved by priority-agnostic fragmentation for Foreman encoded at 720 Kbps. 30
Figure 2.9: Average PSNR and VQM achieved by priority-agnostic fragmentation for Foreman encoded at 960 Kbps. 30
Figure 2.10: Average PSNR and VQM achieved by priority-agnostic fragmentation for Foreman encoded at 1080 Kbps. 31
Figure 2.11: Gains achieved by 150 byte slices over 900 byte slices in priority-agnostic fragmentation for Foreman, in terms of (a) expected goodput, (b) average PSNR, and (c) average VQM. 32
Figure 2.12: Average PSNR gain achieved by priority-agnostic fragmentation over baseline system for Foreman encoded at (a) 720 Kbps and (b) 960 Kbps, and corresponding average VQM gain at (c) 720 Kbps and (d) 960 Kbps. 33
Figure 2.13: Average PSNR achieved by priority-aware fragmentation for Foreman encoded at (a) 720 Kbps, (c) 960 Kbps, and (e) 1080 Kbps, and corresponding average VQM at (b) 720 Kbps, (d) 960 Kbps, and (f) 1080 Kbps. 37
Figure 2.14: 126th frame of Foreman encoded at 720 Kbps and BER of 5×10^{-5} in (a) baseline system: average PSNR=20.3 dB, average VQM=0.83, (b) priority-agnostic fragmentation: average PSNR=25.2 dB, average VQM=0.68, and (c) priority-aware fragmentation: average PSNR=29.9 dB, average VQM=0.46. 40
Figure 2.15: 126th frame of Silent encoded at 720 Kbps and BER of 5×10^{-5} in (a) baseline system: average PSNR=22 dB, average VQM=0.81, (b) priority-agnostic fragmentation: average PSNR=28.5 dB, average VQM=0.68, and (c) priority-aware fragmentation: average PSNR=33.3 dB average VQM=0.47. 41

Figure 3.1: Flow diagram of proposed cross-layer system for RTP/UDP/IP video transmission. 44
Figure 3.2: Block diagram of proposed dynamic programming approach to estimate packet sizes and their FEC. 47
Figure 3.3: Packet formation in PF block: linearity model for packet CMSE and size. 48
Figure 3.4: Average video PSNR (dB) and corresponding average VQM comparison computed over 100 realizations of each AWGN channel for Foreman:(a),(b), and Silent:(c),(d). 55
Figure 3.5: Average number of slices discarded per GOP in **EEP-slice-ENH**, **Dual15**, and **DP-UEP** for Foreman. 57
Figure 3.6: Distribution of the final output bits for Foreman at 3 dB channel SNR in **EEP-slice-ENH**, **Dual15**, and **DP-UEP** schemes. . . . 57
Figure 3.7: Cumulative distribution function (CDF) for the binned observations and fitted distributions of w^l_{ovh}. 63
Figure 3.8: Average video PSNR (dB) and average VQM comparison computed over 100 realizations of each AWGN channel for Akiyo: (a),(d), Foreman: (b),(e) and Stefan: (c), (f). The error-free PSNR values are: 46.5 dB for Akiyo, 37.3 for Foreman, and 29.7 for Stefan. 66

Figure 4.1: (a) Hierarchical prediction structure, and (b) motion-compensated prediction for MGS layers with key pictures. 76
Figure 4.2: Average R-D characteristic curves in terms of (a) CMSE, and (b) IMSE for different temporal and quality layers. 77
Figure 4.3: Video streaming system over a wireless link with time-varying throughput. 78
Figure 4.4: (a) Sampling curve for Table Tennis $R_v(t)$, and outgoing video bits supported by the channel, (b) close up of the sampling curve between t=15 sec and t=16 sec. 79
Figure 4.5: Video streaming timing diagram demonstrating frame deadlines and time-to-expiry computation. 80
Figure 4.6: Sample iterations of our proposed dynamic programming algorithm over a window of frames at the video server. 86
Figure 4.7: Average goodput of the EDF-based, CMSE-based, and proposed scheduling schemes. 90
Figure 4.8: Percentage of expired whole frames in EDF-based, CMSE-based, and proposed schemes over 120 random channel realizations. . . . 91
Figure 4.9: Percentage of expired whole frames in different temporal layers of EDF-based, CMSE-based, and proposed schemes over 120 random channel realizations. 91
Figure 4.10: Total percentage of expired NAL units in 120 random channel realizations for EDF-based, CMSE-based, and proposed schemes. 93

Figure 4.11: Percentage of expired NAL units in different SNR quality layers from 120 random channel realizations of EDF-based, CMSE-based, and proposed schemes. 94
Figure 4.12: Percentage of expired NAL units in different temporal layers from 120 random channel realizations of EDF-based, CMSE-based, and proposed schemes. 95
Figure 4.13: Average video PSNR of the EDF-based, CMSE-based, and proposed schemes over 120 random channel realizations. 96
Figure 4.14: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.1s. 97
Figure 4.15: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.1s. 97
Figure 4.16: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.1s. 98
Figure 4.17: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.4s. 98
Figure 4.18: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.4s. 99
Figure 4.19: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.4s. 99

LIST OF TABLES

Table 2.1: Optimal number of slices in high and low priority fragments after running BnB algorithm. 24
Table 2.2: (a) PSNR and (b) VQM gains of priority-aware over priority-agnostic fragmentation with modified drop-tail slice discard for Foreman at 720, 960, and 1080 Kbps. 35
Table 2.3: (a) PSNR and (b) VQM gain due to proposed slice discard for Foreman at 960 Kbps (1080 Kbps). 35
Table 2.4: (a) PSNR and (b) VQM gains of priority-aware over priority-agnostic fragmentation (without slice fragmentation) for Foreman at 960 Kbps (1080 Kbps). 38
Table 2.5: PSNR gains of priority-aware over priority-agnostic fragmentation (without slice fragmentation) for Silent at 960 Kbps (1080 Kbps). 38
Table 2.6: Average PSNR gain of priority-aware (with slice fragmentation) over priority-agnostic fragmentation for Foreman (Silent) at (a) 720 Kbps, (b) 960 Kbps, and (c) 1080 Kbps. 39
Table 2.7: Average VQM gain of priority-aware (with slice fragmentation) over priority-agnostic fragmentation for Foreman (Silent) at (a) 720 Kbps, (b) 960 Kbps, and (c) 1080 Kbps. 42

Table 3.1: Goodness of fit statistics for maximized likelihood function based on three information criteria. 62
Table 3.2: Final generalized linear model factors and coefficients including interactions. 64

Table 4.1: Bit rates (Kbps) of sub-streams of 720×480 resolution (a) Table Tennis and (b) Stefan. 88
Table 4.2: (a) Average NAL unit sizes (bytes) and (b) average CMSE values of Table Tennis. 89
Table 4.3: (a) Average NAL unit sizes (bytes) and (b) average CMSE values of Stefan. 89
Table 4.4: Average normalized CMSE values of 720×480 resolution (a) Table Tennis and (b) Stefan. 92

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my deepest gratitude to my advisors, Prof. Pamela Cosman and Prof. Sunil Kumar, for their support, guidance, and invaluable advice for the past several years.

Prof. Pamela Cosman was instrumental in providing the right research direction and making sure that I got all the technical advice and support to achieve my goals. Her constructive criticism and encouragement spurred me on to better understand and improve myself. I am also thankful to her for always finding time to read and correct all my research papers. I learned a lot from her vast knowledge and she will always be a great source of inspiration.

I am grateful to Prof. Sunil Kumar for introducing new areas of research to me. He was instrumental in helping me achieve my goals. I am thankful to him for arranging the financial support throughout my graduate studies and for providing infinite research opportunities and invaluable technical advice. He was not only a good Ph.D. advisor but also a good mentor and a friend.

I am also thankful to my other committee members, Prof. Truong Nguyen, Prof. Fred Harris and Prof. Geoff Voelker, for providing helpful comments and suggestions during my Ph.D. I am thankful to all my fellow researchers Mayank Tiwari, Seok-Ho-Chang, Ting-Lan Lin, Hobin Kim, and Suayb Arslan in the Information Coding Laboratory at UCSD, and Srinivas Alla, Gene Ko, Senthil Kumar, Siddharth Khimsara, and Deepak Rawat in the Multimedia Wireless and Networks Research Laboratory at SDSU for the friendly and dynamic research atmosphere.

I thank my sisters Divya and Sindhu, for their unwavering encouragement. I thank my in-laws Dhanalaxmi and Rama Murti Paluri for their faith in me and their guidance, and valuable advice. I thank my wife, Seethal, for going through both the good and bad times with me throughout my Ph.D. It would have been impossible to complete this journey without her persistent assistance and her immense faith in me.

Finally, I would like to thank my father Narayana Prasad and my mother Hema for their unconditional love and support through all these years. They provided me the best education possible with their sacrifices at various stages of life. They have instilled good virtues and principles which have helped me in this endeavor. I will always be indebted to them.

Chapter 2 of this dissertation contains material that appears in Kashyap K. R. Kambhatla, S. Paluri, S. Kumar, and P. Cosman, and “Wireless H.264 Video Quality Enhancement through Optimal Prioritized Packet Fragmentation”, *IEEE Transactions on Multimedia*, Vol. 14, No. 5, pp. 1480-1495, Oct. 2012 and Kashyap K. R. Kambhatla, S. Kumar, and P. Cosman, “Prioritized Packet Fragmentation for H.264 Video”, *IEEE International Conference on Image Processing*, Brussels, Belgium, pp. 3233-3236, 11-14 Sept. 2011. I was the primary author and co-author Seethal Paluri had contributed to a part of the work. Co-authors Dr. Pamela Cosman and Dr. Sunil Kumar directed and supervised the research which forms the basis for Chapter 2.

Chapter 3 of this dissertation contains material that appears in Kashyap K. R. Kambhatla, S. Kumar, and P. Cosman, “H.264/AVC Video Packet Aggregation and Unequal Error Protection for Noisy Channels”, *IEEE International Conference on Image Processing*, Orlando, Florida, pp. 1649-1652, Sept. 30 - Oct. 3, 2012 and Kashyap K. R. Kambhatla, S. Paluri, S. Kumar, P. Cosman, “Cross-Layer Prioritized H.264 Video Packetization and Error Protection Over Noisy Channels”, EURASIP Journal on Wireless Communication and Networking, in review. I was the primary author and and co-author Seethal Paluri had contributed to a part of the work. Co-authors Dr. Pamela Cosman and Dr. Sunil Kumar directed and supervised the research which forms the basis for Chapter 3.

Chapter 4 of this dissertation contains material that will appear in conference and/or journal publications. I am the primary author and the co-authors Dr. Pamela Cosman and Dr. Sunil Kumar directed and supervised the research which forms the basis for Chapter 4.

VITA

2004	B.Tech., *Honors*, Electronics and Computer Engineering (Electronics and Communications), Jawaharlal Nehru Technological University, Hyderabad, India.
2004-2006	Graduate Research Assistant, Clarkson University, Potsdam, NY, USA.
2006	M.S., Electrical and Computer Engineering (Wireless Communication and Networking), Clarkson University, Potsdam, USA.
2006-2007	Graduate Research Assistant, University of Southern California, Los Angeles, USA.
2007-2008	Graduate Teaching Assistant, Computational Science Research Center, San Diego, USA.
2008-2013	Graduate Teaching Assistant, Joint Doctoral Program, San Diego State University, San Diego, USA.
2014	Ph.D., Engineering Science (Electrical and Computer Engineering), University of California San Diego, and San Diego State University, USA.

PUBLICATIONS

Kashyap K. R. Kambhatla, Sunil Kumar, and Pamela Cosman, “Prioritized Packet Fragmentation for H.264 Video”, *IEEE International Conference on Image Processing*, Brussels, Belgium, pp. 3233-3236, 11-14 Sept. 2011.

Kashyap K. R. Kambhatla, Seethal Paluri, Sunil Kumar, and Pamela Cosman, “Wireless H.264 Video Quality Enhancement through Optimal Prioritized Packet Fragmentation”, *IEEE Transactions on Multimedia*, Vol. 14, No. 5, pp. 1480-1495, Oct. 2012.

Kashyap K. R. Kambhatla, Sunil Kumar, and Pamela Cosman, “H.264/AVC Video Packet Aggregation and Unequal Error Protection for Noisy Channels”, *IEEE International Conference on Image Processing*, Orlando, Florida, pp. 1649-1652, Sept. 30 - Oct. 3, 2012.

Seethal Paluri, Kashyap K. R. Kambhatla, Sunil Kumar, Barbara Bailey, Pamela Cosman, and John Matyjas, “Predicting Slice Loss Distortion in H.264/AVC Video for Low Complexity Data Prioritization”, *IEEE International Conference on Image Processing*, Orlando, Florida, pp. 689-692, Sept. 30 - Oct. 3, 2012.

Kashyap K. R. Kambhatla, Seethal Paluri, Sunil Kumar, and Pamela Cosman, "Cross-Layer Prioritized H.264 Video Packetization and Error Protection Over Noisy Channels", *EURASIP Journal on Wireless Communication and Networking*, in review.

Seethal Paluri, Kashyap K. R. Kambhatla, Sunil Kumar, Barbara A. Bailey, and Pamela Cosman, "A Low Complexity Model for Predicting Slice Loss Distortion in H.264/AVC for Real-Time Applications", *IEEE Transactions on Consumer Electronics*, in review.

Siddharth Khimsara, Kashyap K. R. Kambhatla, Sunil Kumar, and John D. Matyjas, "AM-AOMDV: Adaptive Multi-metric Ad-Hoc On-Demand Multipath Distance Vector Routing", *Springer LNICST ADHOCNETS'09*, Niagara Falls, Canada, pp. 884-895, Sept. 2009.

Sunil Kumar, Siddharth Khimsara, Kashyap K. R. Kambhatla, Kalyani Girivanesh, John D. Matyjas, and Michael Medley, "Robust On-Demand Multipath Routing with Dynamic Path Upgrade for Delay-Sensitive Data over Ad Hoc Networks", *Hindawi Journal of Computer Networks and Communications*, Vol. 2013, pp. 1-13, Feb. 2013.

FIELDS OF STUDY

Major Field: Electrical and Computer Engineering
Studies in Communication Theory and Systems.
Professors Pamela Cosman and Sunil Kumar

ABSTRACT OF THE DISSERTATION

Cross-Layer Prioritized Video Transmission: Adaptive Packetization, FEC Protection and Scheduling Methods

by

Kashyap Kodanda Ram Kambhatla

Doctor of Philosophy in Engineering Science (Electrical and Computer Engineering)

University of California, San Diego, 2014
San Diego State University, 2014

Professor Pamela Cosman, Chair
Professor Sunil Kumar, Co-Chair

The quality of H.264/AVC compressed video delivery over time-varying and error-prone wireless channels is affected by packet losses. To support quality of service (QoS) for video delivery over wireless networks cross-layer schemes have been discussed in the literature. We introduce a cross-layer priority-aware packet fragmentation scheme at the medium access control (MAC) layer to enhance the quality of pre-encoded H.264/AVC compressed bitstreams over bit-rate limited error-prone links in wireless networks. Larger fragments are more likely to be in error but smaller fragments require more overhead. The H.264 slices are classified in four priorities at the encoder based on their cumulative mean square error (CMSE) contribution towards the received video quality. The slices of a priority class in each frame are aggregated into video packets of corresponding priority at the application (APP) layer. We derive the optimal fragment size for each priority class which achieves the maximum expected weighted goodput at different encoded video bit rates, slice sizes and bit error rates. Priority-aware packet fragmentation invokes slice discard in the buffer due to channel bit rate constraints on allocating fragment header bits. We propose a slice discard scheme using frame importance and slice CMSE contribution to control error propagation effects. Packet fragmentation is then extended to slice fragmentation by modifying the conventional H.264 decoder to handle partial slice decoding. Priority-aware slice fragmentation combined with the proposed slice discard scheme provides considerable peak signal-to-noise ratio (PSNR) and video quality metric gains as compared to priority-agnostic fragmentation.

Distortion due to channel errors can be alleviated by assigning stronger channel code rates, at the cost of reduced rate for source coding. Besides MAC layer fragmentation, aggregating H.264/AVC slices at the APP layer to form video packets with sizes adapted to their importance can also improve transmission reliability. We present a cross-layer dynamic programming (DP) approach to minimize the expected received video distortion by jointly addressing the priority-adaptive packet formation at the APP layer and rate compatible punctured convolutional (RCPC) code rate allocation at the physical layer for pre-encoded prioritized slices of each group of pictures (GOP). Our scheme discards some low priority slices in order to improve protection to more important slices and meet the channel bit-rate limitations, whenever necessary. Simulation results show that our proposed approach significantly improves received video quality compared to other error protection schemes. Further, we extend our cross-layer DP-based scheme to slices of each frame by predicting the expected channel bit budget per frame for real-time transmission. The prediction uses a generalized linear model developed over the parameters - CMSE per frame, channel SNR, and normalized compressed frame bit budget - determined over a video dataset that spans high, medium and low motion complexity. This predicted frame bit budget is used to derive the packet sizes and their corresponding RCPC code rates for transmission using our DP-based approach. Simulation results show good correlation with the results of our DP-based scheme applied over the GOP.

Unique characteristics of video traffic, such as the temporal and spatial dependencies between different video frames and their deadline constraints, pose a challenge in supporting the video quality rendered to the clients over time-varying, bandwidth-limited channels. Scalable Video Coding (H.264/SVC) enables the transmission and decoding of partial bit streams to provide video services with lower temporal or spatial resolutions or reduced fidelity while retaining a reconstruction quality that is high relative to the rate of the partial bit streams. We propose a sliding-window based flow control for scheduling the network abstraction layer (NAL) units in the post-encoding buffer of the streaming server for a real-time scalable video transmission scenario over a fast time-varying channel. Our scheduling scheme considers the importance of the NAL unit in terms of (*i*) its CMSE distortion contributed to the received video quality, (*ii*) its size in bits, and (*iii*) its time-to-expiry in seconds. The scheduling problem of determining the appropriate order of transmission is formulated as a 0-1 knapsack problem and a DP solution is proposed which runs in polynomial time. Our scheduling approach significantly reduces the number of whole frames discarded as compared to (a) a CMSE-based scheme which considers the importance of the NAL units only in terms of their CMSE contribution, and (b) the earliest deadline first scheme which minimizes the dwelling time of the NAL units in the post-encoding buffer. Simulation results show significant PSNR gains for different video sequences at different pre-roll delays.

Chapter 1

Introduction

Video streaming over wireless networks invokes a strong interest for many delay sensitive and bandwidth-intensive applications, and an increasing number of systems are being deployed. The gaming applications involving real-time multimedia transmission among various players are growing rapidly. Video streaming and sharing of news, TV, movies, and sports clips to mobile phones is now widely available on popular applications such as YouTube, Netflix, and Hulu. Video conferencing applications, such as Cisco telepresence, and Google Hangout, are slowly replacing the need for business travel. For surveillance applications, cameras can be flexibly and cheaply installed, if a wireless network provides connectivity. A wireless local area network (WLAN) now connects various audiovisual entertainment devices in a home and allows them to interact with one another. Also in search-and-rescue operations, real-time audiovisual communication over wireless ad-hoc networks can save lives. Existing wireless networks, however, provide only limited and time-varying quality of service (QoS) support for the above applications [1].

H.264 advanced video coding (AVC) [2] and scalable video coding (SVC) [3] are the most widely used video compression standards jointly developed by the ITU and ISO. Compressed video transmission is vulnerable to packet losses in wireless networks due to network and channel impairments. Lost video packets induce different levels of quality degradation due to temporal and spatial dependencies in the compressed bitstream. An important problem which affects video quality is error propagation where an error in a reference frame propagates to future reconstructed frames which are predicted from that reference frame. This problem has led to the design of error-resiliency features such as flexible macroblock ordering (FMO), data partitioning, and error concealment schemes in H.264 [2, 4, 5].

Though H.264 error-resiliency features reduce the distortion from packet losses, they are still decoupled from various network-centric QoS provisions. QoS support involves several areas, ranging from applications, terminals, and networking architectures to network management, business models, and finally the main target, end users [6]. Enabling QoS in an environment involving mobile hosts under different wireless access technologies is very challenging, since available resources (e.g., bandwidth, battery life, etc.) in wireless networks are scarce and dynamically change over time. Since the capacity of the channel in a wireless network varies randomly with time, providing deterministic QoS (i.e., zero QoS violation probability) for video is not only difficult but will also likely result in conservative guarantees and waste of resources. Hence, statistical QoS guarantees in terms of received video quality, goodput based on successfully received data, probability of packet loss, and packet delay have gained importance. There are several fundamental challenges in supporting the end-to-end QoS for video delivery over wireless networks [6–8]:

1. QoS support depends on a wide range of technological aspects, including video coding, high-performance physical and link layer support, efficient packet delivery, congestion control, error control, and power control.

2. Different applications have diverse QoS requirements in terms of data rates, delay bounds, and packet loss probabilities. For example, unlike non-real-time data packets, video services are sensitive to packet delivery delay but can tolerate some transmission errors and even frame losses.

3. Different types of networks inherently have different characteristics, usually referred to as network heterogeneity. The internet is based on Internet Protocol (IP) (designed for wireline environment), which basically only offers best-effort services. The network conditions, such as bandwidth, packet loss ratio, delay, and delay jitter, vary over time in a wireless environment. An important characteristic of current wireless networks is that the heterogeneous wireless access technologies co-exist, such as WLAN access, 3G/4G cellular access, and

Bluetooth. Bit-error rate (BER) in a wireless network is much higher than in the wireline network. Moreover, link layer error control schemes, such as automatic repeat request (ARQ), are widely used to overcome wireless channel errors. This further increases the dramatic variation of bandwidth and delay in wireless networks. To make things even more complicated, the packet loss in wireless networks can be caused by either congestion leading to buffer overflow or by a noisy channel leading to packet errors.

4. There is dramatic heterogeneity among end users in terms of latency requirements, video visual quality, processing capabilities, power, and bandwidth. It is thus a challenge to design a delivery mechanism that not only achieves efficient resource utilization but also meets the heterogeneous requirements of the end users.

To address the above challenges, the QoS requirement should be supported in all components of the video delivery system using a cross-layer perspective, which include (a) QoS provisioning from networks, (b) scalable and/or prioritized video presentation from applications, and (c) network adaptive congestion/error/power control. To deliver the best end-to-end performance for such wireless systems, video coding, reliable transport and wireless resource allocation must be considered jointly, thus moving from the traditional layered system architecture to a cross-layer design. Broadly, this dissertation addresses cross-layer QoS issues for video packet delivery over wireless links through: (1) prioritized transmission control schemes that can derive and adjust the bit-budget for prioritized video data, and (2) cross-layer QoS adaptation that can optimally choose statistical QoS guarantees for each video priority class of a prioritized transmission system so as to provide better video quality.

Adaptation of packet size and forward error correction (FEC) are two well-known techniques to combat packet loss due to channel impairments. In this dissertation, we use them as QoS adaptation techniques for prioritized video data. Packet size adaptation can be carried out at different layers such as the application (APP), transport, and medium access control (MAC) layers. FEC adaptation can be carried out at the APP and physical (PHY) layers.

Packet segmentation and reassembly, carried out at the transport layer of the source and gateway nodes to comply with the maximum packet size requirements of intermediate networks [9, 10], cannot efficiently adapt to the varying channel conditions at the intermediate nodes. Furthermore, video streaming uses real-time transport protocol (RTP) and user datagram protocol (UDP), where the transport layer is less important for error protection and bandwidth adaptation [1]. Packet fragmentation at the MAC layer is primarily done to adapt the packet size to the channel error characteristics, in order to improve the successful packet transmission probability and reduce the cost of packet retransmissions. MAC layer fragmentation and retransmission also avoid costly transport layer retransmissions [11–16]. Fragmentation calls for a trade-off between reducing the total number of overhead bits by using large fragments and reducing the transmission error rate by using small fragments. However, maximum throughput does not guarantee the minimum video distortion at the receiver due to the following reasons - *First*, unlike data packets, loss of H.264 compressed video packets induces different amounts of distortion in the received video. Therefore the fragment size should be adaptive to the packet priority. However, existing payload (i.e., packet size) adaptation schemes in the literature do not consider the distortion contribution of the packet. *Second*, conventional packet fragmentation schemes discard a packet unless all its fragments are received correctly. However, video data is loss tolerant and a packet can be partially decoded even when some of its fragments are lost. Also real-time video transmission is delay-sensitive and retransmission of corrupted fragments may not be feasible. Packet size adaptation can be carried out at the APP layer by aggregating the smaller-sized network abstraction layer (NAL) units belonging to the different priority classes into packets of different sizes. However, there is an upper bound on the size of the APP layer packets known as maximum transmission unit (MTU) size for wireless networks.

The PHY layer applies FEC to video packets to combat low channel signal-to-noise ratio (SNR). Past research in the literature proposed optimizing joint source channel coding for video transmission at the APP layer [17–20]. However, FEC code rates have to be derived from channel characteristics such as noise level variation and channel bit rate limitation. Recent research has demonstrated the promise of cross-layer protocols for supporting the QoS demands of multimedia applications over wireless networks [21–23]. Van der Schaar et al. [22] discuss different cross layer solutions and extend the MAC-centric approach to demonstrate that the joint APP-MAC-PHY approach is best suited for transmitting multimedia (e.g., video streaming) over wireless networks. The joint APP-MAC-PHY cross-layer interface is desirable to achieve our objective of QoS adaptation by using the channel noise information, bit rate constraints, and network packet size limitation.

1.1 Background on Payload Adaptation

MAC frame length control has been studied in the past for different purposes such as (i) maximizing wireless network or user or link throughput [24–27], (ii) optimizing energy efficiency and transmission range [11, 28, 29], (iii) dynamic multi-rate link adaptation [30], and (iv) goodput enhancement, delay and retransmission control [31]. Yin et al. [24] determine the feasible packet size which achieves maximum network saturation throughput for a WLAN operating in the IEEE 802.11 distributed coordination function mode. The channel conditions are assumed to be constant throughout the network and the overhead due to retransmission bounds at the MAC layer is taken into consideration. The derived packet size is fixed for every user in the network. Throughput enhancement over time-varying Ricean fading channels in WLANs is studied in [25]. The authors dynamically vary the MAC fragmentation threshold based on the ratio of transmission rate allowed by the receiver (based on its channel conditions) to the sender's packet generation rate. Though a lower fragmentation threshold provides better throughput, this scheme does not consider the effect of additional overhead bits and the back-off time on packet delay. A cross-layer design between the MAC and PHY layers in WLANs is studied in [26]. The optimal packet size is derived for direct sequence spread spectrum and frequency hopping spread spectrum systems by using the channel BER as an interface parameter. The performance of ARQ combined with dynamic packet fragmentation is studied during wireless channel failures (i.e., unavailability of channel) for highly concentrated Gaussian or exponentially distributed data transmitted over distributed computing systems [27]. The authors show that MAC retransmissions of packets of different sizes result in power-law delays and poor utilization of network resources. They propose aggregating small packets into larger ones and using dynamic fragmentation depending on the channel availability period to improve network throughput.

The throughput enhancement in the above schemes [24–27] is observed for poor channel conditions at the cost of longer packet delay. Adaptive MAC frame length control has also been used for increasing energy efficiency in [28] and latency control for meeting packet delay requirements in real-time environments on test beds in [11, 29]. Goodput analysis combined with link rate adaptation for IEEE 802.11a is carried out as a function of the payload size in [30]. This scheme uses an auto rate fallback mechanism which alternates between 1 Mbps and 2 Mbps PHY rates depending on the result of the timeout function and the missed ACK frames. The goodput is computed as the ratio of the payload transmission time to the total time needed to transmit a data packet. Another dynamic fragmentation scheme for goodput enhancement is discussed in [31]. In the above schemes, a successful data packet reception requires the successful transmission/retransmission of each packet fragment. This introduces considerable delay which may not be suitable for real-time multimedia applications [19].

Lately, some cross-layer packet aggregation and fragmentation schemes have been proposed for enhancing H.264 compressed video transmission over wireless networks [32–35]. Fallah et al. [32] proposed the fragmentation and aggregation of H.264 NAL units in order to enhance the quality of the decoded video stream over IEEE 802.11 WLAN. They showed that the video quality is increased when the fragmentation is done at the APP layer through slicing the video compared to when fragmentation is done at the MAC layer. However, they did not combine the APP layer slicing and aggregation with MAC layer fragmentation and also did not consider the packet priorities. Connie et al. [33] extended the idea of APP layer fragmentation proposed in [32] to 3G UMTS networks in both uplink and downlink transmissions. The extended profile of H.264 divides the video data into three partitions with different levels of importance. In order to support the QoS requirements of a H.264 bitstream, Ksentini et al. [34] mapped the IEEE 802.11e MAC access categories to these different data partitions. Fallah et al. [35] extended the idea in [34] and employed a controlled access phase scheduling in the IEEE 802.11e hybrid coordination function controlled channel access mode.

Packet headers and protocol layer overhead reduce the effective throughput.

The need for adapting the payload length and data rate are discussed in [36]. To address the variation in network conditions, solutions for adaptive packet size adjustments at the APP layer have been discussed in [12,14,16,37–42]. The effect of packet size on the loss rate and delay characteristics in a wireless real-time application was studied in [14]. It was shown that APP level packet size optimization could facilitate efficient usage of wireless network resources, improving the service provided to all end users sharing the network.

Choi et al. [37] designed cross-layer schemes to study the effect of optimal packet size, MAC layer retransmissions, and APP layer FEC on multimedia delivery over wireless networks. They noted that the packet size is tightly related to the packet delay and channel conditions. An algorithm that allows an ARQ protocol to dynamically optimize the packet size based on the wireless channel bit error rates was proposed in [12]. Lee et al. [16, 38] developed an analytic model to evaluate the impact of channel BER on the quality of streaming a MPEG-4 video with fine granular scalability. They proposed a video transmission scheme, which combines the adaptive assignment of packet size with unequal error protection (UEP) to increase the end-to-end video quality.

Shih [39,42] proposed a scheme which integrated the packet size control mechanism with the optimal packet-level FEC in order to enhance the efficiency of FEC over wireless networks. Both the degree of FEC redundancy and the transport packet size were adjusted simultaneously in accordance with a minimum bandwidth consumption strategy to transmit video frames with delay bound and target frame error rate constraint. Lin et al. [40] formulated an optimization problem to minimize the required resource units for a single user by adjusting payload length, modulation, block size, and code rate for wireless channels. An adaptive packet and block length FEC control mechanism is discussed in [41]. Lin and Cosman [43] studied code rate allocation with slice discarding for pre-encoded H.264 video slices of a group of pictures (GOP). Each slice consisted of a horizontal row of macroblocks and was considered to be an independent packet.

In [36], authors present a mathematical framework to maximize a single user throughput by using the symbol rate, the packet length, and the constellation size of the modulation. In [44, 45], authors provide a theoretical framework without retransmission to optimize single user throughput by adjusting the source bit rate and payload length as a function of channel conditions. However, the maximal throughput transmission does not ensure the packet error rate (PER) requirement. A cross-layer design considering retransmission is shown in [46]. Authors optimize the length of payload and suggest the associated physical transmission modes, which include modulation and coding scheme, for a given channel SNR.

1.2 Thesis outline

Chapter 2 of this dissertation contains material that appears in publications [47] and [1]. We address the problem of assigning optimal fragment sizes to the individual priority packets at the MAC layer maximizing the expected weighted goodput within the channel bit-rate limitations and under known link conditions. The proposed cross-layer approach considers real-time streaming of pre-encoded H.264 video streams. The video slices are classified into four priority classes based on the distortion contributed by their loss to the received video quality. The slices of a priority class in each frame are aggregated into corresponding video packets whose size is bounded by the network MTU.

The scheme provides higher transmission reliability to the high priority packets by using smaller fragments, at the expense of *(i)* allowing larger fragment sizes for the low priority packets, and *(ii)* discarding some low priority packets to meet the channel bit-rate limitations, when necessary. We also propose a slice discard scheme at the MAC layer based on the frame importance and the CMSE contribution of the slice. Packet fragmentation is extended to slice fragmentation by modifying the conventional H.264 decoder to handle partial slice decoding, and use of various slice sizes. We show that adapting fragment sizes to the packet priority classes reduces the overall expected video distortion at the receiver. Our scheme does not assume retransmission of lost fragments and packets.

Chapter 3 of this dissertation contains material that appears in publications [48] and [49]. Our objective here is minimizing the expected received video distortion by jointly optimizing the packet sizes at the APP layer and estimating their FEC code rates to be allocated at the PHY layer for noisy channels. Some low priority slices are also discarded in order to increase the protection to more important slices and meet the channel bit-rate limitations. The proposed scheme ensures that higher priority slices which contribute more distortion are sent in smaller packets with stronger FEC coding. At the same time, it also efficiently controls the overhead incurred from the total protocol header bits associated with the formed packets. The simulation results show that the proposed scheme efficiently transmits video over noisy channels.

To avoid the delays associated with optimizing the packet sizes and their associated FEC code rates for entire slices of a GOP, we extend the proposed scheme to work on each frame independently by predicting its expected channel bit budget using a generalized linear model (GLM). The GLM is developed over the factors (a) normalized CMSE per frame, (b) channel SNR, and (c) normalized compressed frame bit budget allocated by the H.264 encoder. The three factors are determined from a video dataset that spans high, medium and low motion complexity. Further, to avoid the complexity associated with computing the CMSE distortion contributed by a video slice, we use our low-complexity GLM scheme for predicting the slice CMSE [50].

In Chapter 4, we propose a slice CMSE and deadline aware scheduling algorithm which exploits the temporal scalability (frame rate control) and SNR scalability (i.e., quality control) of a H.264/SVC compressed bit stream, and derives a subset (i.e., scalable) bit stream for transmission over a wireless link with time-varying bit rate. The subset bit stream provides graceful degradation in bad channel conditions. Our proposed scheduling algorithm tries to prevent whole frame losses by taking into consideration the relative importance and time-to-expiry (TTE) of the NAL units of different temporal and SNR quality layers.

We propose a sliding window based flow control at the post-encoding buffer of the streaming server. The flow control determines how many and which particular NAL units, from a window of temporal and quality layers, are to be scheduled for transmission during every transmission time interval (TTI). The set of scheduled NAL units improve the received video quality for the available channel resources. The optimization problem of maximizing the expected received video quality is reduced to maximizing the product of the normalized CMSE value with the inverse of the TTE value. The TTE value of a NAL unit varies with the channel conditions, and the number and sizes of the NAL units scheduled to be transmitted before it. Our proposed algorithm effectively trades off the importance of the NAL units with their deadlines and determines a good transmission order for the NAL units in the sliding window.

We study the effect of the scheduled NAL units on the received video quality through simulations. We compare the performance of our proposed approach to (*i*) an earliest deadline first (EDF) [51] motivated scheduling scheme, which has been used in the recent related literature [52–55], and (*ii*) a scheme where the NAL units in the sliding window are scheduled based only on their CMSE contribution.

In Chapter 5, we summarize the contribution of this dissertation, discuss the various open problems and potential future work in this direction.

Chapter 2

Video Quality Enhancement through Prioritized Packet Fragmentation

2.1 Introduction

In this chapter, we propose a cross-layer approach for real-time streaming of the pre-encoded H.264 video streams. Under known link conditions, we address the problem of finding the optimal fragment sizes for the individual priority packets at the MAC layer to maximize the expected weighted goodput within the channel bit-rate limitations. The video slices are classified into four priority classes based on the distortion contributed by their loss to the received video quality. The slices of a priority class in each frame are aggregated into corresponding video packets whose size is bounded by the network MTU. The proposed scheme provides higher transmission reliability to the high priority packets by using smaller fragments, at the expense of *(i)* allowing larger fragment sizes for the low priority packets, and *(ii)* discarding some low priority packets to meet the channel bit-rate limitations, whenever necessary. The Branch and Bound (BnB) algorithm along with an interval arithmetic method [56–58] is used to find the maximum expected weighted goodput and derive the optimal fragment sizes. Other features include a slice discard scheme based on the frame importance and the cumulative mean square error (CMSE) contribution of the slice, a slice fragmentation approach by modifying the conventional H.264 decoder to handle partial slice decoding, and use of various slice sizes. We show that adapting fragment sizes to the packet priority classes reduces the overall expected video distortion at the receiver. Our scheme does not assume retransmission of lost fragments and packets since real-time video transmission is delay-sensitive.

Section 2.2 formulates the expected weighted goodput maximization problem and presents the interval arithmetic analysis for determining the fragment sizes. The comparison between the performance of priority-aware and priority-agnostic fragmentation is discussed in Section 2.3. Experimental results for different combinations of video bit rates, slice sizes, and varying channel conditions are discussed in Section 2.4.

2.2 Proposed Cross-Layer Fragmentation Scheme

2.2.1 H.264 Slice and Video Packet Formation

We consider videos which are pre-encoded using H.264/AVC with fixed slice size configuration. In this configuration, macroblocks (MBs) are aggregated into a slice such that their accumulated size does not exceed the pre-defined slice size. However, the chosen slice size represents the upper limit and some slices may be smaller [59].

The network limits the number of bytes that can be transmitted in a single packet based on the MTU bound. The slices formed at the encoder are aggregated into a video packet for transport over IP networks and each of these packets is appended with RTP/UDP/IP headers of 40 bytes [60] as shown in Figure 2.1. This aggregation of slices helps to control the amount of network overhead added to the video data. If the video slices are classified in two or more priority classes as explained in Section 2.2.2, the priority slices of each frame are separately aggregated to form packets. The video packets are fragmented at the data link layer and each fragment is attached with 50 byte MAC and PHY layer headers. Figure 2.1 illustrates the cross-layer fragmentation approach.

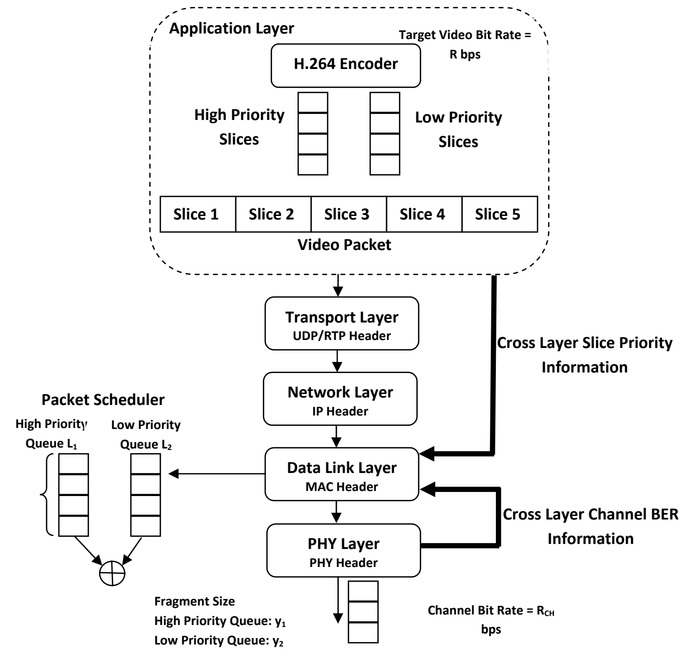


Figure 2.1: Cross-layer prioritized packet fragmentation approach for RTP, UDP, IP based transmission.

We use a binary symmetric channel $BSC(p_b)$ where p_b is the BER. The data link layer fragments the packets using channel BER information from the PHY layer and slice priority information from the APP layer. Here we assume that the data link layer is continuously updated with the channel BER from the PHY layer.

2.2.2 Slice Priority

H.264 slices are prioritized based on their distortion contribution to the received video quality. The loss of a slice introduces error in the current reference frame and could propagate to other frames in the GOP. We compute the total distortion by using the CMSE introduced by a slice loss, since it takes into consideration the error propagation within the entire GOP. Suppose the video resolution is $H \times W$, represented in terms of the number of pixels along the height (H) and width (W) of a video frame. Let $\widehat{Pel}_{i,j,k}$ represent the pixel intensity value at location (j, k) in the reconstructed frame i at the encoder without the slice loss and $\widetilde{Pel}_{i,j,k}$ represent the corresponding pixel intensity value in the same frame decoded at the receiver with the slice loss. The CMSE contributed by the loss of the slice is computed in Equation 2.1 as the sum of MSE over the current and all the other frames in the GOP.

$$CMSE = \sum_{i=current\ frame\ with\ slice\ loss-t}^{last\ frame\ of\ GOP} \left\{ \frac{1}{H \times W} \sum_{j=1}^{H} \sum_{k=1}^{W} \left(\widehat{Pel}_{i,j,k} - \widetilde{Pel}_{i,j,k} \right)^2 \right\} \quad (2.1)$$

Here, t is the temporal duration of a reference frame in the backward direction. The bi-directionally predicted (B) frames in the backward temporal direction are also covered by Equation 2.1.

All slices in a GOP are equally distributed into four priority classes based on their pre-computed CMSE values. Priority 1 slices induce the highest distortion whereas priority 4 slices induce the least distortion to the received video quality. The slice priority value is stored in the 2-bit nal_ref_idc field of the slice header [61]. We combine priority 1 and priority 2 slices into a 'high priority' class and priority 3 and priority 4 slices into a 'low priority' class to reduce the complexity of the optimization algorithm discussed in Section 2.2.5. However, the original four-level priority information can still be accessed from the nal_ref_idc field of each slice and will be used in our proposed slice discard scheme.

2.2.3 Video Packet Fragmentation

Optimal fragment size is determined to maximize the expected weighted goodput which will be explained in Section 2.2.4. We design two types of packet fragmentation schemes - video packet priority-agnostic and priority-aware. Each of these schemes has two types, *slice fragmentation disabled* and *slice fragmentation enabled*. When slice fragmentation is disabled, the fragment size cannot be smaller than the target slice size and each fragment contains one or more slices in their entirety. This restriction on the fragment size is not needed when slice fragmentation is enabled. As a result, a fragment can contain partial slice data, including the cases of less than one slice and more than one slice. In Figure 2.2, we illustrate an example where the computed optimal fragment size is larger than the target slice size but smaller than twice the target slice size. In the slice fragmentation disabled case, Fragment 1 contains only slice 1 since slice 2 is too large to fit entirely in it. Similarly, Fragment 2 and

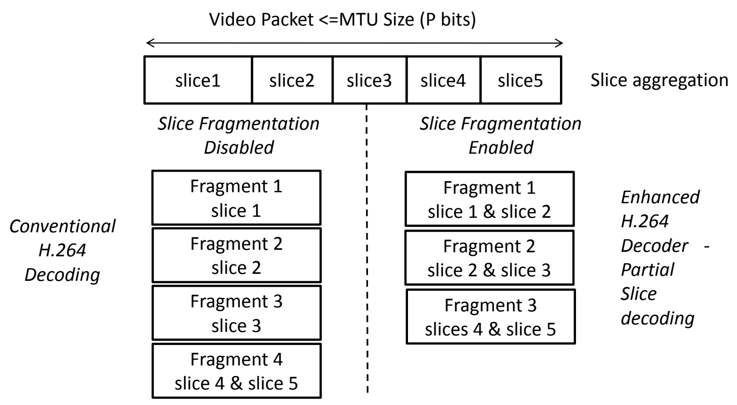

Video Packet <=MTU Size (P bits)
slice1
slice2
slice3
slice4
slice5
Slice aggregation
Slice Fragmentation Disabled
Slice Fragmentation Enabled
Conventional H.264 Decoding
Fragment 1 slice 1
Fragment 2 slice 2
Fragment 3 slice 3
Fragment 4 slice 4 & slice 5
Fragment 1 slice 1 & slice 2
Fragment 2 slice 2 & slice 3
Fragment 3 slices 4 & slice 5
Enhanced H.264 Decoder - Partial Slice decoding

Figure 2.2: Illustration of video packet fragmentation with and without slice fragmentation.

Fragment 3 contain only slice 2 and slice 3, respectively, since the next slice cannot fit in them. Slices 4 and 5 are small enough to be aggregated in Fragment 4. For the slice fragmentation enabled case, Fragment 1 contains slice 1 and the initial portion of slice 2, Fragment 2 contains the remaining portion of slice 2 and the complete slice 3, and Fragment 3 contains slices 4 and 5.

To enable slice fragmentation, the H.264 decoder is modified to perform partial slice decoding [62, 63]. If the first fragment of a slice containing the slice header is lost, the entire slice is discarded. When an intermediate fragment is in error, the MB data contained in fragments before the corrupted fragment is successfully decoded, and the remaining MB data is concealed at the decoder. We use the concealment implemented in JM 14.2 [59].

2.2.4 Problem Formulation for Determining Optimal Fragment Sizes

In conventional packet fragmentation schemes, the entire packet is discarded if any one of its fragments is not received properly. In our case, the decoder reconstructs the lost packets or fragments using error concealment. Video traffic can also tolerate some low priority slices being discarded to accommodate a higher fragmentation overhead when the overall video bit rate exceeds the channel bit rate. In this section, we discuss the priority-agnostic and priority-aware fragmentation schemes. Here the priority-agnostic fragmentation scheme is based on [11, 32, 33].

Priority-agnostic fragmentation

A measure of the reliable transmission of packets over error-prone channels is *goodput*. We define the goodput G as the expected number of successfully received video bits per second (bps) normalized by the target video bit rate R bps. G depends on the fragment success rate (f_{sr}) which is a function of the fragment size (y) and the channel BER (p_b). We assume that each slice is x bits long in our theoretical formulation. A fragment is successfully received iff all the bits of that fragment are received without error. The f_{sr} is expressed as

$$f_{sr} = (1 - p_b)^y, y = nx + h \tag{2.2}$$

Here, the fragment size is y bits, containing nx bits of slice data (i.e., payload) and h MAC and PHY header bits. For a given value of y, F_{TX} is the corresponding number of fragments transmitted every second and F_{RX} is the corresponding expected number of successfully received fragments. F_{RX} is computed as $F_{RX} = (fsr)(F_{TX})$. We assume that the channel bit rate is R_{CH} bps, the average video bit rate is R bps, and on average $N = R/x$ slices are generated every second. The number of payload bits in a fragment can vary from 1 to P bits, where P represents the MTU size. Therefore, the feasible number of slices in each fragment varies as $n \ \epsilon \ [\frac{1}{x} \ \frac{P}{x}]$. If slice fragmentation is disabled, n is an integer with minimum value of 1. The expected goodput G is computed, after excluding the header bits associated with each fragment, as

$$G = \frac{F_{RX}(y - h)}{R} = \frac{F_{TX}(1 - p_b)^y (y - h)}{R} \tag{2.3}$$

Here, the objective is to find the optimal fragment size y such that G is maximum:

$$y = \operatorname*{argmax}_{y} G = \operatorname*{argmax}_{y} \frac{F_{TX}(1-p_b)^y (y-h)}{R} \tag{2.4}$$

$$F_{TX} = \begin{cases} (\frac{N}{n}); (\frac{N}{n}) \leq \frac{R_{CH}}{y} \\ \frac{R_{CH}}{y}; (\frac{N}{n}) > \frac{R_{CH}}{y} \end{cases}$$

Condition $\left(\frac{N}{n}\right) \leq \frac{R_{CH}}{y}$ in Equation 2.4 implies that sufficient bits are available to allocate headers to all the fragments generated in one second. The condition $\left(\frac{N}{n}\right) > \frac{R_{CH}}{y}$ implies that for a fragment size of y bits, the requirement for the number of overhead bits exceeds the channel bit rate. Therefore the corresponding number of application layer packets that would be discarded is $\left\lceil \frac{\left(\left(\frac{N}{n}\right)-\left(\frac{R_{CH}}{y}\right)\right)n}{\left(\frac{P}{x}\right)} \right\rceil$. The corresponding number of discarded slices (D_S) is

$$D_S = \left\lceil \left(N - \left(\frac{R_{CH}}{y} \right) n \right) \right\rceil \tag{2.5}$$

We use a quantized exhaustive search algorithm with a step size of 50 bytes for 30 possible fragment sizes between 50 and 1500 bytes to compute the optimal fragment size in Equation 2.4. Figure 2.3(a) shows the variation in expected goodput G for different fragment sizes and channel BERs for a video encoded at $R = 960$ Kbps with 150 byte slices. The channel bit rate R_{CH} is set to 1 Mbps for all the cases discussed in this chapter. The maximum video data in a fragment is limited by P = 1500 bytes. For a fragment of 1500 bytes, the maximum value of G is 55% for $p_b = 5 \times 10^{-5}$ which increases to 98% for a lower channel BER $p_b = 10^{-6}$, because the f_{sr} increases as the channel BER decreases. The expected goodput also depends on the number of slices discarded. Note that more slices are discarded as the fragment size decreases since the requirement for header bits increases. Therefore, for a fragment size of 150 bytes, though f_{sr} is higher than that for larger fragment sizes, the corresponding G is lower. We observe that the value of G for $p_b = 5 \times 10^{-5}$ is significantly lower than for lower values of p_b. The system achieves a higher value of G at this BER when the encoding bit rate is lower, as shown in Fig. 2.3(b) for the 720 Kbps video bit rate. There lies an optimal point in each case which trades off the losses due to channel errors with the packet discards. For example, the maximum value of G is achieved at fragment sizes of 300 and 750 bytes for $p_b = 5 \times 10^{-5}$ and 10^{-5}, respectively.

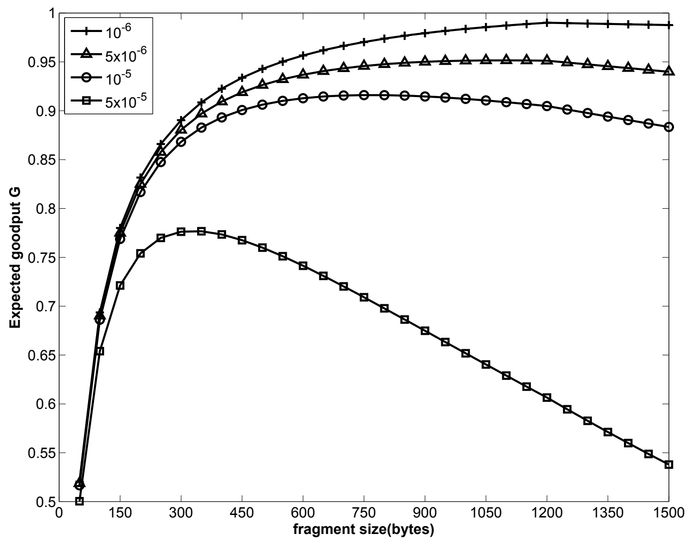
Figure 2.3(b) illustrates the variation in G for different fragment sizes and three different encoded video bit rates at $p_b = 5 \times 10^{-5}$. For R= 720 Kbps, sufficient bits are available to allocate headers to each fragment. So every slice of the video packet can be transmitted independently in a fragment with maximum $G = 93\%$. However, the maximum achievable G decreases as the encoded video bit rate increases and gets close to R_{CH} (i.e. R = 960 Kbps) or exceeds R_{CH} (i.e. R = 1.08 Mbps). This is because fewer bits are now available for allocating fragment headers. More header bits can only be accommodated by discarding some slices. As a result, the maximum value of G decreases to 77% and 69% for video bit rates of 960 Kbps and 1080 Kbps, respectively, when each fragment contains two slices.

Figure 2.4 shows the amount of discarded data for different video encoding rates at $p_b = 5 \times 10^{-5}$. As the video encoding rate increases, more slices are generated every second. When the encoding rate is 720 Kbps, sufficient bits are available to allocate fragment headers and hence no slice is discarded when fragment size $\geq$ 150 bytes. When R increases to 960 Kbps, the amount of discarded data increases. When the encoding rate (1080 Kbps) exceeds R_{CH}, 14.1 Kbytes worth of slice data is discarded every second even for a 1500 byte fragment size (i.e., no fragmentation). Though one may be inclined to choose a large fragment size to reduce the number of discarded slices, it also decreases the fragment success rate as explained in Figure 2.3 and shown in Equation 2.2.

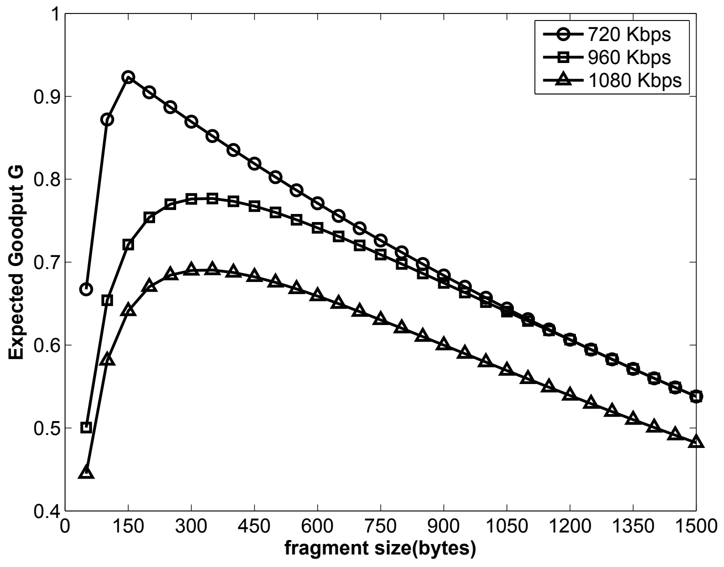
Priority-aware fragmentation

We extend the fragmentation scheme to make it adaptive to individual packet priority classes. We assign smaller fragment sizes to higher priority packets to increase their transmission success probability. The video packets are divided in two priorities and the link layer scheduler (shown in Figure 2.1) transmits all the high priority fragments before low priority fragments during every second. We define a new performance parameter called the expected weighted goodput G_W, which is computed as a linear combination of individual priority goodput :

$$G_W = w_1 g_1 + w_2 g_2 \tag{2.6}$$



(a)



(b)

Figure 2.3: Expected goodput G vs. fragment size y at (a) R =960 Kbps and different p_b, and (b) $p_b = 5 \times 10^{-5}$ and different R.

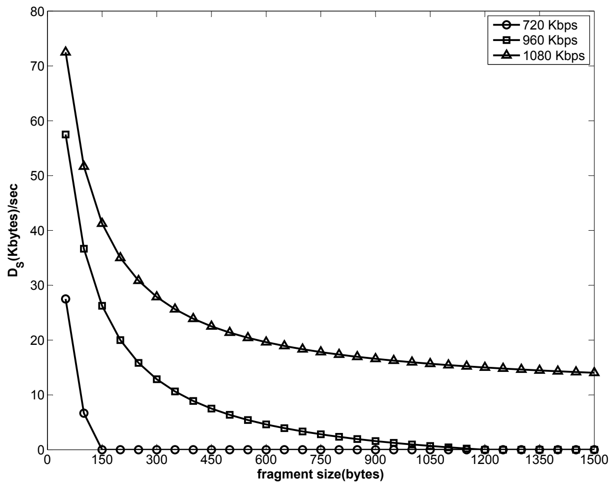


Figure 2.4: Slice data in Kbytes discarded per second at channel BER of 5×10^{-5}.

The weights w_1 and w_2 capture the relative distortion contribution per bit from the individual slice priorities. w_1 is computed as the ratio of the mean CMSE of the high priority slices to the mean CMSE of all slices in the pre-encoded video, and $w_2 = 1 - w_1$. We used the median of all slice CMSE values as the CMSE threshold for assigning slice priority. The weights depend on this threshold, video content, and encoding parameters such as target encoding rate R and slice size x. We define $n_1, n_2 \, \epsilon \, [\frac{1}{x} \; \frac{P}{x}]$ as the number of slices that are aggregated into each fragment of the high priority and low priority packets. The corresponding fragment sizes would be $y_1 = n_1 x + h, y_2 = n_2 x + h$ bits. Let N be the total number of slices generated in one second, and l_1 and l_2 be the corresponding numbers of high priority and low priority slices generated per second. During each second it is difficult to predict the number of packets in each priority queue at the data link layer. If the video has high motion activity for some period of time, it would have more slices with CMSE values greater than the threshold. As a result, there will be more high priority packets. In any given second, the number of high priority slices can vary from $[0 \; N]$ and the expected number is $\frac{N}{2}$. Hence, a truncated normal distribution, which is symmetric about $\frac{N}{2}$ and spanning from 0 to N is considered here:

$$p(l_1 = k) = \frac{K_1}{\sqrt{2\pi}} e^{-\frac{(k - \frac{N}{2})^2}{2}} \text{ for } k = 0, 1, 2, 3, ..., N \text{ and } l_1 + l_2 = N \qquad (2.7)$$

where K_1 is a normalization constant to make this a proper probability mass function.

Now, we find $\overline{y} = [y_1\ y_2]$ which maximizes G_W averaged over all possible queue lengths from $[0\ N]$,

$$\max_{\overline{y}} G_W = \max_{\overline{y}} \sum_{l_1} p(l_1)(w_1 g_1 + w_2 g_2) \tag{2.8}$$

Here, $p(l_1)$ is the probability of a given high priority queue length l_1. The individual priority goodputs g_1 and g_2 for a given high priority queue length l_1 and corresponding low priority queue length l_2 are therefore computed using the expected goodput formula expressed in Equation 2.4.

$$g_1 = \begin{cases} (1-p_b)^{y_1}; & \left(\frac{l_1}{n_1}\right) \leq \frac{R_{CH}}{y_1} \quad (a) \\ \frac{\left(\frac{R_{CH}}{y_1}\right)(y_1-h)(1-p_b)^{y_1}}{R}; & \left(\frac{l_1}{n_1}\right) > \frac{R_{CH}}{y_1} \quad (b) \end{cases} \tag{2.9}$$

$$g_2 = \begin{cases} (1-p_b)^{y_2}; \quad \left(\frac{l_2}{n_2}\right) \leq \frac{\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)}{y_2} & (a) \\ \frac{\left[\frac{\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)}{y_2}\right](y_2-h)(1-p_b)^{y_2}}{l_2 x}; & \\ \qquad \left(\frac{l_2}{n_2}\right) > \frac{\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)}{y_2} & (b) \end{cases} \tag{2.10}$$

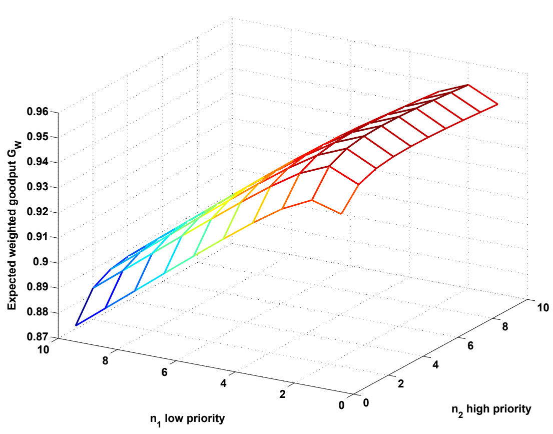
The low priority goodput g_2 is computed from the bits remaining to be allocated after all the high priority fragments have been transmitted during each second. Condition (a) in Equations 2.9 and 2.10 implies that sufficient bits are available to allocate fragment headers when high and low priority fragments are transmitted at sizes y_1 and y_2. Condition (b) in Equation 2.9 implies that all the low and some high priority slices should be discarded to accommodate the overhead demand and satisfy the channel bit rate constraint while transmitting at a fragment size y_1. The slice discard scheme is discussed later in Section 2.4.3. Further, Condition (b) in Equation 2.10 implies that there are sufficient bits to transmit all high priority fragments at size y_1, but not for transmitting all low priority fragments at size y_2. Therefore, some low priority slices should be discarded. Combining Equations 2.8, 2.9 and 2.10 and substituting $l_2 = N - l_1$, we formulate the objective function to maximize G_W as:

$$\max_{\overline{y}} G_W = \max_{\overline{y}} \sum_{l_1} p(l_1) \begin{cases} \frac{w_1 n_1 x}{y_1 R} R_{CH}(1-p_b)^{y_1} = f_1(n_1) \; ; \; C_1 \; : \; \left(\frac{l_1}{n_1}\right) \geq \frac{R_{CH}}{y_1} \\ w_1(1-p_b)^{y_1} + \frac{\frac{w_2 n_2}{y_2}\left(R_{CH} - \left(\frac{l_1}{n_1}\right) y_1\right)(1-p_b)^{y_2}}{N-l_1} = f_2(n_1, n_2) \; ; \\ \qquad C_2 \; : \; \left(\frac{l_1}{n_1}\right) y_1 + \left(\frac{N-l_1}{n_2}\right) y_2 > R_{CH}, \; \left(\frac{l_1}{n_1}\right) < \frac{R_{CH}}{y_1} \\ w_1(1-p_b)^{y_1} + w_2(1-p_b)^{y_2} = f_3(n_1, n_2) \; ; \\ \qquad C_3 \; : \; \left(\frac{l_1}{n_1}\right) y_1 + \left(\frac{N-l_1}{n_2}\right) y_2 \leq R_{CH} \end{cases} \tag{2.11}$$

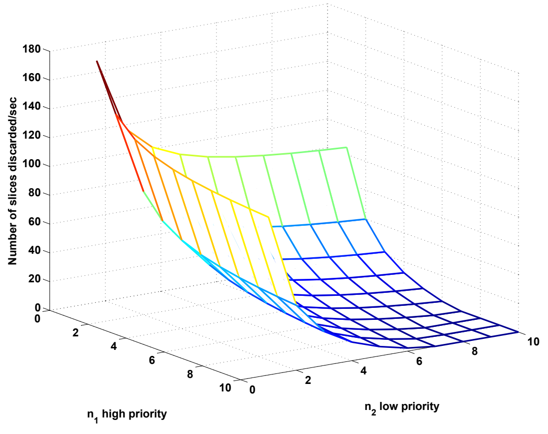
Figure 2.5 shows G_W and the number of discarded slices during one second for the CIF Foreman video encoded at $R = 960$ Kbps over a channel with $R_{CH} =$ 1 Mbps at $p_b = 10^{-5}$. The weights $(w_1, w_2) = (0.89, 0.11)$ used were derived for the Foreman video sequence. The mean CMSE value of high priority slices contributes 89% of the received video distortion whereas the mean CMSE value of low priority slices contributes only 11%. The optimal fragment sizes are determined in terms of the number of 150 byte slices that can be aggregated into each priority fragment. In Figure 2.5(a), $(n_1, n_2) = (2, 5)$ and $[(y_1, y_2) = (300, 750) + h]$ are the optimal high and low priority fragment sizes which achieve the maximum G_W of 0.954. This is achieved at the cost of discarding 56 low priority slices per second as shown in Figure 2.5(b). As the fragment size decreases, the fragment success rate increases but the number of discarded slices also increases due to higher fragment overhead. When $(n_1, n_2) = (1, 1)$, more than 175 slices are discarded as shown in Figure 2.5(b) and the corresponding G_W decreases to 0.92 in Figure 2.5(a). Also when $(n_1, n_2) = (10, 1)$, i.e. low priority packets are transmitted at smaller fragment sizes and high priority packets are transmitted at larger fragment size, the corresponding G_W reaches its minimum value of 0.87 and 85 low priority slices are discarded.

2.2.5 Branch and Bound (BnB) Optimization using Interval Arithmetic Analysis

We used the BnB technique along with interval arithmetic analysis to solve the priority-aware expected weighted goodput optimization problem [56]. BnB is a global optimization technique used for non-convex problems, especially in discrete



(a)



(b)

Figure 2.5: (a) Expected weighted goodput, and (b) slices discarded for Foreman encoded at R=960 Kbps and x=150 bytes.

and combinatorial optimization. The original domain of the optimization variables is divided into smaller sub-regions, and interval arithmetic analysis is performed in each sub-region to compute the lower and upper bounds. The interval arithmetic analysis uses inclusion functions $f_1(n_1)$, $f_2(n_1, n_2)$, and $f_3(n_1, n_2)$ derived from our main objective function in Equation 2.11 to compute the bounds. Depending on the computed bounds, a decision is made on whether a sub-region is retained or pruned [56–58]. In Equation 2.11, the number of slices in the high and low priority fragments (i.e., n_1, n_2) and the conditions C_1, C_2, and C_3 define the search region.

Each sub-region is defined by the lower and upper bounds of n_1 and n_2 as $[\underline{n}_1 \quad \overline{n}_1]$ and $[\underline{n}_2 \quad \overline{n}_2]$ and these are used to compute the bounds of G_W in that sub-region using the above inclusion functions. The lower and upper bounds of function $f_1(n_1)$ satisfying condition C_1 are derived as

$$\begin{aligned} \underline{f}_1 &= \left(\frac{w_1 \underline{n}_1 x}{(\underline{n}_1 x + h)R} R_{CH} (1-p_b)^{\overline{n}_1 x + h} \right) \\ \overline{f}_1 &= \left(\frac{w_1 \overline{n}_1 x}{(\overline{n}_1 x + h)R} R_{CH} (1-p_b)^{\underline{n}_1 x + h} \right) \end{aligned}$$

This is because $f_1(n_1)$ in Equation 2.11 can be expressed as a product of two functions, $A(n_1) = \frac{(w_1 n_1 x)}{(n_1 x + h)R} R_{CH}$ and $B(n_1) = (1-p_b)^{(n_1 x + h)}$. Here, $A(n_1)$ is minimum at $\underline{n}_1$ and $B(n_1)$ is minimum at $\overline{n}_1$, and both are positive. Therefore, the lower bound

Table 2.1: Optimal number of slices in high and low priority fragments after running BnB algorithm.

BER	720 Kbps	960 Kbps	1080 Kbps
10^{-6}	(1,1)	(5,10)	(6,10)
5×10^{-6}	(1,1)	(2,7)	(3,7)
10^{-5}	(1,1)	(2,5)	(2,5)
5×10^{-5}	(1,1)	(1,2)	(1,2)
10^{-4}	(1,1)	(1,2)	(1,2)

$\underline{f}_1 = A(n_1)|_{\underline{n}_1} \times B(n_1)|_{\overline{n}_1}$. Similarly, $A(n_1)$ is maximum at $\overline{n}_1$ and $B(n_1)$ is maximum at $\underline{n}_1$, and both are positive. Therefore, the upper bound $\overline{f}_1 = A(n_1)|_{\overline{n}_1} \times B(n_1)|_{\underline{n}_1}$. Similarly, the lower and upper bounds of functions $f_2(n_1, n_2)$ and $f_3(n_1, n_2)$ are also derived. The lower (LB) and upper (UB) bounds of G_W in a sub-region are computed as the expected lower and upper bounds of the inclusion functions in that sub-region over queue length l_1 varying over $[0\ N]$.

In the BnB algorithm, the overall search region of the variables n_1 and n_2 and the sub-regions generated from them form a tree. The sub-regions contain the estimated LB and UB of G_W. Depending on the values of these bounds, a decision is made to either retain or prune a sub-region. The following steps are applied iteratively until the maximum G_W value is obtained:

1) In the set of unexpanded sub-regions, find the maximum 'LB' say LB_{max} and prune all those sub-regions whose UB is less than LB_{max}.

2) From the set of unpruned and unexpanded sub-regions, select the sub-region with maximum 'UB' and further spawn it to form two more sub-regions.

3) Modify the set of unexpanded sub-regions and repeat step 1.

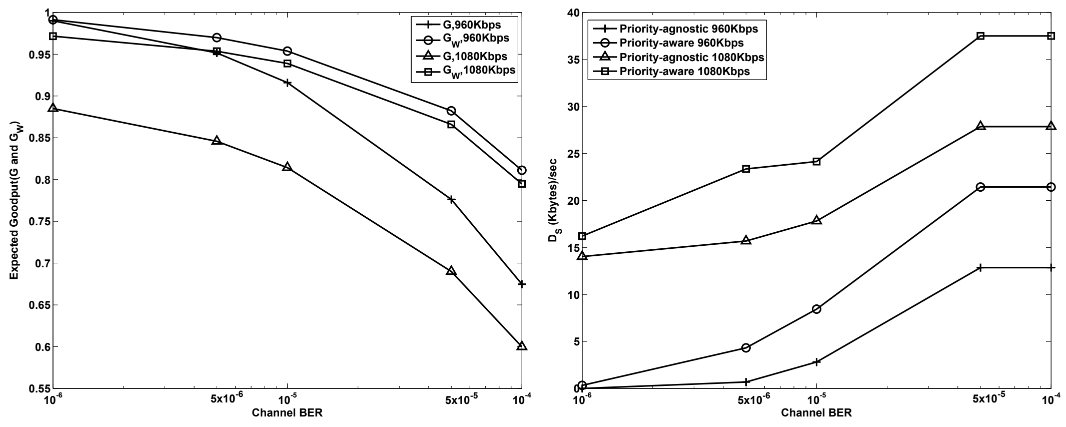
The BnB algorithm reduces the number of times (i.e., 36 times for R = 960 Kbps, $p_b = 5 \times 10^{-5}$, and 150 byte slice size) the expected weighted goodput values have to be computed as compared to the exhaustive search case (i.e., 100 times for all combinations of $n_1 \in [1\ 10]$ and $n_2 \in [1\ 10]$) .

Table 2.1 shows the optimal (n_1, n_2) values derived for different encoding rates and channel BERs. (n_1, n_2) remains the same at R=720 Kbps for different channel BERs since sufficient bits are available to allocate headers to each fragment. At R=960 Kbps and 1080 Kbps, the (n_1, n_2) decreases as the channel BER increases in order to increase the fragment success rate. Also as the encoding rate increases towards the channel transmission rate R_{CH}=1 Mbps for a given BER, (n_1, n_2) increases in order to limit the increase in the number of slices discarded. The computation time for the BnB algorithm is directly dependent on the number of slices generated during every second. The exhaustive search takes 76 ms to determine the optimal point whereas the BnB algorithm takes 56 ms on a Core 2 Duo 2.6 GHz Intel processor with 4 GB RAM. This value is obtained when the distribution of slices in the priority queues (l_1, l_2) is assumed unknown and we need to compute the expected G_W. However, the encoder would compute the distribution of slices in the priority queues during each second. When this information is available, the BnB algorithm takes only 13 ms to compute the optimal point whereas the exhaustive search takes 28 ms.

2.3 Priority-agnostic vs. Priority-aware Fragmentation

In this section, we compare the goodput and slice discard rates of the priority-agnostic and priority-aware schemes analytically, for two bit rates of 960 Kbps and 1080 Kbps over a 1024 Kbps channel. In the next section, we compare the performance of priority-agnostic and priority-aware schemes using simulations with video sequences. The maximum expected goodput G in priority-agnostic fragmentation and G_W in priority-aware fragmentation shown in Figure 2.6(a) are computed using Equations 2.3 and 2.11, respectively. The slice size is 150 bytes and the weights w_1 and w_2 for the priority-aware fragmentation are derived for the Foreman video sequence as discussed in Section 2.2.4. For 960 Kbps, the weights $(w_1,\ w_2) = (0.89,\ 0.11)$ and for 1080 Kbps $(w_1,\ w_2) = (0.9,\ 0.1)$. The corresponding optimal fragment sizes for priority-aware fragmentation were listed in Table 2.1. The number of Kbytes/sec discarded in Figure 2.6(b) in order to achieve maximum goodput is computed using Equation 2.5.

As shown in Figure 2.6(a), priority-aware fragmentation achieves a goodput gain of 14% over priority-agnostic fragmentation at $R = 960$ Kbps and $p_b = 10^{-4}$, even when it discards 8.6 Kbytes of additional data per second as shown in Figure 2.6(b). However, the performance of both fragmentation schemes starts converging as the

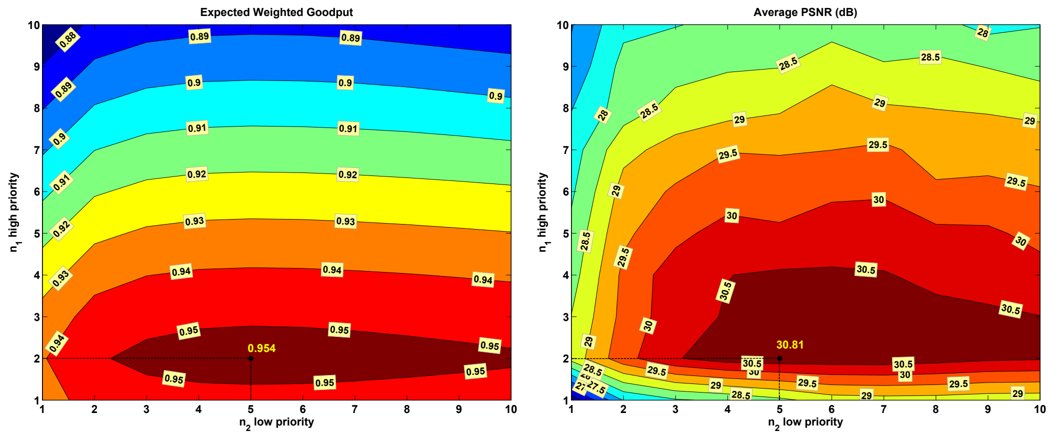


(a) (b)

Figure 2.6: (a) Expected goodput, and (b) slice discard comparisons between priority-agnostic and priority-aware fragmentation.

channel BER decreases from 10^{-4} to 10^{-6}. This is expected because, when the channel is good enough, slices are not discarded (see Figure 2.6(b)) and packets are not lost, and so fragmentation does not need to use priority information. In fact, the priority-agnostic case uses 8 slices per fragment at BER of 10^{-6} with no slice data discarded whereas the priority-aware case uses 5 slices per fragment for high priority and 10 slices per fragment for low priority discarding only 320 bytes of low priority slice data. Similarly, Figure 2.6(a) shows that the priority-aware fragmentation achieves a goodput gain of 20% over priority-agnostic fragmentation at R=1080 Kbps and $p_b = 10^{-4}$. We discard 9.6 Kbytes of additional data to achieve this gain as shown in Figure 2.6(b). Unlike R=960 Kbps, the priority-aware fragmentation achieves a goodput gain of 9% over priority-agnostic fragmentation at lower BER ($p_b = 10^{-6}$) for R=1080 Kbps in Figure 2.6(a). Note that (*i*) priority-aware fragmentation uses 6 and 10 slices per fragment for the high and low priority packets, respectively, as compared to 10 slices per fragment in the priority-agnostic case, and (*ii*) discards slightly more low priority slice data (i.e., 16.2 Kbytes) as compared to discarding 14 Kbytes in priority-agnostic fragmentation as shown in Figure 2.6(b).

Though the priority-aware fragmentation provides goodput gain by increasing the transmission reliability of higher priority packets, we have also investigated if this G_W gain corresponds to better video quality. We illustrate the results for Foreman encoded at 960 Kbps with a slice size of 150 bytes and transmitted over a 1 Mbps



(a) (b)

Figure 2.7: (a) Expected weighted goodput G_W, and (b) average PSNR at R= 960 Kbps, R_{CH}= 1 Mbps, and $p_b = 10^{-5}$.

channel with BER $= 10^{-5}$. The expected goodput values for $n_1 \epsilon [1\ 10]$, $n_2 \epsilon [1\ 10]$ were shown in Figure 2.5(a). We have also computed the video quality (in terms of average PSNR (dB)) for these values of n_1 and n_2. Figures 2.7(a) and 2.7(b) illustrate the contour plots displaying the isolines (line connecting the points of equal value) of G_W and video PSNR. The distance between the isolines is equivalent to the gradient which represents the improvement in the corresponding values of G_W and video PSNR. The plots show that higher G_W generally corresponds to higher video PSNR values. For example, the dark red region in the contour plots represents the highest value of G_W = 0.954 for $(n_1, n_2) = (2, 5)$ which also corresponds to the highest PSNR of 30.81 dB. Similarly, the dark blue region in the contour plots represents the lowest value of G_W = 0.87 for $(n_1, n_2) = (10, 1)$ which also corresponds to the lowest PSNR of 25.65 dB. We have observed a similar behavior for other video sequences (e.g., CIF Silent video) and encoding rates.

2.4 Experimental Results and Discussion

2.4.1 Simulation Setup

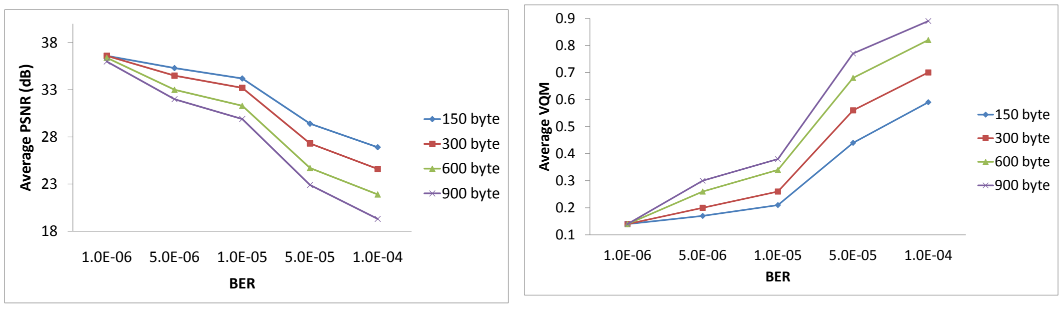
This section evaluates the performance of the baseline system, and priority-agnostic and priority-aware fragmentation. The baseline system does not include slice prioritization and the packets are transmitted at the network limited MTU size of 1500 bytes. Two CIF resolution (352 x 288) video sequences Foreman and Silent are used in our experiments, where Silent has lower motion activity than Foreman. They are encoded using H.264/AVC JM 14.2 reference software [59] for a GOP length of 20 frames with GOP structure IDR B P B ... P B IDR at 30 frames/sec (fps) and encoding rates of 720 Kbps, 960 Kbps and 1080 Kbps. Slice sizes of 150, 300, 600 and 900 bytes are used and the slices are formed using dispersed mode FMO with two slice groups. Two reference frames are used for predicting the P and B frames, with error concealment enabled using temporal concealment and spatial interpolation. The error concealment in a frame depends on the frame type and the type of losses encountered. If an entire frame (IDR, P or B) is lost, first the motion vectors and reference indices of the co-located MBs in the previously decoded reference frame are copied and then motion compensation is used to reconstruct the lost frame based on the copied motion information [64, 65]. If some slices of a predicted (P or B) frame are lost, the decoder verifies the availability of motion vector information for the lost MBs. If the motion vectors are available, motion copy is performed else co-located MBs of the previous reference frame are directly copied. If some slices of an IDR frame are lost, the corresponding MBs are concealed using spatial interpolation. Error concealment is enabled for all the schemes evaluated in this section. The channel transmission rate is 1 Mbps and the PHY and MAC layer header h is set to 50 bytes.

Three video encoding rates are chosen to study the following cases, *(i)* At 720 Kbps video bit rate, the 1 Mbps channel can support the fragment overhead; *(ii)* At 960 Kbps, the channel may not have sufficient bits to accommodate fragment overhead without slice discard; and *(iii)* at 1080 Kbps, the channel cannot even support the encoded video rate. For both test sequences, the PSNR increases with encoding rate as well as slice size. As the slice size increases, more MBs are encoded in each slice; this can more effectively exploit the spatial correlation in neighboring MBs. A 900 byte slice provides a 0.4 - 0.5 dB PSNR gain compared to a 150 byte slice size. Similarly, when the encoding rate is increased from 720 Kbps to 960 Kbps, the PSNR increases by 1 - 1.5 dB. In addition to computing PSNR, the subjective quality of the resultant videos is also evaluated using the perceptually based Video Quality Metric (VQM) discussed in [66, 67]. VQM is reported as a single number for the entire sequence and has a nominal output range from zero to one, where zero represents best quality.

2.4.2 Priority-Agnostic Fragmentation

Priority-agnostic packet fragmentation ignores the packet priorities and uses the optimal fragment size derived by maximizing the expected goodput G as discussed in Section 2.2.4. The average video PSNR and average VQM achieved by the priority-agnostic fragmentation for the Foreman sequence are shown in Figures 2.8, 2.9, and 2.10. The average PSNR in Figures 2.8, 2.9, and 2.10 decreases when the channel BER increases. For 720 Kbps video, the video quality is purely determined by the impact of channel errors as sufficient bits are available to allocate a header to each fragment. At high channel BER, the fragment success probability (Equation 2.2) decreases, resulting in more errors. The deterioration in expected goodput G due to increasing channel BER was shown in Figure 2.3(a). For higher video bit rates 960 and 1080 Kbps and larger slice sizes, the fragment sizes may be higher and some slices are also discarded to meet the channel bit rate constraint which leads to more video quality deterioration. For example, a video bit rate of 1080 Kbps requires 27 Kbytes of slice data to be discarded every second as shown in Figure 2.4 in order to achieve a maximum expected goodput G of only 0.67 by transmitting 300 byte fragments as shown in Figure 2.3(b). The fragments formed from smaller slice sizes, though, provide better PSNR performance as compared to fragments formed from larger slice sizes in Figures 2.8(a), 2.9(a), and 2.10(a). This is because smaller slice size allows a finer aggregation of video data into fragments. For example, each fragment contains eight and two 150 byte slices at BER of 10^{-6} and 10^{-4}, respectively, as compared to only one 900 byte slice.

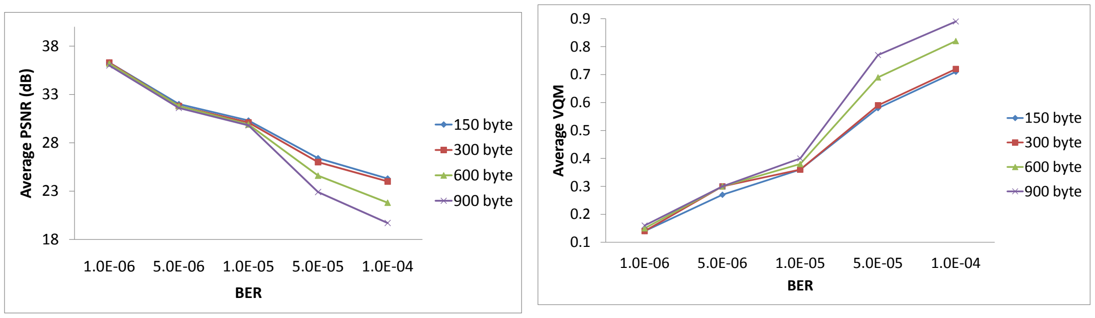
The VQM plots illustrated in Figures 2.8(b), 2.9(b), and 2.10(b) agree with the trends observed in average PSNR values. A high level of perceived impairment can be observed at high channel BERs, high video bit rates, and large slice sizes. Figure 2.11 compares the expected goodput, video PSNR and VQM gains achieved by fragments formed from 150 byte slices over those formed from 900 byte slices for Foreman. The gains generally increase with BER for each video bit rate. At 720



(a) (b)

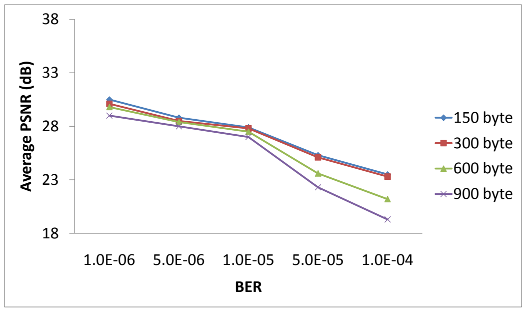
Figure 2.8: Average PSNR and VQM achieved by priority-agnostic fragmentation for Foreman encoded at 720 Kbps.

Kbps and BER of 10^{-4}, a large goodput gain of more than 45%, a PSNR gain of 7.6 dB, and VQM gain of 0.3 is achieved. Similarly, 31% gain in goodput is achieved for video bit rates 960 and 1080 Kbps with corresponding PSNR gains of 4.6 dB and 4.2 dB, and VQM gains of 0.18 and 0.17, respectively.

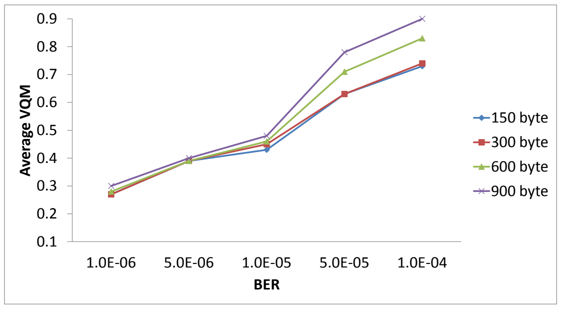


(a) (b)

Figure 2.9: Average PSNR and VQM achieved by priority-agnostic fragmentation for Foreman encoded at 960 Kbps.



(a)



(b)

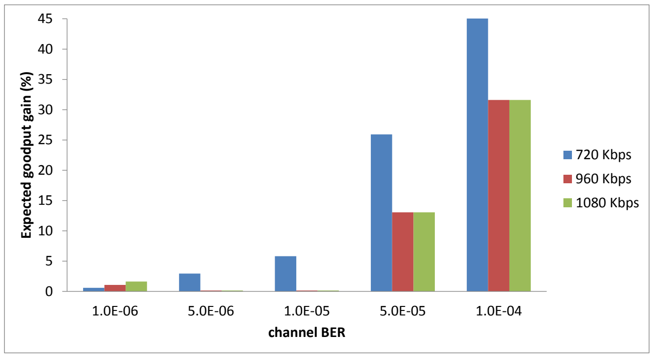
Figure 2.10: Average PSNR and VQM achieved by priority-agnostic fragmentation for Foreman encoded at 1080 Kbps.

The PSNR and VQM gain achieved by the priority-agnostic fragmentation over the baseline system is shown in Figure 2.12. Considerable gains are achieved for 150, 300, and 600 byte slice sizes due to enhanced fragment success rate in priority-agnostic fragmentation as compared to the baseline system. The gain for these slice sizes generally increases with BER for 720 and 960 Kbps video bit rate. Also smaller slice sizes achieve larger PSNR gains as they can achieve finer aggregation of video data in fragments. For example, a PSNR gain of 10.4 dB is achieved for a 720 Kbps video encoded using 150 byte slices at a channel BER of 10^{-4}. For 900 byte slices, the gain is only up to 1 dB. At a low channel BER of 10^{-6}, different slice sizes achieve less than 2 dB gain, since very few fragments are corrupted by error. Figures 2.12(c) and 2.12(d) show that priority-agnostic fragmentation significantly improves the perceptual video quality as compared to the baseline transmission.

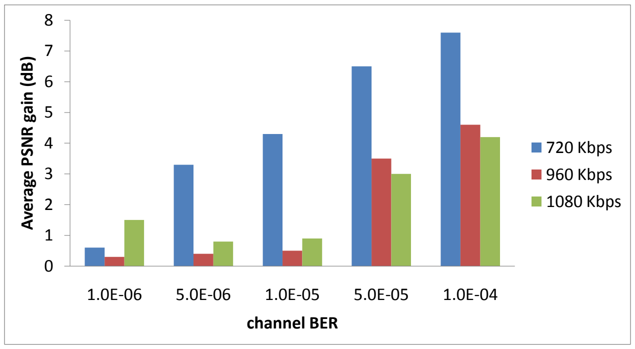
2.4.3 Priority-Aware Fragmentation

We adapt the fragment size to slice priorities as explained in Section 2.2.4, i.e., larger fragment size is used for the low priority slices, along with slice fragmentation. Priority-aware fragmentation thus maximizes the expected weighted goodput G_W by increasing the transmission reliability of high priority packets.

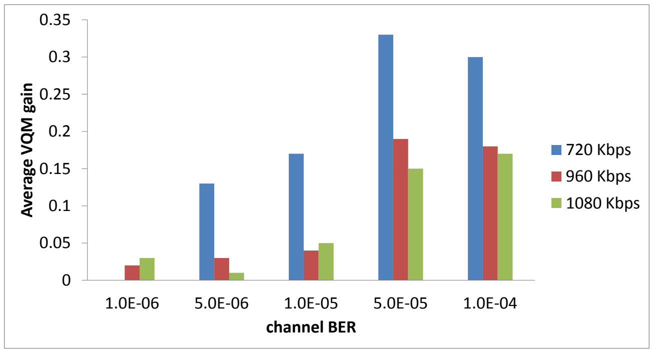
In order to discard the excess slices as discussed in Section 2.2.4, we first use a modified drop-tail based slice discard scheme which first discards the lowest priority (i.e., S=4) slices, followed by S=3 and S=2 slices, for each one second interval.



(a)

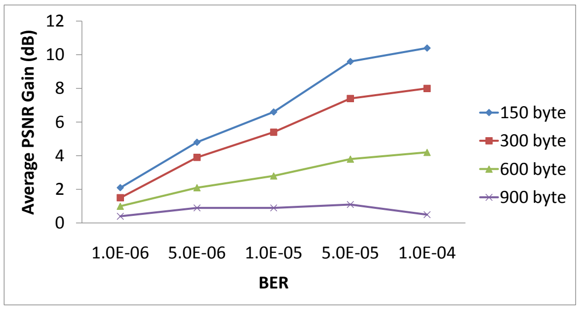


(b)

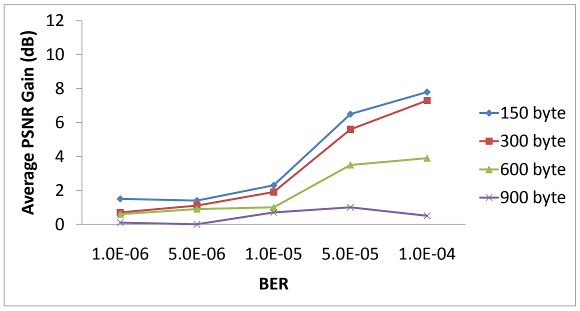


(c)

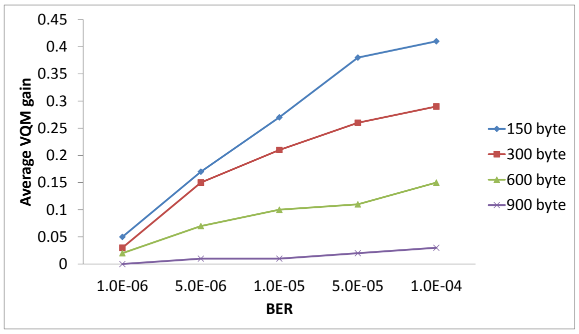
Figure 2.11: Gains achieved by 150 byte slices over 900 byte slices in priority-agnostic fragmentation for Foreman, in terms of (a) expected goodput, (b) average PSNR, and (c) average VQM.



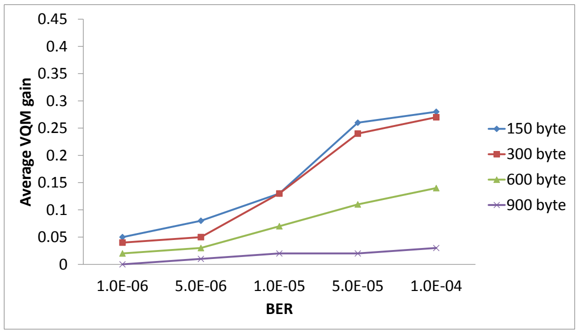
(a)



(b)



(c)



(d)

Figure 2.12: Average PSNR gain achieved by priority-agnostic fragmentation over baseline system for Foreman encoded at (a) 720 Kbps and (b) 960 Kbps, and corresponding average VQM gain at (c) 720 Kbps and (d) 960 Kbps.

Tables 2.2(a) and 2.2(b) show the PSNR and VQM gains for Foreman achieved by the priority-aware fragmentation over the priority-agnostic fragmentation when both schemes use the modified drop-tail mechanism for slice discard.

To further increase the PSNR values of the priority-aware fragmentation, we also propose a slice discard scheme for dispersed mode FMO with two slice groups as follows. The total slices discarded is given by Equation 2.5 and shown in Figure 2.6(b).

Step 1: Consider the B-frame starting from the end of the GOP and drop slices from Slice Group 1 (SG1) corresponding to the two lowest priorities (i.e., S=3 and 4).

Step 2: Repeat Step 1 for the next B-frame until slices from all the B-frames are discarded.

Step 3: Repeat Steps 1 and 2 for SG2 slices of B frames.

Step 4: Consider the P-frame starting from the end of GOP and discard the lowest priority (i.e., S=4) slices from SG1.

Step 5: Repeat Step 4 for the next P-frames.

Step 6: Drop the lowest priority (i.e., S=4) slices from SG1 of the IDR frame.

Step 7: Drop high priority slices (i.e., S=1 and 2) from SG1 and SG2 of B-frames starting from the end of the GOP.

Since dispersed mode FMO is used, if some low priority slices from a slice group are discarded, the lost MBs will be concealed from spatially adjacent MBs belonging to the other slice group. Since error concealment may not be effective when spatially adjacent MBs are discarded, we discard the lowest priority slices from only one slice group in P and IDR frames. Since B-frame slices do not cause error propagation and can be effectively concealed, our scheme allows the discard of slices from both B-frame slice groups. Unlike the proposed slice discard scheme, the modified drop-tail based scheme does not consider the slice group, frame type and frame location information. Table 2.3 shows the additional gain, both in terms of PSNR and VQM, achieved by the proposed slice discard scheme over the modified drop-tail scheme in the priority-aware fragmentation. We have achieved similar gains for the Silent video sequence. Note that no additional gain is achieved for 720 Kbps video as no slices are discarded for this bit rate.

Figure 2.13 shows the expected received video PSNR and VQM for priority-

Table 2.2: (a) PSNR and (b) VQM gains of priority-aware over priority-agnostic fragmentation with modified drop-tail slice discard for Foreman at 720, 960, and 1080 Kbps.

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.5, 0.3, 0.1	0.3, 0.9, 0.3	0.2, 1.1, 0.4	0.2, 0.4, 0.4	0.1, 1, 1.2
300 byte	0.5, 0.2, 0.5	1.1, 1.1, 0.4	1.2, 1.3, 0.3	1.7, 0.2, 0.4	2.1, 0.7, 1
600 byte	0.7, 0.2, 0.8	2.6, 1, 0.5	2.7, 0.9, 0.4	4.2, 1.3, 1.4	4.2, 2.7, 2.7
900 byte	1, 0.3, 1.4	3.1, 0.7, 0.3	4.2, 0.5, 0.3	5.1, 2.7, 2.5	5.9, 4.3, 4.2

(b)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.01, 0.01, 0.05	0.01, 0.03, 0.08	0.01, 0.08, 0.09	0.01, 0.08, 0.06	0.01, 0.09, 0.09
300 byte	0.01, 0.01, 0.05	0.03, 0.06, 0.08	0.05, 0.08, 0.1	0.1, 0.07, 0.05	0.1, 0.07, 0.07
600 byte	0.01, 0.01, 0.06	0.08, 0.06, 0.08	0.12, 0.06, 0.09	0.22, 0.11, 0.1	0.22, 0.16, 0.15
900 byte	0.02, 0.01, 0.07	0.12, 0.03, 0.07	0.2, 0.05, 0.09	0.29, 0.18, 0.16	0.25, 0.22, 0.2

Table 2.3: (a) PSNR and (b) VQM gain due to proposed slice discard for Foreman at 960 Kbps (1080 Kbps).

(a)

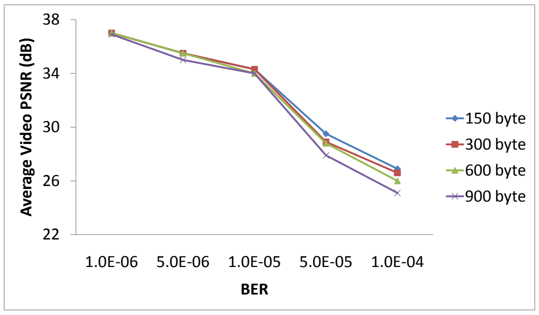
BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.1 (1.7)	0.4 (1.3)	0.3 (1.4)	0.8 (0.5)	0.5 (0.4)
300 byte	0.2 (1.1)	0.4 (1.0)	0.2 (0.8)	0.5 (0.3)	0.3 (0.3)
600 byte	0.2 (0.9)	0.3 (0.5)	0.2 (0.6)	0.2 (0.5)	0.2 (0.2)
900 byte	0.3 (0.3)	0.2 (0.6)	0.3 (0.4)	0.2 (0.3)	0.3 (0.3)

(b)

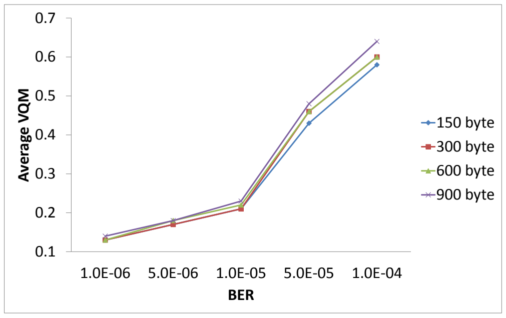
BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.01 (0.04)	0.01 (0.03)	0.01 (0.03)	0.02 (0.01)	0.02 (0.01)
300 byte	0.01 (0.03)	0.01 (0.03)	0.01 (0.02)	0.01 (0.01)	0.01 (0.01)
600 byte	0.01 (0.02)	0.01 (0.01)	0.01 (0.02)	0.01 (0.01)	0.01 (0.01)
900 byte	0.01 (0.01)	0.01 (0.02)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)

aware fragmentation with the proposed slice discard scheme for Foreman. The variation of average video PSNR with BER and slice size is similar to our observations for the priority-agnostic fragmentation shown in Figures 2.8, 2.9, and 2.10. In particular, we notice in Figures 2.13(a) and 2.13(c) that the use of slice fragmentation in priority-aware fragmentation has improved the PSNR performance of large slice sizes and narrowed the gap between large and smaller slices sizes as compared to the priority-agnostic fragmentation (in Figures 2.8, 2.9, and 2.10). It also has reduced the amount of visual impairment as measured by VQM compared to the priority-agnostic fragmentation (see Figures 2.13(b) and 2.13(d)). However, as discussed in Section 2.2.3, the loss of the first fragment containing the slice header causes the subsequent fragments of a packet to be discarded at the receiver even though they were successfully received. This still keeps the performance of large slices slightly inferior as compared to 150 byte slices when the channel error rate is high and the resulting fragment success rate is low. Slice fragmentation of a large slice also causes more slices to be discarded from the buffer when the video is encoded at high bit rates and results in adjacent slices of the frame to be dropped. This causes the large slices to underperform as compared to the 150 byte case even at low channel error rates at 1080 Kbps as shown in Figure 2.13(e).

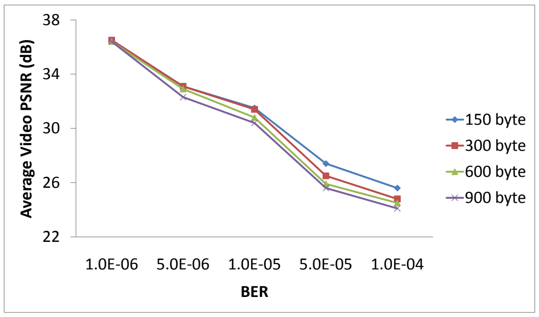
Tables 2.4 and 2.5 show the video PSNR and VQM gains achieved by priority-aware over priority-agnostic fragmentation without using slice fragmentation for Foreman and Silent at 960 Kbps. The corresponding gain for 1080 Kbps video is shown in brackets. The priority-aware fragmentation scheme adapts the fragment sizes to the individual packet priority levels which results in better received video quality. Note that both schemes have the same PSNR performance at a video bit rate of 720 Kbps because the margin of 304 Kbps for 1024 Kbps (i.e., 1 Mbps) channel bit rate is sufficient to transmit one slice per fragment. At a given channel BER, it is also observed that higher gain is achieved for smaller slice sizes. Increasing the slice size decreases the flexibility in choosing the fragment sizes as each fragment contains one or more slices in their entirety. For example, the fragment size can be either 600 bytes or 1200 bytes for a 600 byte slice size. Moreover a 900 byte slice allows us only 1 slice/fragment at 1500 bytes MTU. This restricts the PSNR gain that can be achieved by priority-aware over priority-agnostic fragmentation. A similar trend



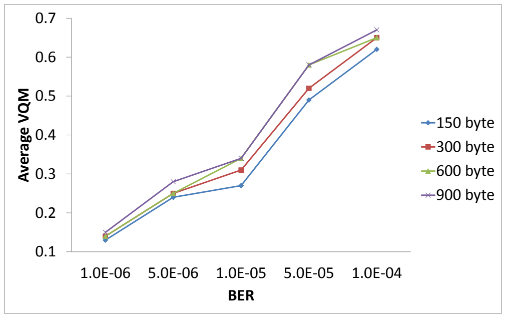
(a)



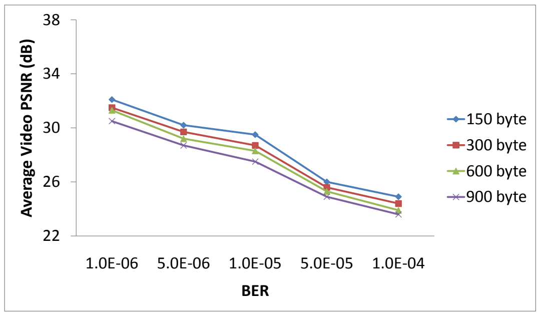
(b)



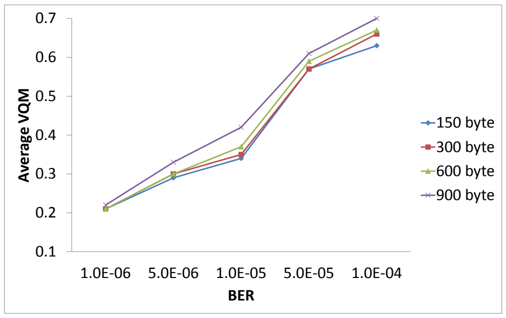
(c)



(d)



(e)



(f)

Figure 2.13: Average PSNR achieved by priority-aware fragmentation for Foreman encoded at (a) 720 Kbps, (c) 960 Kbps, and (e) 1080 Kbps, and corresponding average VQM at (b) 720 Kbps, (d) 960 Kbps, and (f) 1080 Kbps.

Table 2.4: (a) PSNR and (b) VQM gains of priority-aware over priority-agnostic fragmentation (without slice fragmentation) for Foreman at 960 Kbps (1080 Kbps).

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.3 (1.7)	1 (1.5)	1.3 (1.7)	1.2 (0.8)	1.4 (1.5)
300 byte	0.1 (1.5)	1 (1.2)	1 (0.8)	0.6 (0.6)	0.2 (0.6)
600 byte	0.1 (1.3)	0.8 (0.7)	0.9 (0.8)	0.2 (0.6)	0.2 (0.5)
900 byte	0.2 (0.7)	0.7 (0.6)	0.5 (0.6)	0.2 (0.5)	0.2 (0.7)

(b)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0 (0.07)	0.05 (0.09)	0.05 (0.1)	0.08 (0.06)	0.08 (0.09)
300 byte	0 (0.06)	0.03 (0.09)	0.04 (0.09)	0.05 (0.05)	0.02 (0.02)
600 byte	0 (0.06)	0.02 (0.09)	0.03 (0.08)	0.02 (0.02)	0 (0.01)
900 byte	0 (0.05)	0.01 (0.06)	0.02 (0.05)	0.01 (0.01)	0 (0.01)

Table 2.5: PSNR gains of priority-aware over priority-agnostic fragmentation (without slice fragmentation) for Silent at 960 Kbps (1080 Kbps).

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.2 (2.1)	0.3 (2.5)	0.9 (2.3)	1.6 (0.8)	2 (2.1)
300 byte	0.2 (2.5)	0.7 (2.7)	0.8 (2.1)	1.1 (0.8)	0.8 (0.8)
600 byte	0.4 (2.1)	0.5 (2.3)	1.1 (2.1)	0.3 (0.6)	0.3 (0.4)
900 byte	0.2 (2.3)	0.1 (1.5)	0.4 (2)	0.3 (0.5)	0.3 (0.5)

(b)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0 (0.07)	0.01 (0.09)	0.04 (0.09)	0.1 (0.05)	0.1 (0.1)
300 byte	0 (0.09)	0.03 (0.1)	0.03 (0.07)	0.04 (0.05)	0.03 (0.03)
600 byte	0 (0.07)	0.03 (0.1)	0.04 (0.08)	0.01 (0.02)	0.01 (0.01)
900 byte	0 (0.07)	0.01 (0.06)	0.01 (0.05)	0 (0.01)	0 (0.01)

is also observed for the VQM gain in Tables 2.4(b) and 2.5(b). Smaller slice sizes provide relatively more improvement in the perceptual video quality at the receiver.

This inflexibility can be mitigated by allowing the fragmentation of slices by suitably modifying the decoder to handle partial slice data. This further increases the PSNR gain that can be achieved by priority-aware fragmentation as discussed below. Table 2.6 shows the PSNR gains achieved by priority-aware (with slice fragmentation) over priority-agnostic fragmentation for Foreman and Silent at 720, 960, and 1080 Kbps. Table 2.7 shows the VQM gains achieved corresponding to the PSNR gains in Table 2.6. The priority-aware fragmentation achieves better PSNR and VQM values as compared to the priority-agnostic fragmentation for all the slice sizes and BERs.

Table 2.6: Average PSNR gain of priority-aware (with slice fragmentation) over priority-agnostic fragmentation for Foreman (Silent) at (a) 720 Kbps, (b) 960 Kbps, and (c) 1080 Kbps.

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.5 (0.4)	0.3 (0.2)	0.2 (0.1)	0.2 (0.1)	0.1 (0.2)
300 byte	0.5 (0.3)	1.1 (0.9)	1.2 (1)	1.7 (1.8)	2.1 (1.8)
600 byte	0.7 (0.6)	2.6 (2.1)	2.8 (2.5)	4.2 (5.3)	4.2 (5.2)
900 byte	1 (0.9)	3.1 (3.4)	4.2 (4)	5.1 (6.8)	5.9 (7.5)

(b)

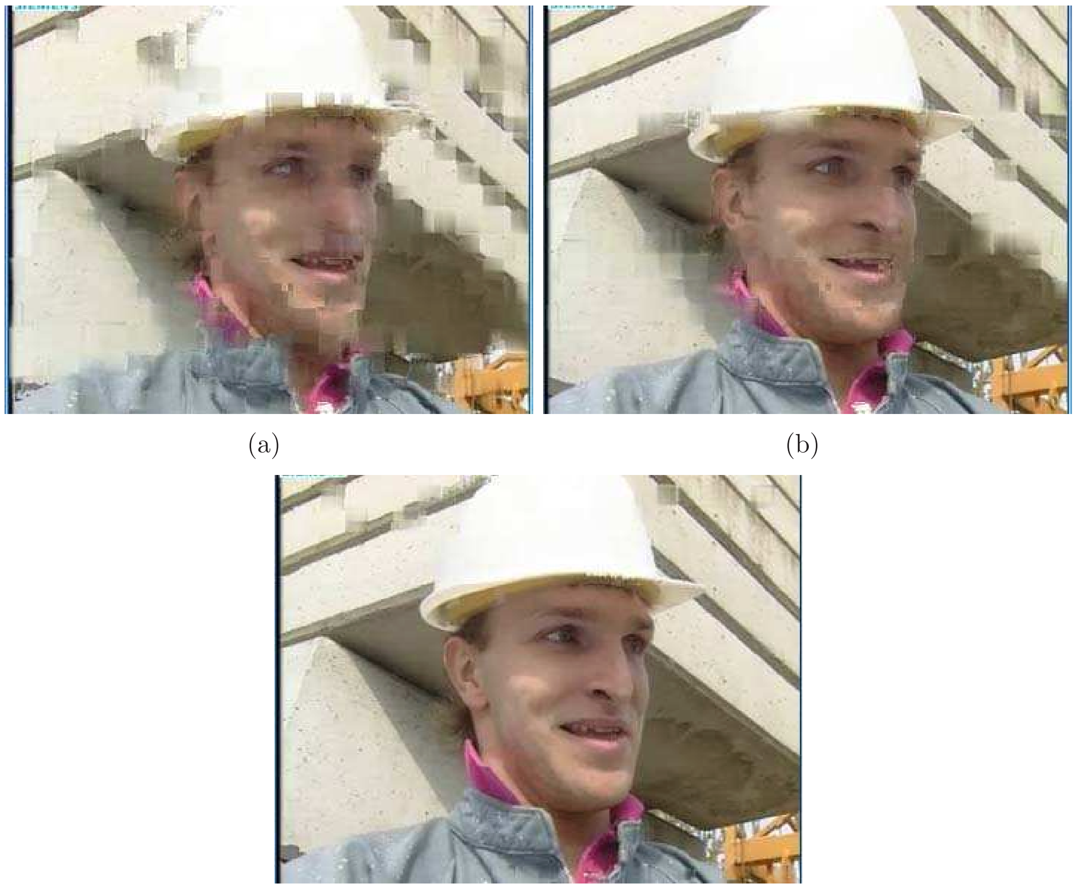
BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.4 (0.7)	1.3 (1.3)	1.4 (1.7)	1.2 (1.7)	1.5 (2.2)
300 byte	0.4 (0.3)	1.5 (1.6)	1.5 (1.3)	0.7 (1.2)	1 (2)
600 byte	0.4 (0.6)	1.3 (0.9)	1.1 (1.2)	1.5 (2.4)	2.9 (4.2)
900 byte	0.6 (0.3)	0.9 (0.8)	0.8 (1.1)	2.9 (4.5)	4.6 (6.7)

(c)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	1.8 (1.9)	1.6 (2.5)	1.8 (2.2)	0.9 (0.6)	1.6 (2)
300 byte	1.6 (2.5)	1.4 (2.6)	1.1 (2.6)	0.7 (0.8)	1.3 (1.5)
600 byte	1.7 (2.1)	1 (2.2)	1 (2.1)	1.9 (2.1)	2.9 (3.7)
900 byte	1.7 (2.3)	0.9 (2)	0.7 (2.3)	2.8 (3.6)	4.5 (5.8)

For Foreman, a maximum PSNR gain of 5.9 dB and corresponding VQM gain of 0.25 is achieved at 720 Kbps and 900 byte slice size at channel BER of 10^{-4} as shown in Tables 2.6(a) and 2.7(a), respectively. Similarly, 4.6 dB PSNR gain and corresponding VQM gain of 0.23 is achieved for Foreman encoded at 960 Kbps for the same slice size and channel BER in Tables 2.6(b) and 2.7(b), respectively. The maximum gains for the Silent video sequence are 7.5 dB at 720 Kbps (Table 2.6(a)) and 6.7 dB at 960 Kbps (Table 2.6(b)) for 900 byte slice size and channel BER of 10^{-4}. Also the corresponding VQM gains are 0.28 in Table 2.7(a) and 0.26 in Table 2.7(b). Similar gains are also observed when the video encoding rate (1080 Kbps) exceeds the channel bit rate of 1 Mbps.

Figure 2.14 shows the 126th frame of Foreman encoded at 720 Kbps using a slice size of 600 bytes at a BER of 5×10^{-5}. Figure 2.15 shows the 126th frame of Silent using these same specifications. The average VQM values are also shown in Figures 2.14 and 2.15.



(a) (b) (c)

Figure 2.14: 126th frame of Foreman encoded at 720 Kbps and BER of 5×10^{-5} in (a) baseline system: average PSNR=20.3 dB, average VQM=0.83, (b) priority-agnostic fragmentation: average PSNR=25.2 dB, average VQM=0.68, and (c) priority-aware fragmentation: average PSNR=29.9 dB, average VQM=0.46.

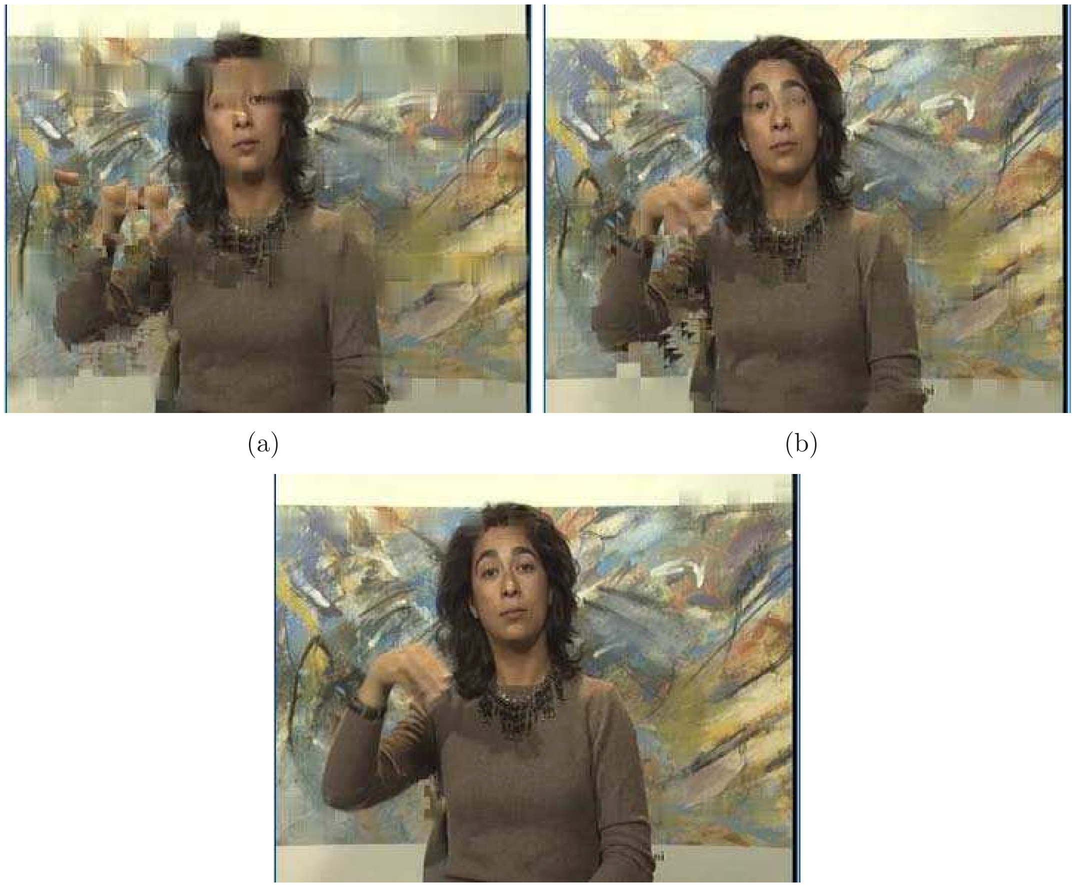


Figure 2.15: 126th frame of Silent encoded at 720 Kbps and BER of 5×10^{-5} in (a) baseline system: average PSNR=22 dB, average VQM=0.81, (b) priority-agnostic fragmentation: average PSNR=28.5 dB, average VQM=0.68, and (c) priority-aware fragmentation: average PSNR=33.3 dB average VQM=0.47.

Table 2.7: Average VQM gain of priority-aware (with slice fragmentation) over priority-agnostic fragmentation for Foreman (Silent) at (a) 720 Kbps, (b) 960 Kbps, and (c) 1080 Kbps.

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)
300 byte	0.01 (0.01)	0.03 (0.04)	0.05 (0.04)	0.1 (0.13)	0.1 (0.11)
600 byte	0.01 (0.01)	0.08 (0.07)	0.12 (0.12)	0.22 (0.21)	0.22 (0.18)
900 byte	0.02 (0.03)	0.12 (0.13)	0.2 (0.18)	0.29 (0.31)	0.25 (0.28)

(b)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.02 (0.01)	0.04 (0.05)	0.09 (0.08)	0.1 (0.11)	0.11 (0.11)
300 byte	0.02 (0.01)	0.07 (0.06)	0.09 (0.09)	0.08 (0.09)	0.08 (0.1)
600 byte	0.02 (0.01)	0.07 (0.05)	0.07 (0.08)	0.12 (0.12)	0.17 (0.19)
900 byte	0.02 (0.01)	0.04 (0.03)	0.06 (0.06)	0.19 (0.19)	0.23 (0.26)

(c)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.09 (0.08)	0.11 (0.1)	0.12 (0.11)	0.07 (0.05)	0.1 (0.11)
300 byte	0.08 (0.08)	0.11 (0.11)	0.12 (0.1)	0.07 (0.05)	0.08 (0.09)
600 byte	0.08 (0.07)	0.09 (0.1)	0.11 (0.1)	0.11 (0.12)	0.16 (0.19)
900 byte	0.08 (0.09)	0.09 (0.1)	0.10 (0.09)	0.18 (0.18)	0.21 (0.24)

2.5 Acknowledgement

Chapter 2 of this dissertation contains material that appears in Kashyap K. R. Kambhatla, S. Paluri, S. Kumar, and P. Cosman, and "Wireless H.264 Video Quality Enhancement through Optimal Prioritized Packet Fragmentation", *IEEE Transactions on Multimedia*, Vol. 14, No. 5, pp. 1480-1495, Oct. 2012 and Kashyap K. R. Kambhatla, S. Kumar, and P. Cosman, "Prioritized Packet Fragmentation for H.264 Video", *IEEE International Conference on Image Processing*, Brussels, Belgium, pp. 3233-3236, 11-14 Sept. 2011. I was the primary author and co-author Seethal Paluri had contributed to a part of the work. Co-authors Dr. Pamela Cosman and Dr. Sunil Kumar directed and supervised the research which forms the basis for Chapter 2.

Chapter 3

Cross-Layer Prioritized Video Packetization and Error Protection

3.1 Introduction

In this chapter, we describe our cross-layer scheme which minimizes the expected received video distortion by jointly optimizing the packet sizes at the APP layer and estimating their FEC code rates to be allocated at the PHY layer for noisy channels. Some low priority slices are also discarded in order to increase the protection to more important slices and meet the channel bit-rate limitations. Our proposed scheme ensures that higher priority slices which contribute more distortion are sent in smaller packets with stronger FEC coding. At the same time, it also efficiently controls the overhead incurred from the total protocol header bits associated with the formed packets. The distortion contributed by each slice is determined by its CMSE. Simulation results show that the proposed scheme efficiently transmits video over noisy channels.

To avoid the delays associated with optimizing the packet sizes and their associated FEC code rates for entire slices of a GOP, we extend our scheme to work on each frame independently by predicting its expected channel bit budget. This prediction uses a GLM developed over the factors (a) normalized CMSE per frame, (b) channel SNR, and (c) normalized compressed frame bit budget allocated by the H.264 encoder. The three factors are determined from a video dataset that spans

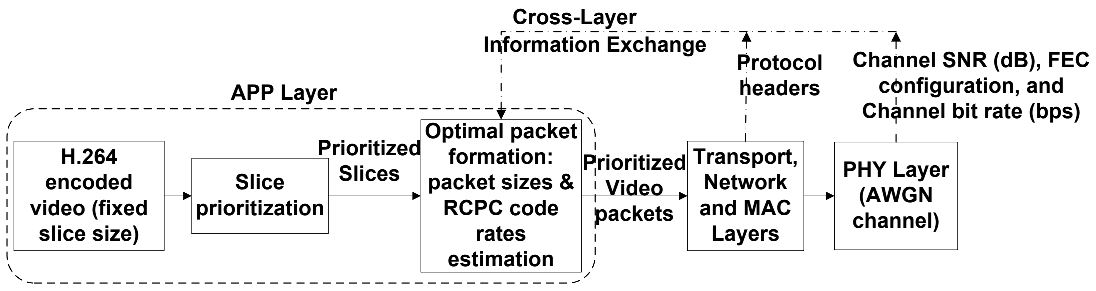

Cross-Layer
Information Exchange
APP Layer
Protocol headers
Channel SNR (dB), FEC configuration, and Channel bit rate (bps)
H.264 encoded video (fixed slice size)
Slice prioritization
Prioritized Slices
Optimal packet formation: packet sizes & RCPC code rates estimation
Prioritized Video packets
Transport, Network and MAC Layers
PHY Layer (AWGN channel)

Figure 3.1: Flow diagram of proposed cross-layer system for RTP/UDP/IP video transmission.

high, medium, and low motion complexity. Further, to avoid the complexity associated with computing the CMSE distortion contributed by a video slice, we use our low-complexity GLM defined in [50] for predicting the slice CMSE.

This chapter is organized as follows. Section 3.2 gives an overview of our proposed cross-layer approach. Section 3.3 discusses packet formation and optimal packet code rate allocation. Section 3.4 discusses the problem formulation for other error protection schemes. Simulation results and performance comparisons for an AWGN channel are presented in Section 3.5. Section 3.6 investigates the feasibility of our proposed cross-layer approach for live streaming. It also discusses the factors influencing the frame bit budget. Section 3.7 proposes and validates our GLM model for predicting the frame bit budget for live streaming.

3.2 Proposed Cross-Layer Approach

Figure 3.1 illustrates a flow diagram of our proposed cross-layer approach at the transmitter. The APP layer carries out two functions: CMSE based slice prioritization and optimal packet formation (illustrated further in Figure 3.2) for H.264 video slices.

3.2.1 CMSE Computation/Prediction of H.264 Video Slices

The video frames in a GOP are encoded using the fixed slice size configuration in H.264/AVC, where MBs of a frame are aggregated into slices with fixed size [2,68]. The slices are prioritized based on their CMSE values computed using Equation 2.1 in Chapter 2. However, the computation of slice CMSE introduces high computational overhead as it requires decoding the entire GOP for every slice loss. This overhead can be avoided by predicting the slice CMSE using our low-complexity GLM recently proposed in [50]. This model reliably predicts the slice CMSE values by extracting the encoded frame and the error frame features. The encoded frame features consist of motion characteristics, signal characteristics, maximum residual energy, and total number of MB sub-partitions in a slice. The error frame features consist of the temporal duration, initial mean square error, and initial structural similarity index. The actual slice CMSE values were used as ground truth. The readers are encouraged to refer to [50] for more details. The slice contributing the highest distortion is the most important slice (i.e., highest priority). This process defines the relative importance order for the slices in the GOP [1]. Note that our joint video packetization and error protection scheme proposed in this chapter will also work well with other slice distortion computation schemes such as Li and Liu [69].

3.2.2 H.264 Video Packet Formation

The optimal packet formation block uses a joint optimization scheme to form variable-sized packets (by aggregating pre-encoded slices according to their CMSE) and estimate their corresponding optimal FEC code rates that are applied at the PHY layer, in order to minimize the received video distortion as will be discussed in Section 3.3.

The FEC configuration contains a mother code rate and a family of rate compatible punctured convolutional (RCPC) code rates [70]. We use binary phase shift keying (BPSK) modulation and the packet size is constrained by the wireless network MTU [71]. The optimal packet formation block uses the information about the MTU size, RTP/UDP, IP and MAC layer headers which remain unchanged for a given network, and the channel SNR, FEC configuration and channel bit rate information from the PHY layer. The RTP/UDP/IP overhead appended to each packet formed at the APP layer is 4 bytes after robust header compression (RoHC) [60]. Each packet is also appended with 50 bytes of MAC and PHY layer headers. Our scheme studies the video quality improvement that can be achieved by exploiting the slice priorities and the trade-offs between the priority-adaptive packet sizes and RCPC code rates with the total incurred overhead (FEC + network protocol header) for a given channel SNR, channel bit rate, and source bit rate.

3.3 Expected Video Distortion Minimization

We introduce a DP-based approach to minimize the expected video distortion. R_{CH} is the channel transmission rate in bits per second. The video is encoded at a frame rate of f_s fps. The total outgoing bit budget for a GOP of length L_G frames is $\frac{R_{CH} L_G}{f_s}$. We use n_s to denote the total number of slices generated within a GOP; n_s is a constant. We use n_p to denote the number of packets formed from these slices in the GOP; n_p is variable. $S_p(i)$ is the i^{th} packet size before adding network headers of size h bits and parity bits from the selected RCPC code. The RCPC code rates are chosen from a candidate set, $\mathbf{R}$, of punctured code rates $\{R_1, R_2, R_3, ..., R_K\}$. The number of packets discarded is n_{pd} which will be described in Sections 3.3.2 and 3.3.4.

3.3.1 Packet Formation (PF) Block

The proposed scheme is a recursive process between two blocks: Packet formation (PF) block and Optimal RCPC code rate allocation (OCRA) block as shown in Figure 3.2. The PF block initializes $n_p = n_s$ and $n_{pd} = 0$, and calls the OCRA block after sorting the $n_p = n_s$ packets of a GOP in descending priority order. The OCRA block determines the optimal RCPC packet code rates and the number of packets discarded, n_{pd}, to minimize a dual cost function value (computed over the GOP) described in Section 3.3.2. The OCRA block then forwards the computed parameters to the PF block as shown in Figure 3.2.

The PF block aggregates the two packets with least CMSE contribution from

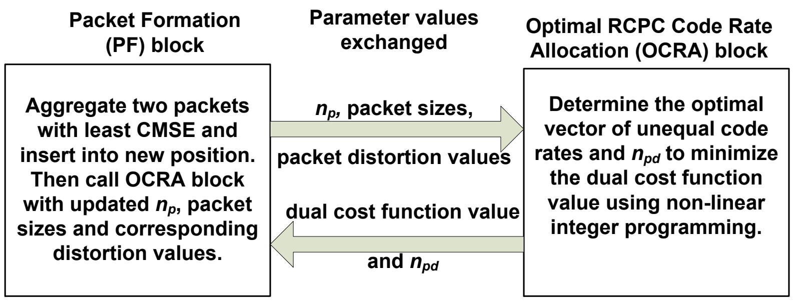

Packet Formation (PF) block
Parameter values exchanged
Optimal RCPC Code Rate Allocation (OCRA) block
Aggregate two packets with least CMSE and insert into new position. Then call OCRA block with updated n_p, packet sizes and corresponding distortion values.
n_p, packet sizes,
packet distortion values
dual cost function value
and n_{pd}
Determine the optimal vector of unequal code rates and n_{pd} to minimize the dual cost function value using non-linear integer programming.

Figure 3.2: Block diagram of proposed dynamic programming approach to estimate packet sizes and their FEC.

the remaining set of packets not discarded by the OCRA block. The aggregated packet is inserted into a new position in the sorted list based on its distortion computed as the sum of the CMSE values of both packets. This maintains the decreasing order of packet distortion. It calls the OCRA block again to determine optimal RCPC code rates for the new set of packets. The parameters shown in Figure 3.2 are exchanged recursively between the blocks until aggregating packets is no longer beneficial to reduce the dual cost function value. As an example, Figure 3.3 shows one iteration of our proposed scheme in the PF block. The first packet in each iteration is the most important and contributes the maximum distortion. After returning from the OCRA block, the number of packets is updated to $n_p = n_s - n_{pd}$ since n_{pd} packets were dropped in the OCRA block. The two least important packets are then aggregated and inserted into a new position while the remaining packets are simply retained. The aggregated packet is at position $n_p - j$. The $n_p - 1$ packets with their sizes and distortion values are once again sent to the OCRA block, to estimate their new optimal packet code rates.

The size of the aggregated packets is constrained by the MTU size for wireless networks. Aggregating packets reduces the total overhead from network protocol headers; the bits saved are used to increase the FEC protection to more important packets. Since the PF block aggregates the least important packets, this ensures that packets contributing higher distortion are transmitted with smaller sizes, and the OCRA block ensures that they have stronger FEC hence lower packet error probabilities.

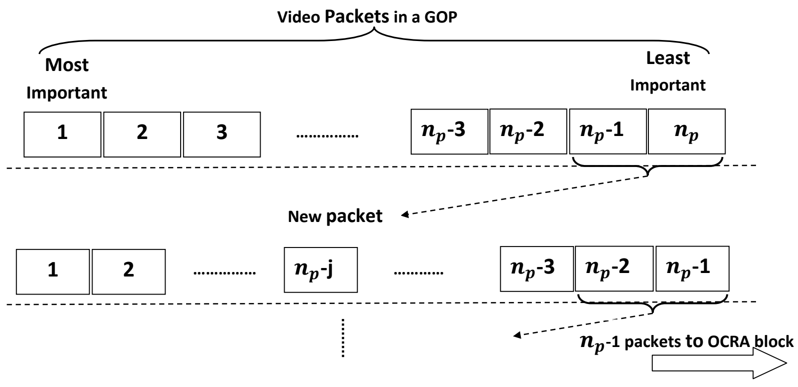


Figure 3.3: Packet formation in PF block: linearity model for packet CMSE and size.

3.3.2 Distortion Minimization with Optimal RCPC Code Rate Allocation (OCRA) Block

The distortion due to the compression is neglected in this formulation because the slices are pre-encoded and assumed to be at relatively high quality, so compression distortion is small compared to distortion from slice losses and discards. The initial values are $n_p = n_s$ and $n_{pd} = 0$. The expected video distortion within a GOP, $E[\tilde{D}_{GOP}]$, is modeled as the sum of the distortion due to channel-induced packet loss and distortion from packets discarded at the sender as in [43].

$$E[\tilde{D}_{GOP}] = \sum_{i=1}^{n_p-n_{pd}} E[\tilde{D}_p(i)] + \sum_{i=n_p-n_{pd}+1}^{n_p} D_p(i) \tag{3.1}$$

$D_p(i)$ is the distortion caused due to the loss of packet i and is computed as the sum of the CMSE of individual slices contained in the packet. Each video packet is appended with a h bit network header and parity bits for a code rate r_i selected from the set $\mathbf{R}$. We consider a discrete-time memoryless AWGN channel. A video packet is in error if at least one bit is in error after channel decoding at the receiver. If the bit errors following decoding were independent from bit to bit, then the packet error probability, $p_{pkt}(i)$, which depends on the channel SNR, packet size, and the selected RCPC code rate could be computed as in [20, 36, 43, 45, 72, 73]:

$$p_{pkt}(i) = 1 - (1 - p_b(SNR, r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)} \tag{3.2}$$

where $p_b(SNR, r_i)$ is the bit error probability after channel decoding for code rate r_i. We use the above expression for packet error probability in the design procedure to determine the FEC rates. For a given value of n_{pd}, the distortion due to the discarded packets in Equation 3.1 is a constant K_1. The optimization problem for minimizing expected video distortion over the GOP by allocating optimal code rates is formulated as:

$$\begin{gathered}
\min_{\mathbf{r}} \left\{ \sum_{i=1}^{n_p - n_{pd}} \left[1 - (1 - p_b(SNR, r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)} \right] D_p(i) + K_1 \right\} \\
= K_1 + \min_{\mathbf{r}} \left\{ \sum_{i=1}^{n_p - n_{pd}} \left[1 - (1 - p_b(SNR, r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)} \right] D_p(i) \right\} \\
\text{subject to} \\
(C1) \quad \sum_{i=1}^{n_p - n_{pd}} \frac{h+S_p(i)}{r_i} \leq \left(\frac{R_{CH} L_G}{f_s} \right) \\
(C2) \quad r_{i-1} \leq r_i \quad \text{for} \quad i = 2, 3, 4, 5, 6, ..., (n_p - n_{pd}) \\
\text{where} \quad \mathbf{r} = \left[r_1, r_2, ..., r_{n_p - n_{pd}} \right] \text{ and } r_i \,\epsilon\, \mathbf{R}
\end{gathered} \tag{3.3}$$

Constraint 1 in Equation 3.3 is the channel bit rate constraint. Constraint 2 ensures that higher priority packets have code rates at least as good as those allocated to lower priority packets. This speeds up the optimization process by narrowing down the selection set of packet code rates. To solve this non-linear integer programming problem, we first relax the constrained optimization problem in Equation 3.3 to an unconstrained problem [74,75]. By absorbing the constraints into the objective using Lagrange multipliers $\boldsymbol{\lambda} = \left[\lambda_1, \lambda_2, ..., \lambda_{n_p - n_{pd}} \right]$ with each $\lambda_i \,\epsilon\, \mathbb{R}^+$, we construct the Lagrangian cost function as:

$$\begin{aligned}
F_{GOP}(\mathbf{r}, \boldsymbol{\lambda}) = K_1 &+ \sum_{i=1}^{n_p - n_{pd}} \left[1 - (1 - p_b(SNR, r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)} \right] D_p(i) \\
&+ \lambda_1 \left(\sum_{i=1}^{n_p - n_{pd}} \frac{h+S_p(i)}{r_i} - \frac{R_{CH} L_G}{f_s} \right) + \sum_{i=2}^{n_p - n_{pd}} \lambda_i (r_{i-1} - r_i) \\
\text{where} \quad \boldsymbol{\lambda} &= \left[\lambda_1, \lambda_2, ..., \lambda_{n_p - n_{pd}} \right]
\end{aligned} \tag{3.4}$$

We form the dual cost function $d_{GOP}(\boldsymbol{\lambda})$ by minimizing the Lagrangian cost function for a given $\boldsymbol{\lambda}$, where $\boldsymbol{\lambda}$ is searched using a subgradient approach which will be discussed in Section 3.3.3. Let $\mathcal{C}$ be the space of all possible combinations of $r_i, i = 1, 2, ..., n_p - n_{pd}$ selected from $\mathbf{R}$ that can be applied to the packets before transmission. The dual function is computed as:

$$\begin{aligned}
d_{GOP}(\boldsymbol{\lambda}) &= \min_{\mathbf{r}\,\epsilon\,\mathcal{C}}\ F_{GOP}(\mathbf{r},\boldsymbol{\lambda}) \\
&= \min_{\mathbf{r}\,\epsilon\,\mathcal{C}}\ \sum_{i=1}^{n_p-n_{pd}} \left\{ \left(1-(1-p_b(SNR,r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)}\right) D_p(i) \right\} \\
&\qquad +\lambda_1 \left(\sum_{i=1}^{n_p-n_{pd}} \frac{h+S_p(i)}{r_i} - \frac{R_{CH}L_G}{f_s}\right) + \sum_{i=2}^{n_p-n_{pd}} \lambda_i(r_{i-1}-r_i) + K_1 \\
&= K_2 + \min_{\mathbf{r}\,\epsilon\,\mathcal{C}} \sum_{i=1}^{n_p-n_{pd}} \left(1-(1-p_b(SNR,r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)}\right) D_p(i) \\
&\qquad + \min_{\mathbf{r}\,\epsilon\,\mathcal{C}} \sum_{i=1}^{n_p-n_{pd}} \lambda_1 \left(\frac{h+S_p(i)}{r_i}\right) + \sum_{i=2}^{n_p-n_{pd}} \lambda_i(r_{i-1}-r_i)
\end{aligned} \tag{3.5}$$

$K_2 = K_1 - \lambda_1\left(\frac{R_{CH}L_G}{f_s}\right)$ in Equation 3.5 is a constant and the computation of $d_{GOP}(\boldsymbol{\lambda})$ can be further simplified as follows.

Let $A(r_i) = D_p(i)\left(1-(1-p_b(SNR,r_i))^{\left(\frac{h+S_p(i)}{r_i}\right)}\right) + \lambda_1\left(\frac{h+S_p(i)}{r_i}\right)$. Then we can modify the first term in Equation 3.5 as:

$$\begin{aligned}
&\min_{\mathbf{r}\,\epsilon\,\mathcal{C}} \left\{ \sum_{i=1}^{n_p-n_{pd}} A(r_i) + \sum_{i=2}^{n_p-n_{pd}} \lambda_i(r_{i-1}-r_i) \right\} \\
&= \min_{\mathbf{r}\,\epsilon\,\mathcal{C}} A(r_1) + A(r_2) + ... + A(r_{n_p-n_{pd}}) + \lambda_2(r_1-r_2) + \lambda_3(r_2-r_3) + \\
&\qquad\qquad ... + \lambda_{n_p-n_{pd}}(r_{n_p-n_{pd}-1} - r_{n_p-n_{pd}}) \\
&= \min_{r_1\,\epsilon\,\mathbf{R}} \{A(r_1)+\lambda_2(r_1)\} + \min_{r_2\,\epsilon\,\mathbf{R}} \{A(r_2)-\lambda_2(r_2)+\lambda_3(r_2)\} + ... + \\
&\quad \min_{r_{n_p-n_{pd}-1}\,\epsilon\,\mathbf{R}} \left\{A(r_{n_p-n_{pd}-1}) - \lambda_{n_p-n_{pd}-1}(r_{n_p-n_{pd}-1}) + \lambda_{n_p-n_{pd}}(r_{n_p-n_{pd}-1})\right\} \\
&\quad + \min_{r_{n_p-n_{pd}}\,\epsilon\,\mathbf{R}} \left\{A(r_{n_p-n_{pd}}) - \lambda_{n_p-n_{pd}}(r_{n_p-n_{pd}})\right\} \\
&= \min_{r_1\,\epsilon\,\mathbf{R}} \{A(r_1)+\lambda_2(r_1)\} + \sum_{i=2}^{n_p-n_{pd}-1} \min_{r_i\,\epsilon\,\mathbf{R}} \{A(r_i) + r_i(\lambda_{i+1}-\lambda_i)\} \\
&\quad + \min_{r_{n_p-n_{pd}}\,\epsilon\,\mathbf{R}} \left\{A(r_{n_p-n_{pd}}) - \lambda_{n_p-n_{pd}}(r_{n_p-n_{pd}})\right\}
\end{aligned}$$

The dual function can now be expressed in terms of function $A(r_i)$ as:

$$\begin{aligned}
d_{GOP}(\boldsymbol{\lambda}) &= K_2 + \min_{r_1\,\epsilon\,\mathbf{R}} \{A(r_1)+\lambda_2(r_1)\} \\
&\quad + \sum_{i=2}^{n_p-n_{pd}-1} \min_{r_i\,\epsilon\,\mathbf{R}} \{A(r_i)+r_i(\lambda_{i+1}-\lambda_i)\} \\
&\quad + \min_{r_{n_p-n_{pd}}\,\epsilon\,\mathbf{R}} \left\{A(r_{n_p-n_{pd}}) - \lambda_{n_p-n_{pd}}(r_{n_p-n_{pd}})\right\} \\
&= K_2 + \sum_{i=1}^{n_p-n_{pd}} \min_{r_i\,\epsilon\,\mathbf{R}} \tilde{F}_{GOP,i}(r_i,\lambda_i)
\end{aligned} \tag{3.6}$$

$$\text{where } \tilde{F}_{GOP,i}(r_i,\lambda_i) = \begin{cases} A(r_1)+\lambda_2(r_1) \text{ for } i=1 \\ A(r_i)+r_i(\lambda_{i+1}-\lambda_i) \text{ for } i=2,3,4,...,n_p-n_{pd}-1 \\ A(r_{n_p-n_{pd}}) - \lambda_{n_p-n_{pd}}(r_{n_p-n_{pd}}) \text{ for } i=n_p-n_{pd} \end{cases}$$

The minimum of the dual cost function for a given $\boldsymbol{\lambda}$ can be found by minimizing the sub-Lagrangian cost functions $\tilde{F}_{GOP,i}(r_i, \lambda_i)$ individually. The solution space of the minimization of $F_{GOP}(\mathbf{r}, \boldsymbol{\lambda})$ is $(K+1)^{(n_p-n_{pd})}$. Since we can minimize the sub-Lagrangians individually, $d_{GOP}(\boldsymbol{\lambda})$ can be computed with only $(n_p - n_{pd})(K+1)$ evaluations of $\tilde{F}_{GOP,i}(r_i, \lambda_i)$ and comparisons [74]. This reduces the computational complexity involved in deriving the optimal set of packet sizes and their code rates. The frame-based optimization schemes use the slices of a frame (instead of a GOP) to form n_p packets. Therefore, their optimization complexity is much smaller than for a GOP-based scheme.

3.3.3 Determination of $\boldsymbol{\lambda}$

We use the subgradient method [74] to search for the best $\boldsymbol{\lambda}$ over the space $\mathcal{C}$. The dual function $d_{GOP}(\boldsymbol{\lambda})$ is a concave function of $\boldsymbol{\lambda}$ even when the problem in the primal domain is not convex [74,75]. Therefore the optimal $\boldsymbol{\lambda}$ is found by solving $\max_{\boldsymbol{\lambda}\,\epsilon\,\mathbb{R}^+} d_{GOP}(\boldsymbol{\lambda})$. Since the dual is a piecewise linear concave function [74], it may not be differentiable at all points. Nevertheless, subgradients can still be found and are used to compute the optimal value [74]. It can be shown that the subgradient is a descent direction of the Euclidean distance to the set of maximum points of the dual function [74]. This property is used in the subgradient method for the optimization of a non-smooth function. The subgradient method is an iterative search algorithm for $\boldsymbol{\lambda}$. In each iteration, λ_i^{k+1} is updated by the subgradient ξ_i^k of $d_{GOP}(\boldsymbol{\lambda})$ at λ_i^k:

$$\lambda_i^{(k+1)} = \max(0, \lambda_i^k + s_k \xi_i^k / \|\boldsymbol{\xi}^k\|) \tag{3.7}$$

where s_k is the step size. Based on the derivation in [74], the subgradients $\boldsymbol{\xi}^k$ of $d_{GOP}(\boldsymbol{\lambda})$ at $\boldsymbol{\lambda}^k$ are

$$\begin{aligned} \xi_1^k &= g(\mathbf{r}^k) - \tfrac{R_{CH} L_G}{f_s} = \textstyle\sum_{i=1}^{n_p-n_{pd}} \left(\tfrac{h+S_p(i)}{r_i}\right) - \tfrac{R_{CH} L_G}{f_s} \\ \xi_i^k &= r_{i-1} - r_i \text{ for } i = 2, 3, 4, ..., n_p - n_{pd} \end{aligned} \tag{3.8}$$

where $g(.)$ is the rate constraint function of the problem and $\mathbf{r}^k = \left[r_1^k, r_2^k, ..., r_{n_p-n_{pd}}^k\right]$ is the solution to the term $\min_{\mathbf{r}\,\epsilon\,\mathcal{C}}\ F_{GOP}(\mathbf{r}, \boldsymbol{\lambda}^k)$ in Equation

3.5.

3.3.4 Discarding Packets

By explicitly discarding a small number of low priority packets, we gain additional room for packet size adaptation and FEC, and can derive significant benefits overall. To allow either the discarding of less important packets or sending them unprotected, the candidate set of punctured code rates $\mathbf{R}$ is modified to $\{1, R_1, R_2, R_3, ..., R_K, \infty\}$. This neither changes the objective function to be minimized in Equation 3.3 nor does it affect the optimization algorithm discussed in Section 3.3.2. If the code rate of packet i, $r_i = \infty$, then its probability of bit error $p_b(SNR, r_i) = 1$ causing it to be discarded. The induced distortion is accounted for in the overall expected distortion $E[\tilde{D}_{GOP}]$ through component K_1 in Equation 3.3. If $r_i = 1$, the video packet is transmitted uncoded over the channel.

3.4 Problem Formulation of other Error Protection Schemes

We compare our scheme discussed in Section 3.3, denoted from now on as **DP-UEP**, with the **Dual15** [43], and the **EEP-slice-ENH** schemes. The **Dual15** scheme treats every slice as a packet and does not aggregate them to save on the total overhead incurred from network protocol headers of 54 bytes being associated with every slice. It finds the optimal set of punctured code rates to protect the slices based on their importance (i.e., using UEP) and minimize expected received video distortion.

The **EEP-slice-ENH** is similar to our proposed scheme **DP-UEP** in the way pre-encoded slices are aggregated to form packets with more important ones having smaller sizes and error probabilities and also the less important packets being discarded to meet the channel bit rate constraint. However, unlike **DP-UEP**, all packets in **EEP-slice-ENH** are equally protected with the best possible EEP code rate. This scheme is broadly similar to other packet (or payload) size adaptation schemes in the literature [12,14,37,40,76]. The objective of this scheme is to minimize the expected received video distortion and it is formulated in a manner similar to Equation 3.3:

$$\begin{aligned}
&\min_{r\,\epsilon\,\mathbf{R}}\left\{\sum_{i=1}^{n_p-n_{pd}}\left[1-(1-p_b(SNR,r))^{\left(\frac{h+S_p(i)}{r}\right)}\right]D_p(i)+K_1\right\}\\
&=K_1+\min_{r\,\epsilon\,\mathbf{R}}\left\{\sum_{i=1}^{n_p-n_{pd}}\left[1-(1-p_b(SNR,r))^{\left(\frac{h+S_p(i)}{r}\right)}\right]D_p(i)\right\}\\
&\qquad\text{subject to}\quad \sum_{i=1}^{n_p-n_{pd}}\frac{h+S_p(i)}{r}\leq\left(\frac{R_{CH}L_G}{f_s}\right)
\end{aligned}\tag{3.9}$$

Constraint 2 in Equation 3.3 is not valid here since r is no longer a vector. As in Equation 3.3, K_1 is the permanent distortion caused by the discarded packets and is constant for a given value of n_{pd}. Apart from the change that only a single λ and r value needs to be determined, the same DP-based approach described in Sections 3.3.1 and 3.3.2 is used to solve the optimization problem in Equation 3.9.

3.5 Performance of DP-UEP

In this section, we evaluate and compare the performance of **DP-UEP**, **Dual15**, and **EEP-slice-ENH** schemes with video quality measured by PSNR and VQM [66,67].

3.5.1 Simulation Setup

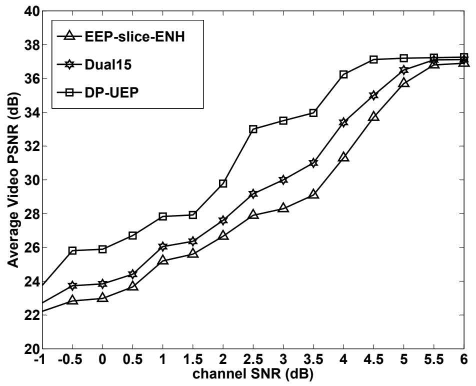
Two CIF (352 x 288) video sequences, Foreman and Silent, are used in our experiments. Silent has lower motion activity than Foreman. They are encoded using H.264/AVC JM 18.5 reference software [68] at an encoding rate of 720 Kbps and transmitted over a 2 Mbps AWGN channel. The GOP structure, length, and frame rate used for simulations in Chapter 2 are also used here. The slice size in the fixed slice size configuration of H.264/AVC is set to 300 bytes. The error concealment discussed in Chapter 2 is enabled for all the schemes evaluated in this chapter.

The total network protocol header size is 54 bytes per packet as discussed in Section 3.2.2. The mother code of the RCPC code has rate $\frac{1}{4}$ with memory M=4 and puncturing period P=8. Log-likelihood ratio (LLR) is used in the Viterbi decoder. The initial RCPC rates available are {(8/9), (8/10), (8/12), (8/14), (8/16), (8/18), (8/20), (8/22), (8/24), (8/26), (8/28), (8/30), (8/32)}. Two additional rates, 8/8 corresponding to no coding and ∞ corresponding to discarding are also included. The performance evaluation of the schemes is based on a bit-level simulation of the compressed videos using the derived packet sizes and FEC code rates over 100 realizations of every AWGN channel SNR. The simulation results in Sections 3.5.2 and 3.6 use the CMSE values computed from Equation 2.1 in Chapter 2.

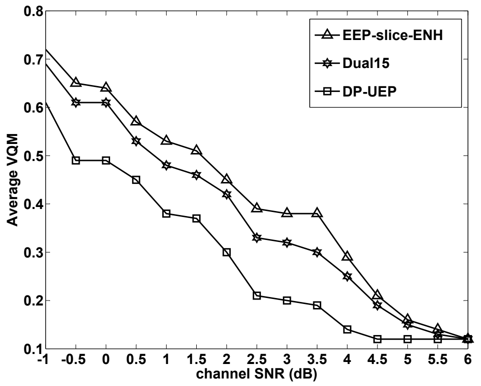
3.5.2 Performance Comparison

Figure 3.4 shows the average PSNR and VQM performance over an AWGN channel. As the channel SNR increases, the packet error decreases and the received videos achieve average PSNRs closer to their error-free PSNR values. The **EEP-slice-ENH** scheme performs the worst. Though it adapts the packet size to the video priority by aggregating the slices and discarding lower priority packets, it is still limited to providing equal protection to all the packets formed. The lowest and highest optimal EEP code rates derived across GOPs were $\left[\frac{8}{20}\ \frac{8}{14}\right]$. However, as the channel SNR deteriorates in Figure 3.4, the lowest code rate $\frac{8}{20}$ is insufficient to protect the packets from channel induced errors.

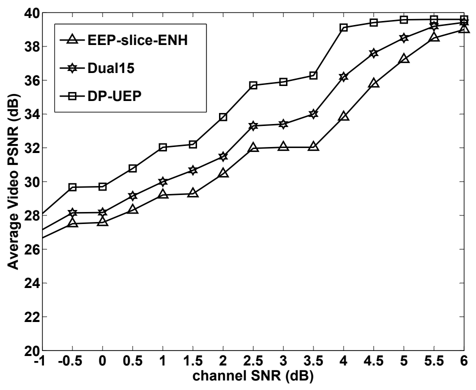
The **Dual15** scheme does not consider packet formation through slice aggregation and only performs optimal (UEP) RCPC code rate allocation to the slices (considered as individual packets) of each GOP [43]. It also discards least important slices, if required to meet the channel bit budget constraints. The slice error probability in the **Dual15** scheme is dependent on the optimal RCPC code rate allocated since the size of each slice is more or less the same. Also every slice in the **Dual15** scheme is attached with the 54 byte network protocol header resulting in more overhead. In contrast, our proposed **DP-UEP** scheme takes advantage of both the priority-adaptive packet sizes and optimal RCPC packet code rate allocation. Our **DP-UEP** scheme assigns optimal code rates as low as $\frac{8}{32}$ to the high priority packets with small packet sizes (e.g. 300 byte, which is the slice size used in encoding, or 600 byte obtained by aggregating two slices) and higher code rates to the lower priority packets with larger packet sizes within every GOP. The packet sizes of the low priority packets are restricted by the network MTU size of 1500 bytes. Figure 3.4



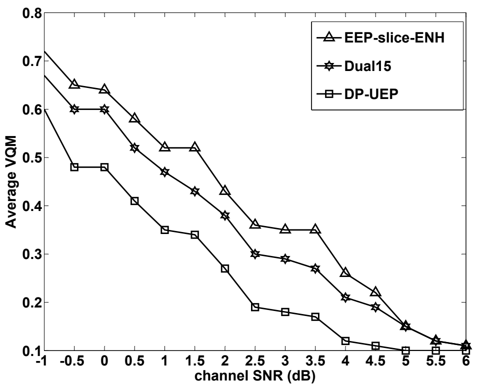
(a)



(b)



(c)



(d)

Figure 3.4: Average video PSNR (dB) and corresponding average VQM comparison computed over 100 realizations of each AWGN channel for Foreman:(a),(b), and Silent:(c),(d).

shows the improvement in video quality of our **DP-UEP** scheme compared to the **EEP-slice-ENH** and **Dual15** schemes. For example, at a channel SNR of 3 dB, the **EEP-slice-ENH**, **Dual15**, and **DP-UEP** schemes achieve average VQM values of 0.38, 0.32, and 0.2, and corresponding average PSNR values of 28.3 dB, 30.2 dB, and 33.5 dB, for Foreman. Our **DP-UEP** scheme achieves maximum PSNR gains of 3.5 dB for Foreman and 2.8 dB for Silent over **Dual15** at a channel SNR of 3 dB. The **DP-UEP** scheme also achieves maximum gains of 5.2 dB for Foreman and 4.3 dB for Silent over the **EEP-slice-ENH** scheme at channel SNR of 3 dB. Similar behavior is also observed in the VQM performance.

This considerable improvement in video quality achieved by our **DP-UEP** scheme can be explained by the following two factors: (*i*) the lower number of slices discarded per GOP shown in Figure 3.5, and (*ii*) the composition of the final transmitted bits in terms of the compressed source bits, network protocol headers, and FEC bits shown in Figure 3.6. Balancing the overhead due to the FEC parity bits allows the **Dual15** scheme to discard fewer slices per GOP as compared to the **EEP-slice-ENH** scheme. Our **DP-UEP** scheme further reduces the number of discarded slices as compared to the **Dual15** scheme by balancing both the overhead due to FEC parity bits as well as the network protocol headers attached to the packets formed by aggregating slices. For example, at a channel SNR of 3 dB in Figure 3.5, our **DP-UEP** scheme does not discard any slices whereas 20 and 35 slices are discarded in every GOP by the **Dual15** and **EEP-slice-ENH** schemes, respectively. As the channel SNR decreases, more slices are discarded by every scheme. For example, at a channel SNR of -1 dB, 101, 62, and 50 slices are discarded by the **EEP-slice-ENH**, **Dual15**, and **DP-UEP** schemes, respectively. This means that though we encode the video at a target bit rate of 720 Kbps, every scheme adjusts this bit rate by discarding the slices in order to minimize the expected received video distortion under the given channel SNR condition and bit budget constraints.

Figure 3.6 shows the bit contribution of the source, network protocol headers, and FEC to the total bits transmitted over a 2 Mbps channel at 3 dB channel SNR for Foreman. Our **DP-UEP** scheme transmits more source bits (i.e., a relatively higher bit rate) than the other two schemes by reducing the network protocol overhead as well as allocating optimal RCPC code rates based on packet priority. It also uses only

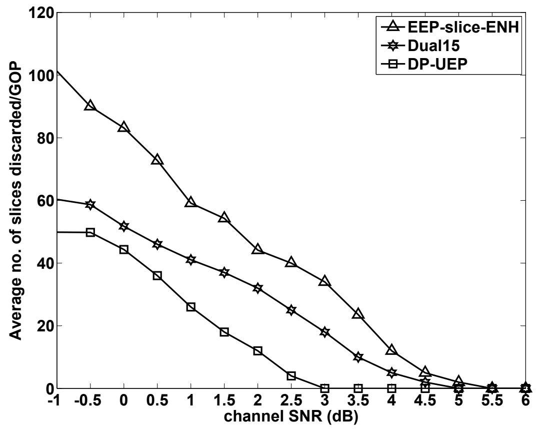


Figure 3.5: Average number of slices discarded per GOP in **EEP-slice-ENH**, **Dual15**, and **DP-UEP** for Foreman.

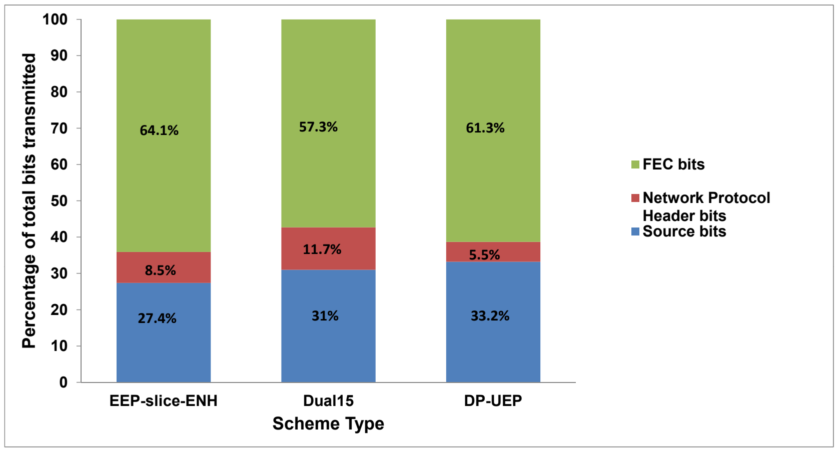


Figure 3.6: Distribution of the final output bits for Foreman at 3 dB channel SNR in **EEP-slice-ENH**, **Dual15**, and **DP-UEP** schemes.

5.5% bits for the network protocol overhead, compared to 8.5% and 11.5% overhead bits for **EEP-slice-ENH** and **Dual15**, respectively. Further, 61.3% bits are allocated for FEC overhead by **DP-UEP** compared to 57.3% in **Dual15**, thus providing better FEC protection. Although **EEP-slice-ENH** uses 64.1% FEC bits, it uses EEP which ignores packet priority. The **DP-UEP** scheme sends the highest percentage of source bits (i.e., 33.2%) which also correlates to no slices being discarded at 3 dB channel SNR, shown earlier in Figure 3.5. A similar trend is also observed for Silent, and for other channel SNRs.

3.6 Frame-Level DP-UEP Scheme

The **DP-UEP** scheme was designed for a pre-encoded video and the cross-layer optimization was performed over each GOP. Its computational complexity and delay are not suitable for live streaming applications, such as live sports events. In this section, we extend **DP-UEP** to be applied over the slices of a single frame instead of the entire GOP to reduce its computational complexity and delay. This requires **DP-UEP** to process the encoded slices of only one frame at a time in the PF and OCRA blocks (shown in Figure 3.2) instead of performing optimization over the slices of an entire GOP. Since a typical GOP consists of different frame types (i.e., IDR, I, P, and B), we require a good estimate of the channel bit budget for that frame in order to allocate the protocol header and FEC bits to its packets. Moreover, different frame types generate different numbers of slices that contribute different amounts of distortion based on the error propagation and video content. Therefore, we need to distribute the channel bit budget for a GOP among the different frames and to that extent we study the video factors which are most influential on the expected channel bit budget estimate of a frame. From now on, we refer to our **DP-UEP** scheme over the slices of the entire GOP as **DP-UEP**(GOP) and over slices of a frame as **DP-UEP**(frame).

Before investigating the important factors influencing the expected channel bit budget for each frame within the GOP, we study how well **DP-UEP**(frame) might perform compared to **DP-UEP**(GOP). We study the average PSNR and average VQM performance of **DP-UEP**(frame) by using the measured slice CMSE values and the channel bit budget allocated to each frame by the **DP-UEP**(GOP) for Foreman and Silent. Later in Section 3.7, we train a GLM for predicting the expected channel bit budget for each frame in real-time. To avoid the delays involved with processing an entire GOP, we will need to use an estimate of the frame bit budget rather than the actual bit budget allocated by the **DP-UEP**(GOP) scheme. However, analyzing the channel bit budget allocation, R_l for the frame l, by the **DP-UEP**(GOP) scheme can provide some motivation for whether the frame-based approach is worth pursuing. To compute R_l, we first derive the overhead bit budget proportion, w_{ovh}^{l} for the frame

l, from the result of the **DP-UEP**(GOP) scheme as:

$$w^l_{ovh} = \frac{\text{\# FEC bits for frame } l}{\text{\# FEC bits for whole GOP}} \tag{3.10}$$

This quantity, while it is explicitly the fraction of FEC bits which a particular frame gets relative to FEC bits for the whole GOP, is taken to be an estimate of overhead bits (both FEC and protocol header bits) which the frame gets relative to the overhead bits for the whole GOP. R_l is then evaluated using w^l_{ovh} for a video bit rate denoted by R_v as:

$$R_l = \sum_{i=1}^{n^l_s} S^l_p(i) + w^l_{ovh} \left\{ \frac{(R_{CH} - R_v)L_G}{f_s} \right\} \tag{3.11}$$

where n^l_s is the number of slices in frame l and $S^l_p(i)$ is the size of slice i in frame l. The video bit rate of 720 Kbps, used in our simulations in Section 3.5, is assigned to R_v. We determine the optimal packet sizes and their corresponding code rates separately for each frame in the GOP using the cross-layer DP-based approach discussed in Section 3.3. We observe that the average PSNR performance of **DP-UEP**(frame) is only slightly lower than that of **DP-UEP**(GOP) (shown later in Figure 3.8), but still higher than the **Dual15** scheme. A small drop in average PSNR and VQM is due to the fact that our optimization scheme for slices of each frame is sub-optimal compared to the DP-UEP(GOP) scheme. In other words, DP-UEP(frame) may have discarded some slices from a frame which were retained in the DP-UEP(GOP) scheme.

From the analysis of the **DP-UEP**(GOP) scheme, we observed that w^l_{ovh} for a frame l is dependent on the following video factors: (a) normalized CMSE for frame l, denoted as w^l_{cmse}, (b) normalized compressed frame bit budget, denoted as w^l_c, (c) channel SNR, and (d) video content. w^l_{cmse} is computed as the ratio of the total CMSE contribution of all slices in frame l to the total CMSE contribution of all slices in the GOP. w^l_c is computed as the ratio of the size of the compressed frame in bits to the total source bit rate for the GOP.

$$w^l_{cmse} = \frac{\sum_{i=1}^{n^l_s} D^l_p(i)}{\sum_{j=1}^{L_G} \sum_{i=1}^{n^j_s} D^j_p(i)} \quad ; w^l_c = \frac{\sum_{i=1}^{n^l_s} S^l_p(i)}{\left(\frac{R_v L_G}{f_s}\right)} \tag{3.12}$$

where $D_p^l(i)$ is the distortion caused due to the loss of slice i in frame l.

3.7 Frame-Level DP-UEP using Prediction

The **DP-UEP**(frame) scheme in the previous section has the following two major issues for live streaming applications: (i) measuring CMSE values of the slices of a frame requires the decoding of current and other frames of the GOP which is computationally intensive and introduces about one GOP time delay, and (ii) determining the channel bit budget for different frames in each GOP in real-time. In this section, we introduce an improved frame-level scheme, denoted as **DP-UEP**(predict), to address these issues.

CMSE Prediction: For the first issue, we use a slice CMSE prediction scheme proposed in [50], which predicts the CMSE corresponding to individual slice losses of a frame in real-time. This scheme uses a combination of video parameters which can be easily extracted during the encoding of a frame without requiring information from future frames.

$\hat{w}_{ovh}^l$ Prediction: To address the second issue we train a GLM to predict the w_{ovh}^l of every frame l, denoted as $\hat{w}_{ovh}^l$, in real-time. The GLM to estimate $\hat{w}_{ovh}^l$ is developed over a database of the factors discussed in Section 3.6 and derived for videos with different types of motion and content. We use a database of 12 CIF video sequences that span (a) low motion: *Silent*, *Mother-Daughter*, *Bridge*, and *Akiyo*; (b) medium motion: *Table Tennis*, *Coastguard*, *Tempete*, and *Foreman*; and (c) high motion: *Soccer*, *Bus*, *Football*, and *Stefan*. We use the first three sequences from each motion category for training and the last one from each category for testing. For a given R_v, we compute the factors w_{ovh}^l, w_{cmse}^l, and w_c^l for the frames of each training video sequence by using the **DP-UEP**(GOP) scheme and store them in the database along with the channel SNR. The GLM, explained later in Section 3.7.1, is trained offline only once. $\hat{w}_{ovh}^l$ is then used to estimate the channel bit budget constraint (as shown in Equation 3.11) and estimate the optimal packet sizes and code rates for the slices of frame l.

3.7.1 Generalized Linear Modeling Approach for Estimating w^l_{ovh}

GLMs are an extension of classical linear models [77, 78]. We train the GLM to predict w^l_{ovh} (i.e., $\hat{w}^l_{ovh}$). Let $\mathbf{Y} = [y_1, y_2, y_3, ..., y_N]$ be a vector of our response variable w^l_{ovh} from the database. Every data point y_i in $\mathbf{Y}$ is expressed as a linear combination of a known covariate vector $[1, x_{i1}, x_{i2}, x_{i3}, ..., x_{ip}]$, where p is the number of factors, and a vector of unknown regression coefficients $\boldsymbol{\beta} = [\gamma, \beta_1, \beta_2, ..., \beta_p]^T$. The covariate vector is a row of matrix $\mathbf{X}$ of order $N \times (p+1)$ with elements x_{ij} for N observations and p factors also from the database.

$$f(\mathbf{Y}) = \mathbf{X}\boldsymbol{\beta} \quad ; \quad f(y_i) = \gamma + \sum_{j=1}^{p} x_{ij}\beta_j. \tag{3.13}$$

where $f(.)$ is called the link function. After estimating $\boldsymbol{\beta}$, we use it to derive the predicted response variable vector $\widehat{\mathbf{Y}} = [\hat{y}_1, \hat{y}_2, \hat{y}_3, ..., \hat{y}_N]$ computed as $f^{-1}(\mathbf{X}\boldsymbol{\beta})$; f^{-1} is the inverse of the link function and $\widehat{\mathbf{Y}}$ is a vector of $\hat{w}^l_{ovh}$.

Response Variable Distribution

To determine the link function for the GLM, we need to know the distribution family of our response variable. We evaluate the goodness of fit for ranking Weibull, Gamma, and Gaussian fitted distributions of w^l_{ovh} by using three information criteria (IC): (a) SIC: Schwarz information criterion, aka Bayesian information criterion [79], (b) AIC: Akaike information criterion [80,81], and (c) HQIC: Hannan-Quinn information criterion [82]. Each information criterion depends on the number of distribution parameters to be estimated. For example, the Gaussian distribution has two parameters, mean and standard deviation, and the Gamma and Weibull distributions have two parameters, scale and shape parameter. Each information criterion also depends on the number of observations of our response variable w^l_{ovh}, and the maximized log-likelihood estimate of the fitted distribution producing the set of observations. For m observations and u distribution parameters, the SIC is the most strict in penalizing loss of degrees of freedom by having more distribution parameters and is computed as $u \times ln(m) - 2 \times ln(L_{max})$, where L_{max} is the maximized value of the likelihood function for the fitted distribution. HQIC holds the middle ground in its penalizing for u and is computed as $\left(\frac{m-2u+2}{m-u+1}\right) ln(ln(m)) - 2 \times u \times ln(L_{max})$. Finally, AIC is the least strict of the three in penalizing loss of degrees of freedom and is computed as $\left(\frac{m-2u+2}{m-u+1}\right) - 2 \times ln(L_{max})$.

Table 3.1: Goodness of fit statistics for maximized likelihood function based on three information criteria.

IC/Fitted Distribution	Weibull	Gamma	Gaussian
SIC	23.71	23.71	25.79
HQIC	12.72	12.72	16.88
AIC	6.25	6.25	8.33

We randomly chose $m = 5000$ observations from the vector of w^l_{ovh} values in the database, obtained from all the training videos at channel SNRs from -2 dB to 6 dB. These are divided into 100 bins from zero to one and the likelihood function is maximized for each of the three fitted distributions. The distribution parameters where the likelihood is maximized are: (a) Gaussian: mean = 0.05, standard deviation = 0.095, (b) Gamma: shape parameter = 1, scale parameter = 0.05, and (c) Weibull: shape parameter = 1, scale parameter = 0.05. Since the shape parameter of both Gamma and Weibull distributions is 1, they are in essence exponential distributions. In Table 3.1, the goodness of fit of all three information criteria are minimum for Weibull and Gamma distributions; therefore our response variable is exponential. Figure 3.7 also shows that the cumulative distributions of Weibull and Gamma are the same and closer to the cumulative distribution of the 5000 observations than the Gaussian cumulative distribution.

Model Fitting and Validation

We use the statistical software R [83] for fitting our GLM and its validation. We classified our response variable as a member of the exponential family of distributions with identity as its link function. The GLM model in R uses the AIC index [80] to determine the order in which three factors, w^l_{cmse}, w^l_c, and channel SNR are fitted. Here, the AIC index is defined as $2p - 2max(L)$, where p is the number of factors and L is the log-likelihood estimate for the model. We let $\mathbf{Y}^k$ represent the model with a

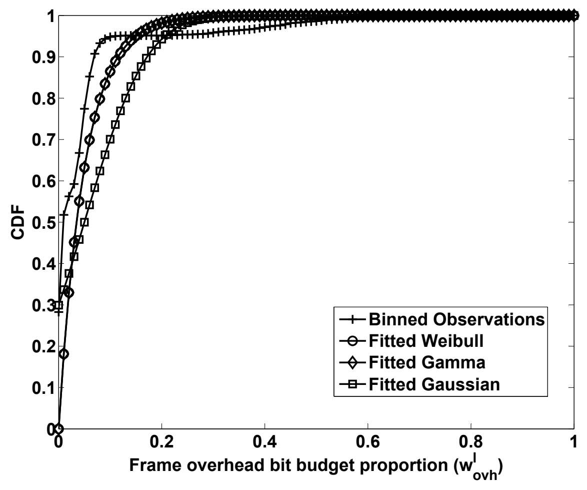


Figure 3.7: Cumulative distribution function (CDF) for the binned observations and fitted distributions of w^l_{ovh}.

subset of k factors. The i^{th} data point in $\mathbf{Y}^k$, y^k_i, where $i = 1, 2, ..., N$ is expressed as:

$$y^k_i = \gamma + \beta^k_1 x_{i1} + \beta^k_2 x_{i2} + ... + \beta^k_j x_{ij} + ... + \beta^k_k x_{ik}. \tag{3.14}$$

Here, γ is the intercept as considered in Equation 3.13, $\beta^k_j = 1, 2, ..., k$ are the fitted coefficients for k factors, and x_{ij} represents the j^{th} factor value for the i^{th} observation in $\mathbf{Y}^k$. The simplest model is the Null Model having only the intercept γ whereas the Full Model has all the p factors, i.e. $k = p$. The factors are also known as covariates. The following forward stepwise approach is used to determine the order of our covariates:

Step 1: We fit a group of p univariate models and compute their AIC values. The best univariate model has the smallest AIC value.

Step 2: We then fit $(p - 1)$ multivariate models where each model has two covariates. The first covariate is from the best univariate model in Step 1 and the second covariate is chosen from the remaining $(p-1)$ available covariates. We compute the AIC values for the $(p - 1)$ multivariate models and choose the best multivariate model with the smallest AIC value. The two covariates fitted at this stage would progress to the next step to be fitted with the third covariate.

Table 3.2: Final generalized linear model factors and coefficients including interactions.

Covariate(Factor)	Coeff. for Final Model	Model Deviance
γ	-0.0193	167.18
w_c^l	1.3240	12.2
w_{cmse}^l	5.37×10^{-5}	11.7
channel SNR	0.0028	11.7
$w_c^l\times$ channel SNR	-0.0564	9.75
$w_{cmse}^l\times$ channel SNR	9.521×10^{-5}	9.7

The covariates and coefficients of our final model are shown in Table 3.2. We also introduced two interactions, $w_c^l\times$channel SNR and $w_{cmse}^l\times$channel SNR.

The goodness of fit for a GLM can be characterized by its deviance, which is a general term of variance [77]. By definition, the deviance is zero for the Full model and positive for all other models. A smaller deviance means a better model fit. After fitting a particular model, the importance of each factor in the model can be evaluated by the resultant increase in deviance when we remove that factor from the model. The third column in Table 3.2 shows the reduction in deviance as each of the covariates in the first column is added to the model using the stepwise approach described above. Model 1 is the best univariate model with w_c^l. Model 2 has both w_c^l and w_{cmse}^l covariates. In addition to these, Model 3 has channel SNR. Model 4 adds the first interaction between w_c^l and channel SNR, and Model 5 includes all the factors in Table 3.2.

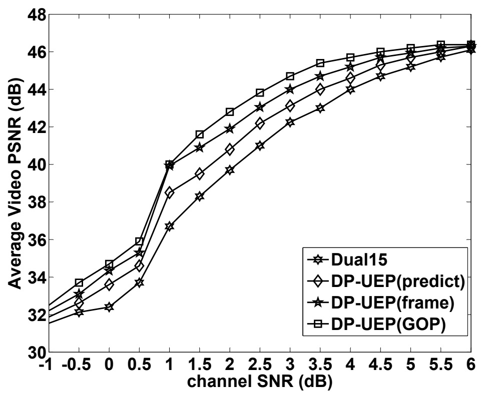
3.7.2 PSNR and VQM Performance

We evaluate the average PSNR and average VQM of our proposed **DP-UEP**(predict) scheme for the three test videos: low motion Akiyo, medium motion Foreman, and high motion Stefan. The predicted channel bit budget for frame l is evaluated as $\hat{w}_{ovh}^l \times \frac{R_{CH}L_G}{f_s}$. The proposed **DP-UEP** in Section 3.3 was used to compute the optimal packet sizes and RCPC code rates for the slices of frame l. $\hat{w}_{ovh}^l$ uses the coefficients of the factors shown in Table 3.2. Since computing the factor w_{cmse}^l for frame l is not feasible in real-time, $\hat{w}_{cmse}^l$ uses the predicted CMSE value of each slice i, $\hat{D}_p^l(i)$ in frame l, as computed in [50]. But the predicted slice CMSE values of the future frames in the GOP will not be available during real-time transmission. We therefore use the total predicted CMSE of all the slices of the previous GOP to compute the normalized predicted CMSE of the frame in the current GOP as shown in Equation 3.15.

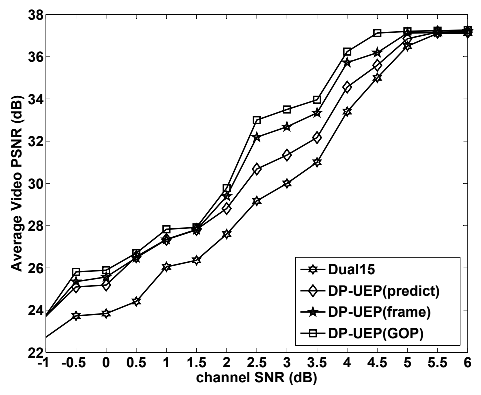
$$\hat{w}^l_{cmse} = \frac{\sum_{i=1}^{n_s^l} \hat{D}^l_p(i)}{\sum_{j=1}^{L_G} \sum_{i=1,previousGOP}^{n_s^j} \hat{D}^j_p(i)} \tag{3.15}$$

For the first GOP, the $\hat{w}^l_{cmse}$ is assumed to be zero. It is reasonable to use the predicted CMSE of the previous GOP because for most GOPs there is a high correlation between the CMSE of adjacent GOPs. On a core 2 Duo 2.6 GHz Intel processor with 4GB RAM, we observed that the average computation time across all test videos and channel SNR from -1 dB to 6 dB, is 75 ms for the IDR frame, 10.5 ms for the P frame, and 1.5 ms for the B frame. Since IDR frames have considerably more slices than P and B frames, and P frames have more slices than B frames, the computation time also varies accordingly. These computational delays are acceptable in live streaming applications.

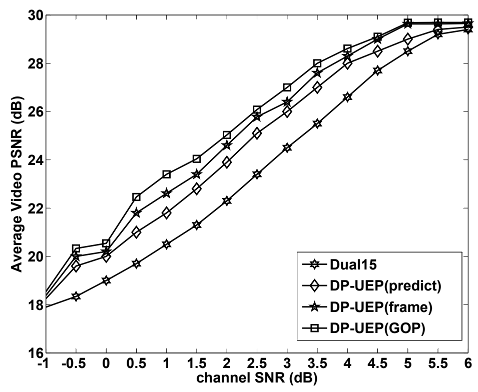
Figure 3.8 shows the performance of the **DP-UEP**(predict), **DP-UEP** (frame), **DP-UEP**(GOP), and **Dual15** schemes on the test videos, in terms of average PSNR and VQM values. The GOP structure, frame rate, and slice size are the same as considered in Section 3.5.1, and error concealment is also enabled. The videos are encoded at 720 Kbps and transmitted over a 2 Mbps AWGN channel. We observe that the error-free PSNR value decreases as the motion in the video increases.



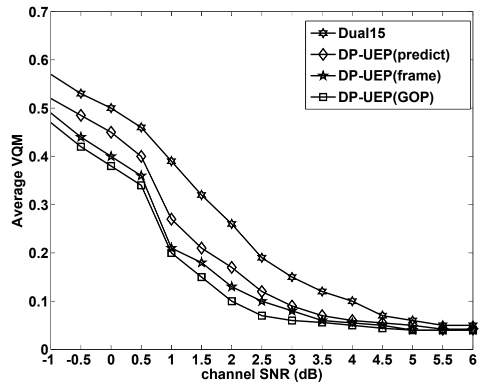
(a)



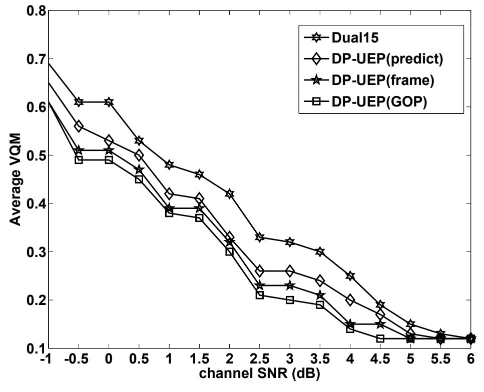
(b)



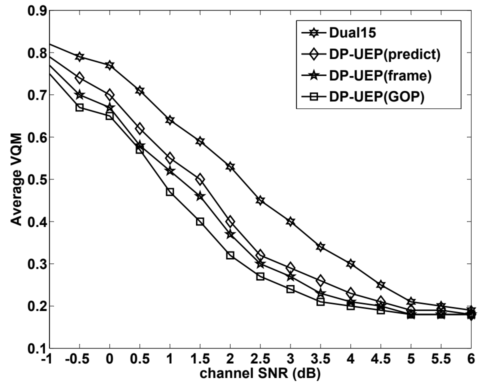
(c)



(d)



(e)



(f)

Figure 3.8: Average video PSNR (dB) and average VQM comparison computed over 100 realizations of each AWGN channel for Akiyo: (a),(d), Foreman: (b),(e) and Stefan: (c), (f). The error-free PSNR values are: 46.5 dB for Akiyo, 37.3 for Foreman, and 29.7 for Stefan.

DP-UEP(predict) has better performance than the **Dual15** scheme for all three test videos. **DP-UEP**(predict) enables real-time packet formation and transmission of videos which is not possible with the other three schemes. However, its performance is lower than **DP-UEP**(GOP) and **DP-UEP**(frame) due to the prediction of channel bit budget and slice CMSE values for each frame. For example, the PSNR gain achieved by **DP-UEP**(GOP) over **Dual15** for Foreman in Figure 3.8(b) is 3.5 dB at a channel SNR of 3 dB. For **DP-UEP**(frame) which knows the required channel bit budget, the PSNR gain drops to 2.7 dB. Predicting the channel bit budget for each frame in **DP-UEP**(predict) causes the PSNR gain to drop further to 1.4 dB. Similar behavior can also be seen in Akiyo and Stefan in Figures 3.8(a) and 3.8(c). The maximum PSNR gains achieved by **DP-UEP(predict)** over **Dual15** are 1.8 dB for Akiyo at 0.5 dB channel SNR, 2.12 dB for Foreman at 1 dB channel SNR, and 1.5 dB for Stefan at channel SNR of 2.5 dB. Similar trends are also observed in the VQM performance of the three videos shown in Figure 3.8. Further, simulations of three more test videos (whale show, Hall Monitor, and Container) from outside our database showed trends similar to those in Figure 3.8.

3.8 Acknowledgement

Chapter 3 of this dissertation contains material that appears in Kashyap K. R. Kambhatla, S. Kumar, and P. Cosman, "H.264/AVC Video Packet Aggregation and Unequal Error Protection for Noisy Channels", *IEEE International Conference on Image Processing*, Orlando, Florida, pp. 1649-1652, Sept. 30 - Oct. 3, 2012 and Kashyap K. R. Kambhatla, S. Paluri, S. Kumar, P. Cosman, "Cross-Layer Prioritized H.264 Video Packetization and Error Protection Over Noisy Channels", EURASIP Journal on Wireless Communication and Networking, in review. I was the primary author and and co-author Seethal Paluri had contributed to a part of the work. Co-authors Dr. Pamela Cosman and Dr. Sunil Kumar directed and supervised the research which forms the basis for Chapter 3.

Chapter 4

Scheduling Scalable Video Streaming over Time-Varying Wireless Links

4.1 Introduction

Video streaming applications over wireless networks are rapidly growing. However, video data has high bit rate and is delay-sensitive. As a result, video transmission is challenging over bandwidth limited and time-varying wireless channels. Additionally, unique characteristics of the video traffic, such as the temporal and spatial dependencies between different video frames and their deadline constraints, pose a challenge in supporting video quality in wireless networks. A survey of the theoretical and practical challenges in wireless video streaming can be found in [84].

To provide better video quality, various technologies have been employed such as scalable video coding [3, 85], error resilient coding [86, 87], video transcoding [88, 89], packet scheduling [90], and playout adaptation [91–94]. Scheduling algorithms employed at the transmitter play a key role in determining the performance of wireless systems. Most of the initial work on scheduling schemes focused on maximizing throughput and optimizing system performance for non-real-time and delay-tolerant traffic. For example, opportunistic schedulers for the problem of downlink scheduling were extensively studied in [95] and [96], wherein a single transmitter at the base station is shared amongst multiple downlink users. Opportunistic scheduling entails exploiting multiuser diversity inherent in wireless systems due to fluctuating channels. However, such schedulers, being oblivious to packet deadlines, video data bit rate variations, and frame dependencies, perform poorly in the context of delay-sensitive video streaming. Therefore, network-adaptive video streaming techniques proposed in [97–99] have gained significant interest. They try to overcome fluctuations due to wireless link impairments by using controls at various layers of the transmitter and/or receiver.

In a streaming media system, the client usually buffers the video data it has received in a playout buffer and begins playback after a short delay (known as the pre-roll delay) of up to several seconds [100]. Smoothing the video in this manner allows it to be transmitted in a less bursty fashion and potentially simplifies operations such as resource allocation and improves network utilization [101, 102].

Adaptive streaming techniques are generally classified as either receiver-driven or transmitter-driven [98]. A receiver-driven technique that allows the streaming media client to control the playout rate of the decoder without the involvement of the transmitter was proposed in [103]. Depending on the video and the playout buffer fullness (amount of data in the playout buffer), playout interval variation from 25% up to 50% was considered. Though this reduces the probability of playout buffer underflow and overflow, noticeable artifacts can still occur in the displayed video.

In the transmitter-driven techniques, rate-distortion (R-D) optimized packet scheduling techniques [90, 104, 105] are the state-of-the art. In every transmission opportunity, the rate is optimized for the scheduled media unit (a group of NAL units) to minimize the expected received video distortion by taking into consideration the transmission errors, retransmission delays, the decoding dependencies (frame types), and the channel bit rate constraint. It also includes selecting the media units to discard for a low channel bit rate constraint. The optimization problem is solved for an average channel by using the Lagrangian R-D formulation and is not designed to adapt and exploit the time-varying transmission rates supported by wireless links. Further, though the above schemes could show noticeable benefits by allowing adaptation to wireless link errors and retransmission delays, they require significant modifications in the streaming client and/or the streaming server [106, 107]. Our scheme focuses on solutions to schedule the video stream over a wireless link with time-varying bit rate, which requires insignificant modifications in the streaming server. At the same time, our scheduling solution provides improved video quality at the receiver by considering the relative importance of the frames and their delay bounds.

Transmission rates on the wireless links could vary significantly in every transmission time interval (TTI) due to impairments such as fading, and multi-user channel access characteristics [108, 109]. These changes in transmission rate impact the end-to-end delay of video frames. When the wireless link is slow and cannot support the video bit rate, compressed video frames fill up the post-encoding buffer eventually causing it to overflow and the frames to timeout. Meanwhile, frames are continuously played out at the client, causing the playout buffer to underflow and eventually causing an outage. Buffer underflow occurs when the number of frames in the playout buffer falls below a pre-determined threshold whereas an empty playout buffer results in an outage [53, 93, 110, 111].

Most of the existing transmitter-based scheduling schemes are based on the single layer coding of H.264/AVC [2] and propose modifications to the rate control module of the encoder. The scalable extension of H.264 enables encoding a high-quality video bit stream containing one or more subset bit streams [3]. This makes it attractive to be used in streaming applications. In this chapter, we propose a transmitter-driven scheduling algorithm which is aware of video packet importance and frame deadlines. It exploits the temporal and SNR scalabilities of a H.264/SVC compressed bit stream, and derives a subset (i.e., scalable) bit stream for transmission over a wireless link with time-varying bit rate. The subset bit stream provides graceful degradation in bad channel conditions. Our scheme uses a sliding-window based flow control at the post-encoding buffer of the streaming server. The flow control determines how many and which particular NAL units, from a window of temporal and quality layers, are to be scheduled for transmission during every TTI. The scheduled NAL units improve the received video quality for the available channel resources. The optimization problem of maximizing the expected received video quality is reduced to maximizing the product of the normalized CMSE value with the inverse of the time-to-expiry (TTE) value.

Section 4.2 discusses the related work. The system model comprising of the video streaming system, scalable video coding, and the wireless channel model is discussed in Section 4.3. Section 4.4 discusses the optimization problem formulation for the baseline and proposed scheduling approaches. The performance of the different scheduling schemes at different pre-roll delays, and under varying channel conditions is discussed in Section 4.5.

4.2 Related Work

Kang and Zakhor [112] proposed a packet scheduling algorithm for streaming an MPEG-4 compressed video over wireless channels with dedicated fixed bandwidth, fixed round trip time, and known channel bit error rate. Different deadline thresholds were assigned to video packets based on their importance. The importance of a video packet was determined by its relative position within the GOP and its motion texture context. Packets with the nearest deadline were transmitted first.

A packet selection algorithm for adaptive transmission of smoothed and layered video over a wireless channel was discussed in [113]. Before transmitting a packet from the current video layer, the scheme proposes to compute the minimum success probability of the next higher priority layer among all the remaining frames. Depending on whether this value is greater than a pre-determined heuristic threshold, the packet from the current layer could either be transmitted or discarded. This is done to maintain similar video quality among the transmitted frames. However, the complexity involved in determining the minimum success probability increases as the number of frames increases. Further, a time-varying channel makes it infeasible to compute the success probability for a large number of remaining frames.

Hung et al. [53] proposed a scheduling scheme based on an active and passive playout adaptation in the receiver buffer. The active playout tries to smooth the video playout by slowly varying its rate in order to overcome bad channel conditions. The passive playout kicks in during serious congestion and the smallest possible playout rate is employed at the receiver buffer. Playout interval variations of up to 50% are considered depending on the video content. However, the playout adaption is still limited in efficiently delivering video packets over a time-varying wireless link and in avoiding playout interruptions. Hence, a deadline-aware packet scheduling scheme is also considered at the transmitter which discards the packets of the frames which have missed their playout deadline. It also uses different numbers of retransmissions for packets belonging to different priority frames and schedules the new packets and the packets to be re-transmitted within channel bit-rate constraints. The scheme does not fully avoid playout buffer outage.

Chen et al. [54] studied an adaptive video scheduling scheme in a Markov decision process (MDP) framework at the transmitter, which requires the knowledge of instantaneous playout buffer status and channel conditions at the receiver. However, the scheduling policy is derived offline and thus is not adaptive to channels with time-varying bit rate. A state space reduction technique is proposed to limit the complexity of the MDP. The scheduling scheme works on a window of frames to be decoded at the receiver. The window size provides a tradeoff between the optimality and complexity of the scheduling scheme.

A priority-based media delivery scheme is discussed by [114] for the pre-buffering and re-buffering in the receiver playout buffer to overcome channel interruptions. The H.264/SVC bit stream is divided into three priorities. The scheduling scheme buffers more high priority data in the playout buffer. This results in pre-buffering the data for a longer playback time compared to the earliest deadline first (EDF) scheme [51]. The scheme has been proposed for both real time protocol (RTP) and hypertext transfer protocol (HTTP) based streaming.

In order to reduce the impact of network bandwidth fluctuation, an adaptive priority ordering algorithm for H.264/SVC bitstreams is proposed in [115]. It arranges the coding layers (i.e. spatial, temporal, and quality scalability) according to their R-D tradeoff within a GOP so that the transmitted video quality can be preserved over dynamic bandwidth conditions.

Stockhammer et al. [106] derived the required initial buffering delay and the receiver buffer size to avoid playout interruption due to buffer underflow or video packet loss due to buffer overflow while streaming a MPEG-4 encoded variable bit rate video. The conditions were derived for a wireless channel with known packet success probability and for pre-encoded video streams. The problem is solved in the framework of the leaky bucket algorithm in the hypothetical reference decoder or video buffering verifier at the receiver.

Recently, Chen et al. [107] described the strict conditions guiding an x264 encoder to design a bandwidth adaptive rate control for the first time. The rate control in [107] derives an upper and lower bound for the target frame size and the corresponding tightest bounds on the encoder and decoder buffer sizes subject to a strict end-to-end delay over a fast time-varying channel. The encoder then fixes the size of the frame to the average of the upper and lower bounds. The scheme depends on the accuracy of channel estimation at the transmitter. It may cause large variation in bits allocated to different frames, resulting in inconsistent video quality due to the emphasis on a strict end-to-end delay bound over a fast time-varying channel. Further, the rate control does not take into account the importance of the frame and the error propagation it may cause (due to the allocated quantization parameter value) at the receiver. To limit the variation in quality from frame-to-frame, accurate R-D models [116, 117] are required to estimate the target frame size for a targeted quality along with some R-D optimization. This has been ignored in [107] since the emphasis of choice on best rate points may cause large delay jitter.

Dua et al. [118] proposed a channel, deadline, and distortion aware scheduling scheme for streaming H.264/AVC compressed videos to multiple video clients in a wireless communication system. The scheduling problem was studied in a DP framework to minimize the aggregate distortion cost incurred over all receivers. The scheme showed significant PSNR gains over benchmark multi-user scheduling schemes such as the round robin, EDF, and best channel first schemes. Distortion for every video packet in a frame was computed as the MSE contributed by its loss. The packets of a frame were then ordered for scheduling based on their distortion. Scheduling was carried out for the packets of a single head-of-line frame of all users at a time, under the assumption that except for the first I-frame, all the other frames in the video are of equal importance. This ignores the fact that video frames contribute different levels of distortion based on their scene complexity, motion level, and type (I, P, and B).

A MDP framework in [119] was used for cross-layer optimization of scheduling at the post-encoding buffer of a video server, the packet size and scheduling at the MAC layer of the base station, and MAC receiver buffer at the client. The scheme derives a foresighted control policy (i.e., the optimal value function) and the optimal policy (set of actions) by using the value iteration algorithm over a constant bit rate BSC. Due to the large dimensionality of the problem, a strong quantization of the values was considered by the different states. The evaluation of the transition probabilities was done offline using the training video sequences. The authors resort to learning techniques, such as reinforcement learning, in order to estimate the optimal policy and also suggest updating the entries of the transition matrix online at each time instant. However, this is not realistic because the base station needs to simultaneously coordinate with the video server and the wireless video client to differentiate bad policies from good ones in real-time and eliminate them. For streaming applications, it will degrade the video quality until the learning is finished. Moreover, the reward matrix cannot consider the immediate effect of a selected set of actions on video quality and only has to use the video quality determined at the source. The framework also does not consider the frame delay constraints normally associated with scheduling in streaming applications. The foresighted control policy in [119], maximizing some long-term discounted sum of rewards linked to the video quality, achieved considerable PSNR gains compared to the short-term myopic policy in [120] which maximizes the immediate reward without paying attention to the consequence the current decision may have on future rewards.

The problem of joint adaptive media playout control at the receiver and video motion-aware packet scheduling across the APP and MAC layers at the transmitter was formulated in a MDP framework by [120]. It employed an online reinforcement learning approach with a layered real-time DP algorithm for adaptive video transmission. In addition to the parameters in [119], it also considered the modulation and coding options, provided by the PHY layer in the 802.11a standard, in the set of actions and states. It preemptively varied the playout speed of scenes, based on the motion intensity, to reduce the perceptible effect of playout speed variation. However, the high computational complexity of this scheme makes it unsuitable for real-time delay-sensitive streaming.

Li et al. [93] proposed an MDP-based joint control of packet scheduling at the transmitter and content-aware playout at the receiver, in order to maximize the quality of video streaming over wireless channels. They also proposed a content-aware adaptive playout control (i.e., slowdown) that considers the video content (i.e., motion characteristics in particular). This scheme improved the quality of the received video with only a small amount of playout slowdown which was mainly placed in low-motion scenes where its perceived effect is lower.

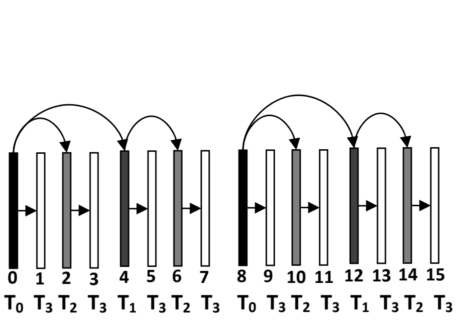
4.3 System Model

4.3.1 Scalable Video Coding

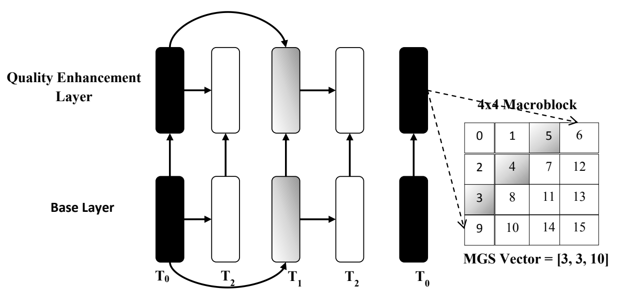
The coded video data of H.264/SVC [3] are organized into NAL units, each containing an integer number of bytes. NAL units are classified into video coding layer (VCL) NAL units, which contain coded slices or coded slice data partitions, and non-VCL NAL units, which contain associated additional information. The most important non-VCL NAL units are parameter sets and Supplemental Enhancement Information (SEI). The pre-roll delay and the playout rate are communicated by the streaming server to the client through the SEI [106, 107, 121].

We use hierarchical prediction with a structural encoding/decoding delay of zero [3] as shown in Figure 4.1(a). The temporal enhancement layers are coded as uni-directionally predicted P-pictures. The darkest colored frames belonging to temporal layer T_0 in Figure 4.1(a) are encoded as key pictures to limit the distortion propagation within a GOP. Our scalable bitstream contains $|\bar{\mathbf{T}}|$ temporal layers (where $|\bar{\mathbf{T}}|$ is the cardinality of the set $\bar{\mathbf{T}}$) with a maximum frame rate of f_r fps (e.g., 30 fps). The GOP size is then computed as $2^{(|\bar{\mathbf{T}}|-1)}$. Figure 4.1(a) has $|\bar{\mathbf{T}}| = 4, \bar{\mathbf{T}} = \{T_0, T_1, T_2, T_3\}$ and GOP size of 8.

We consider medium-grain scalability (MGS) for SNR scalability. Our scalable bitstream contains $|\bar{\mathbf{Q}}|$ quality enhancement layers. Every frame is identified with its index f which could be $0, 1, 2, \ ..., \ F$; F being the last frame index of the sequence. A NAL unit belonging to a frame f and quality enhancement layer $q \in \bar{\mathbf{Q}}$ is identified as $L_{f,q}$. The base layer (BL) NAL unit of frame f is identified as $L_{f,0}$ with $q = 0$. Figure 4.1(b) shows the motion-compensated prediction dependency between the layers for a GOP size of 4. A vertical arrow denotes a spatial prediction signal from the lower layer being used in the upper layer reconstruction. A non-vertical arrow denotes a lower temporal layer being used in the motion-compensated prediction of a higher temporal



(a)

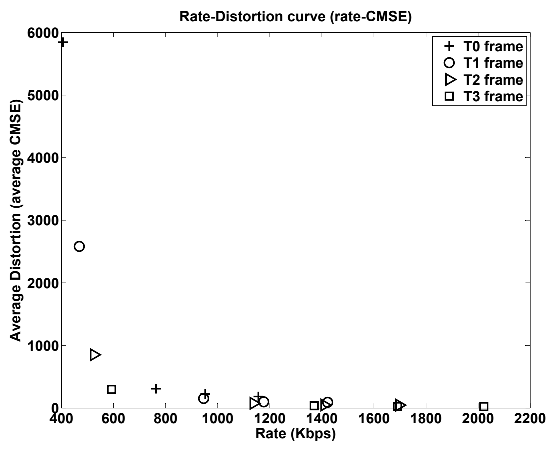


(b)

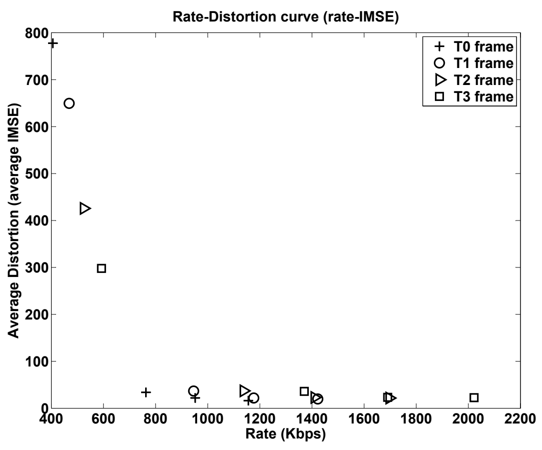
Figure 4.1: (a) Hierarchical prediction structure, and (b) motion-compensated prediction for MGS layers with key pictures.

layer. Together, they determine the error propagation path spatially and temporally. We use a MGS vector, $[3, 3, 10]$, to divide the integer transform coefficients of each 4×4 MB into three quality enhancement layers [52, 122].

Our proposed algorithm uses the CMSE to determine the importance of the VCL NAL units. CMSE values consider the error propagation due to the lost NAL units and are evaluated at the streaming server. CMSE is computed using Equation 2.1 in Chapter 2. Figure 4.2 shows the average R-D characteristic curves for a 480p (720×480) video, Table Tennis, compressed using the H.264/SVC codec JSVM 9.8 [122], and using MGS with $\bar{\mathbf{T}} = \{T_0, T_1, T_2, T_3\}$ and $\bar{\mathbf{Q}} = \{0, 1, 2, 3\}$. Every quality layer is represented by a single non-truncatable NAL unit. When the BL of a frame expires, we perform frame copy concealment in the decoder. The y-axis in Figure 4.2 shows the average distortion (CMSE or IMSE) and the x-axis shows the average bit rate up to a particular temporal layer and quality layer. For example, the four R-D points for temporal frame T_0 correspond to the BL and three quality enhancement layers with corresponding cumulative bit rates of 406, 763, 952, and 1156 Kbps. Similarly, for temporal frame T_3, the R-D points correspond to the BL and three quality enhancement layers with corresponding cumulative bit rates 593, 1371, 1691, and 2022 Kbps. Maximum video quality is achieved if all the temporal and quality layers are decoded at 2022 Kbps. Similar R-D behavior was observed for other test sequences.



(a)



(b)

Figure 4.2: Average R-D characteristic curves in terms of (a) CMSE, and (b) IMSE for different temporal and quality layers.

4.3.2 Video Streaming System

We consider a wireless video streaming system which consists of a streaming server at the transmitter, a wireless channel, and a streaming client at the receiver as shown in Figure 4.3. In streaming applications, the video server rather than the encoder decides the rate at which the frames are input into the post-encoder buffer [106, 107]. Hence, the variable bit rate scalable media stream is characterized by a frame duration Δt and a sampling curve $R_v(t)$. The sampling curve of the video sequence represents the overall amount of data (measured in bits) delivered into the post-encoder buffer by the video server up to time t. The sampling curve of the channel indicates the overall amount of video data transmitted up to time t. The sampling curve is monotonically increasing and has a staircase characteristic. Figure 4.4 shows the sampling curve for the 480p Table Tennis video considered in Section 4.3.1 at $\Delta t = \frac{1}{30}$ seconds, i.e. for video frames being delivered into the post-encoder buffer at 30 fps. The sum average bit rate for all the layers is 2022 Kbps. The non-uniform nature of the jump in the staircase pattern of the video sampling curve is attributed to its bursty nature, i.e. frames with highly fluctuating sizes arrive at a constant interval of Δt. The arrival of a frame belonging to temporal layer T_0 into the post-encoder buffer results in a steeper jump in the Table Tennis sampling curve in Figure 4.4(b). Figure 4.4(a) also illustrates two sampling curves for channels

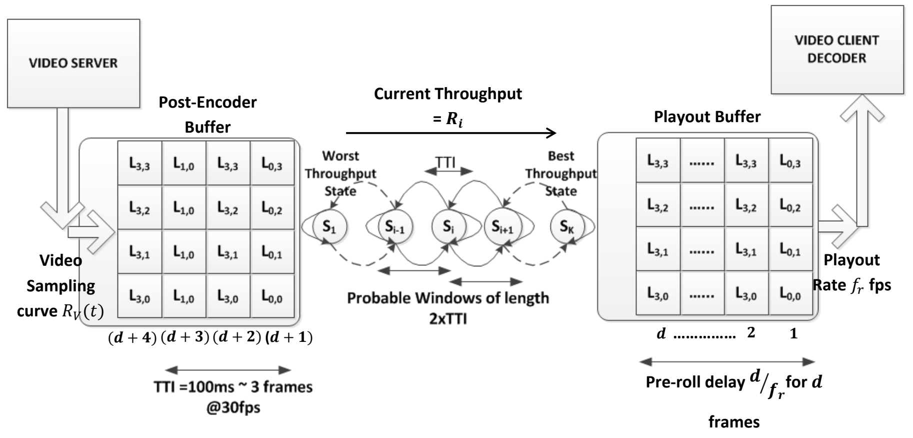
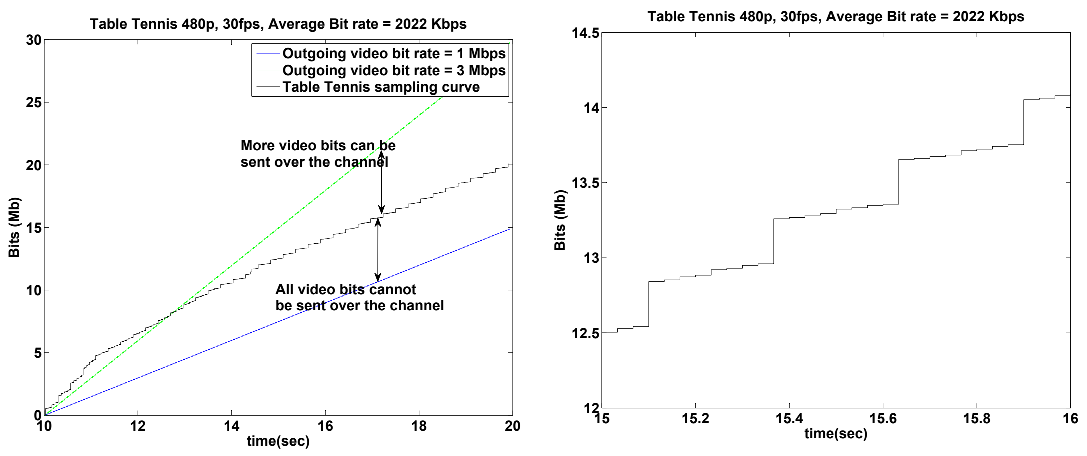


Figure 4.3: Video streaming system over a wireless link with time-varying throughput.

supporting different outgoing video bit rates of 1 and 3 Mbps.

Frames buffered in the post-encoder buffer have fixed frame deadlines. Frame deadline is the time instant at which the frame is expected by the client for decoding and depends on the pre-roll delay and playout rate allowed at the decoder [99]. All the NAL units of a frame have the same deadline. The pre-roll delay depends on the initial number of frames stored in the playout buffer and the playout rate [84,97,98,103–105]. If d frames are initially buffered at the receiver, after which it starts decoding and playing them out at a rate of f_r fps, then the resulting pre-roll delay is $\frac{d}{f_r}$ seconds. The decoder at the client starts decoding at time t. The deadline of a frame $d+i$ in the post-encoder buffer of the video server will then be $t + \frac{d+i}{f_r} = t + \frac{d}{f_r} + \frac{i}{f_r}$. This is the time at which the client's decoder fetches the frame $d+i$ for decoding. If the frame $d+i$ is not available, then the decoder conceals it using frame copy. Figure 4.5 illustrates the video streaming timing diagram for a pre-roll delay of $d = 3$ frames. It shows the times at which the video server begins to transmit the frames in the post-encoder buffer and the times at which they are completely received at the video client. For example, the video server begins transmitting the first frame, belonging to temporal layer T_0, at time t_1. The first frame is completely received by the video client at time t_2 (with $t_2 - t_1$ being the resultant delay). The video server begins the transmission of the second frame, belonging to temporal layer T_3, at time t_2. Here, we have ignored the propagation time. The pre-roll condition of $d = 3$ is satisfied



(a) (b)

Figure 4.4: (a) Sampling curve for Table Tennis $R_v(t)$, and outgoing video bits supported by the channel, (b) close up of the sampling curve between t=15 sec and t=16 sec.

when the third frame, belonging to temporal layer T_2, is received at the video client at time t_4. The receiver then starts the decoding process at time t_4. The video client expects the fourth frame to be available for decoding by $t_4 + \frac{3}{f_r} + \frac{1}{f_r}$, which is its frame deadline. The TTE value of a NAL unit is the time duration between the current time and its frame deadline. For example, the current time at which the fourth frame is scheduled in Figure 4.5 is t_4 and its TTE is equal to $t_4 + \frac{4}{f_r} - t_4 = \frac{4}{f_r}$. The TTE of a frame should at least be equal to the time required to transmit one NAL unit of the frame. If some NAL units of the frame were unable to reach the client within their TTE, then they would expire causing them to be discarded from the post-encoder buffer. The deadlines of the fifth ($t_4 + \frac{5}{f_r}$) and sixth ($t_4 + \frac{6}{f_r}$) frames are also marked on the transmission time axis of the video server and the receive time axis of the video client.

4.3.3 Wireless Channel

We are interested in capturing the time-varying nature of the wireless channel, whether it is 802.11, cellular, or home environment, where the available resources are distributed among multiple users and multiple applications. We model the wireless channel as a first-order ergodic Markov chain with K states, and $\bar{\mathbf{S}} = \{s_1, s_2, ..., s_K\}$

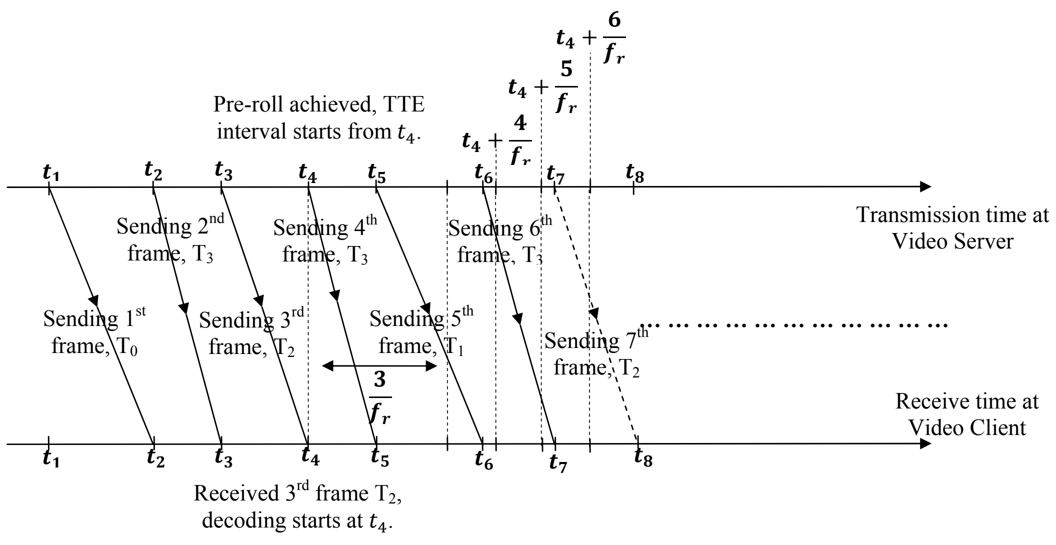


Figure 4.5: Video streaming timing diagram demonstrating frame deadlines and time-to-expiry computation.

denotes its state space [54, 114, 123]. The corresponding video bit rates supported by the states are denoted by $R_i, i \in \{1, 2, 3, ..., K\}$. The channel state supporting the lowest bit rate R_1 is s_1, and s_K is the state supporting the highest bit rate R_K. Let $p_{i,j}, i, j \in \{1, 2, 3, ..., K\}$ be the state transition probability from channel state s_i to s_j, and π_i be the steady-state probability of state s_i. We assume that transitions only happen between adjacent states, i.e. $p_{i,j} = 0$, if $|i - j| > 1$. The duration of each channel state is equal to one TTI and the constraint on the total number of video bits that can be transmitted in channel state s_i is $R_i \times TTI$. The TTI is considered to be a multiple of frame time, for example 100ms $\approx$ 3 frame time at $f_r = 30$ fps.

The estimation of the parameters of the Markov model is an important issue. Several studies [124–126] have estimated these parameters from empirical data for some typical environments. Moreover, [108, 109] elaborate on how first-order ergodic Markov chains with different numbers of states can be used to represent a fading channel.

4.4 Problem Formulation

4.4.1 EDF-based Scheme

In existing video transmission systems, packets are transmitted in the same order as they are played out at the receiver. Recent schemes [52–55, 93, 106, 114, 123] have also adopted the EDF [51] motivated scheduling of compressed scalable video for streaming applications. The EDF-based scheme transmits the BL NAL unit followed by the higher SNR layer NAL units of the frame f with the nearest deadline. The NAL unit of a frame is scheduled only if it can reach the decoder before the frame deadline and this depends on the supported outgoing video bit rate R_i. If the BL NAL unit $L_{f,0}$ expires, the whole frame f is dropped.

The limitation of the EDF-based scheme becomes evident during persistent bad channel conditions. Even when the channel supports the lowest outgoing video bit rate R_1, the EDF-based scheme continues to transmit the higher SNR layers of the unexpired frame. This can cause subsequent frames in the post-encoder buffer to be delayed and eventually expire. Though continuous frame losses are concealed using frame copy, they can severely degrade the received video quality. The EDF-based scheme does not consider the importance of different temporal layers and their contribution to distortion.

4.4.2 CMSE-based Scheme

We try to minimize the expected received video distortion under the constraints of video frame deadlines, and outgoing video bit rates supported by the channel. The CMSE distortion contributed by a NAL unit $L_{f,q}$ in frame f, belonging to a temporal layer in $\bar{\mathbf{T}}$ and spatial layer $q \in \bar{\mathbf{Q}}$, is $D_{f,q}$ and is computed using Equation 2.1 in Chapter 2. The size of the NAL unit is $B_{f,q}$ in bits. Suppose d frames were allowed to be buffered at the receiver (pre-roll of $\frac{d}{f_r}$ seconds) after which the receiver started decoding at time t'. Then at the current time t, the TTE of the NAL unit, $L_{d+i,q}$ in frame $d+i$, scheduled to be sent over a channel with state s_l is computed as

$$TTE(d+i,t) = t' + \frac{d+i}{f_r} - t \qquad (4.1)$$

If the TTE becomes less than the time required by the NAL unit to reach the decoder, then all the higher SNR layer NAL units in frame $d+i$ are also discarded along with it. At the current time t and channel state s_l, the TTE of a frame f must satisfy $t' + \frac{f}{f_r} - t \geq \frac{B_{f,q}}{R_l}$ for the transmission of its NAL unit $L_{f,q}$. Here, $\frac{B_{f,q}}{R_l}$ is the time required to transmit the NAL unit $L_{f,q}$ in the channel state s_l.

Since the video characteristics and channel rate vary over time, we propose a sliding-window flow control scheme. The algorithm determines which NAL units from a window of $w(t)$ frames should be scheduled for transmission. The window contains the BL and SNR layer NAL units belonging to unexpired frames which have to be scheduled in the current TTI. When the channel state supports a low outgoing rate then not every NAL unit in the window can be scheduled during the current TTI. Some higher quality layer NAL units which have not expired and were not scheduled in the current TTI remain in the window and get carried over to the next TTI. This increases the number of frames and NAL units to be scheduled.

The flow control optimization is carried out over the window of $w(t)$ unexpired frames during every TTI to find the set of NAL units $\bar{\mathbf{N}}$ and their scheduling order which minimizes the expected received video distortion under the constraint of the outgoing rate. The set of all NAL units $\bar{\mathbf{W}}$ which forms the search space has a size $w(t) \times |\bar{\mathbf{Q}}|$ and the size of the solution set is $|\bar{\mathbf{N}}| \leq |\bar{\mathbf{W}}|$. The search space $\bar{\mathbf{W}}$ and solution set $\bar{\mathbf{N}}$ contain 2-tuple elements (frame index, SNR layer id), for example (f, n), where f could be $0, 1, 2, \ldots, F$, and $n \in \bar{\mathbf{Q}}$. The j^{th} scheduled NAL unit in the solution set $\bar{\mathbf{N}}$ is accessed as $\bar{\mathbf{N}}_j(1)$, and $\bar{\mathbf{N}}_j(2)$. To minimize the expected received video distortion in the current TTI where the channel state is s_l, we must find and schedule the set of NAL units $\bar{\mathbf{N}}$ which maximizes the objective function formulated as

$$\begin{aligned} &\max_{\{\bar{\mathbf{N}} \in \bar{\mathbf{W}}\}} \left(\sum_{j=1}^{|\bar{\mathbf{N}}|} D_{\bar{\mathbf{N}}_j(1), \bar{\mathbf{N}}_j(2)} \right) \\ &\text{s.t.} \quad (C1): TTE(\bar{\mathbf{N}}_j(1), t) > \frac{B_{\bar{\mathbf{N}}_j(1), \bar{\mathbf{N}}_j(2)}}{R_l}, \quad \forall\, j \\ &\qquad (C2): \sum_{j=1}^{|\bar{\mathbf{N}}|} B_{\bar{\mathbf{N}}_j(1), \bar{\mathbf{N}}_j(2)} \leq (R_l \times TTI)\,. \end{aligned} \tag{4.2}$$

The above objective function assumes that a new TTI starts at the current time

t. The first constraint in Equation 4.2 ensures that only those NAL units are scheduled which can make it to the destination without expiring. The second constraint requires that all the NAL units scheduled in the current TTI must be supported by the rate R_l for the current channel state s_l. The unexpired NAL units belonging to the set $\{\bar{\mathbf{W}} - \bar{\mathbf{N}}\}$ remain in contention to be scheduled in the next TTI.

The scheduling problem in Equation 4.2 is a 0-1 knapsack problem [127, 128] in which each NAL unit is unique as an item, therefore making the number of such copies being selected either 0 or 1. For every item, which is a NAL unit $L_{\bar{\mathbf{N}}_j(1),\bar{\mathbf{N}}_j(2)}$, its distortion $D_{\bar{\mathbf{N}}_j(1),\bar{\mathbf{N}}_j(2)}$ represents the item value and its size $B_{\bar{\mathbf{N}}_j(1),\bar{\mathbf{N}}_j(2)}$ represents the item weight. The maximum weight supported by the channel is $R_l \times TTI$, which represents the number of bits that can be scheduled during that TTI. Each item also has an additional parameter in terms of its TTE value which must satisfy a lower bound (i.e., constraint ($C1$) in Equation 4.2) in order to be in contention to be selected. It is not feasible to solve the formulation in Equation 4.2 directly by exhaustive search [127, 128].

Solution using Dynamic Programming

We solve the optimization problem in Equation 4.2 using a DP approach which runs in polynomial time (in the number of NAL units scheduled and transmitted). In each iteration, we select one of the unexpired NAL units from the window of $w(t)$ frames to be scheduled such that the cumulative sum of the CMSE values of the scheduled NAL units is maximum. Basically, the unexpired NAL units which are contending to be scheduled are ranked based on their CMSE contribution and the one with the highest rank is transmitted in each iteration. Further, when more than one NAL unit have the same CMSE, they are ranked depending on the temporal and SNR layers to which they belong. This scheme gives a higher priority to the NAL units belonging to the lower temporal and SNR layers in the window. These NAL units are generally larger in size and usually contribute high CMSE distortion due to error propagation.

Note that the NAL unit selected in each iteration is a unique solution due to the implicit constraint that the higher SNR layers of a frame cannot contend for selection if its lower layer has not yet been scheduled. Therefore, there will be a maximum of $w(t)$ NAL units contending in each iteration out of which one NAL unit is selected. Suppose θ NAL units from index $j = k - \theta$ to $j = k - 1$ have already been scheduled from the search space $\bar{\mathbf{W}}$ in the current TTI (i.e., the size of the current solution set $|\bar{\mathbf{N}}|$ is θ). Further, say the NAL units contending for the current scheduling spot (index $j = k$) belong to a subset $\tilde{\mathbf{W}} \in \bar{\mathbf{W}}$, whose size is $w(t)$. Then the k^{th} NAL unit is selected recursively as,

$$\max_{\left\{\bar{\mathbf{N}} \in \bar{\mathbf{W}}\right\}} \left(\sum_{j=k-\theta}^{k} D_{\bar{\mathbf{N}}_j(1), \bar{\mathbf{N}}_j(2)} \right) = \sum_{j=k-\theta}^{k-1} D_{\bar{\mathbf{N}}_j(1), \bar{\mathbf{N}}_j(2)} + \arg\max \left\{ D_{\tilde{\mathbf{W}}_1(1), \tilde{\mathbf{W}}_1(2)}, \; \ldots, \; D_{\tilde{\mathbf{W}}_{|\tilde{\mathbf{W}}|}(1), \tilde{\mathbf{W}}_{|\tilde{\mathbf{W}}|}(2)} \right\} \tag{4.3}$$

Equation 4.3 implies that the next step of the optimization process is independent of its past steps, thus forming the foundations of the DP solution. The computational complexity is greatly decreased to $\mathbf{O}(|\bar{\mathbf{N}}|)$, depending only on the total number of NAL units scheduled in the TTI. The NAL unit selected in every recursion of Equation 4.3 is immediately transmitted. This is a significant improvement over the exponential computational complexity of the exhaustive search algorithm.

4.4.3 Proposed Scheme

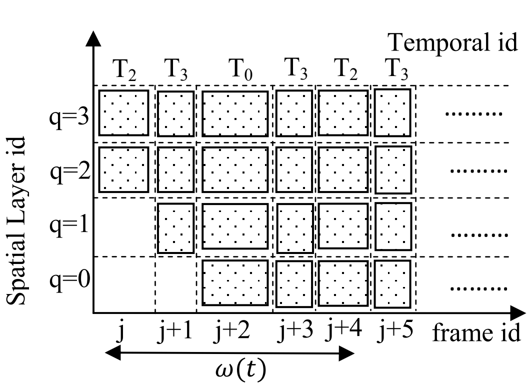
The above CMSE-based scheme does not consider the size (in bits) and the TTE values to rank the contending NAL units. Many NAL units with a large CMSE value also have a large size. Scheduling such a NAL unit may cause more delay to the transmission of subsequent NAL units in the window. We propose a scheduling scheme which considers the importance of NAL units in terms of (a) the CMSE distortion contributed to the received video quality, (b) the size of the NAL unit in bits, and (c) the TTE of the NAL unit in seconds.

We define a new parameter $\mathcal{V}_{L_{\tilde{\mathbf{W}}_j(1), \tilde{\mathbf{W}}_j(2)}}(t)$ to rank every contending NAL unit $L_{\tilde{\mathbf{W}}_j(1), \tilde{\mathbf{W}}_j(2)}$ in the window $\tilde{\mathbf{W}}$ by combining these three parameters. At current time t, the TTE of $L_{\tilde{\mathbf{W}}_j(1), \tilde{\mathbf{W}}_j(2)}$ will be $TTE(\tilde{\mathbf{W}}_j(1), t)$ and it must satisfy $TTE(\tilde{\mathbf{W}}_j(1), t) > \frac{B_{\tilde{\mathbf{W}}_j(1), \tilde{\mathbf{W}}_j(2)}}{R_l}$. $\mathcal{V}_{L_{\tilde{\mathbf{W}}_j(1), \tilde{\mathbf{W}}_j(2)}}(t)$ is computed as

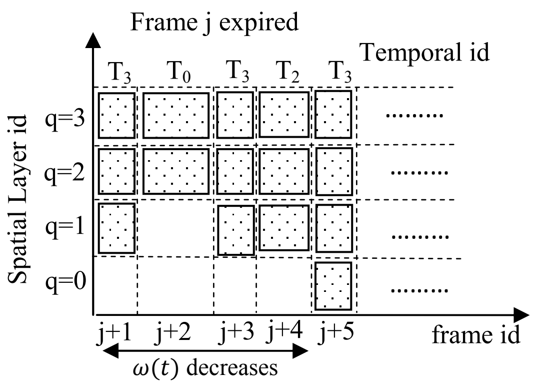
$$\mathcal{V}_{L_{\tilde{\mathbf{W}}_j(1),\tilde{\mathbf{W}}_j(2)}}(t) = \frac{D_{\tilde{\mathbf{W}}_j(1),\tilde{\mathbf{W}}_j(2)}}{B_{\tilde{\mathbf{W}}_j(1),\tilde{\mathbf{W}}_j(2)} \times \left(TTE(\tilde{\mathbf{W}}_j(1),t) - (\frac{B_{\tilde{\mathbf{W}}_j(1),\tilde{\mathbf{W}}_j(2)}}{R_l})\right)} \tag{4.4}$$

In $\mathcal{V}_{L_{\tilde{\mathbf{W}}_j(1),\tilde{\mathbf{W}}_j(2)}}(t)$, the CMSE of the NAL unit divided by its size is its normalized CMSE value while its TTE is updated continuously as time t progresses. During every iteration of the DP solution, we simply transmit the NAL unit with the maximum $\mathcal{V}_{L_{\tilde{\mathbf{W}}_j(1),\tilde{\mathbf{W}}_j(2)}}(t)$ instead of transmitting the NAL unit with the maximum CMSE (shown in Equation 4.3).

Figure 4.6 illustrates a sample of the iterations of our proposed DP solution in a TTI. In Figure 4.6(a), frames j to $j+4$ constitute the window of frames $w(t)$ which are considered for transmission during the current TTI. The frame TTE value increases from frame j to frame $j+5$. The empty spaces in frames j and $j+1$ in Figure 4.6(a) indicate the NAL units that were transmitted in the previous TTI. The leftover NAL units in frames j and $j+1$ have been carried over to the current TTI. The new frames in the current window are $j+2$, $j+3$, and $j+4$ and the window size is $w(t) = 5$ frames. Figure 4.6(b) shows the window after four iterations. The additional empty spaces in Figure 4.6(b) indicate the NAL units that have already been transmitted in the current TTI. Figure 4.6(c) shows the iterations corresponding to the NAL units transmitted in the current TTI. In each iteration, the lowest available SNR layer NAL unit of each frame in the window contend with one another for a scheduling spot. Among the contending NAL units, the one contributing the maximum parameter value (Equation 4.3) is chosen for transmission. For example, the BL ($q = 0$) NAL unit of the frame $j+2$, $L_{j+2,0}$, gets selected for transmission in iteration 1. In iteration 2, the first SNR layer NAL unit of frame $j+2$, $L_{j+2,1}$, comes into contention for a scheduling spot. However, the BL NAL unit of frame $j+4$ gets transmitted in iteration 2, and the first SNR layer NAL unit of frame $j+2$ is transmitted in iteration 3. During this period, frame j expired. The window size then decreases to only 4 frames, i.e. $j+1$ to $j+4$ as shown in Figure 4.6(b). In iteration 4, only four NAL units now contend against each other for a scheduling spot and the BL of frame $j+3$ is scheduled to be transmitted.



(a)



(b)

Iteration	Contending NAL Units					NAL Unit Transmitted
1	$L_{j,2}$	$L_{j+1,1}$	$L_{j+2,0}$	$L_{j+3,0}$	$L_{j+4,0}$	$L_{j+2,0}$
2	$L_{j,2}$	$L_{j+1,1}$	$L_{j+2,1}$	$L_{j+3,0}$	$L_{j+4,0}$	$L_{j+4,0}$
3	$L_{j,2}$	$L_{j+1,1}$	$L_{j+2,1}$	$L_{j+3,0}$	$L_{j+4,1}$	$L_{j+2,1}$
4	$L_{j+1,1}$	$L_{j+2,2}$	$L_{j+3,0}$	$L_{j+4,1}$		$L_{j+3,0}$

(c)

Figure 4.6: Sample iterations of our proposed dynamic programming algorithm over a window of frames at the video server.

4.5 Experimental Results and Discussion

We study the effect of the scheduled NAL units on the received video quality through simulations and compare the performance of our proposed approach to (*i*) the EDF-based scheduling scheme [51], which has also been used recently in [52–55], and (*ii*) the CMSE-based scheme where the NAL units in the sliding-window are scheduled based only on their CMSE contribution. In the past, frame importance and motion-texture have been used to schedule the frames in non-scalable video streaming [112]. Recently, [1] also used CMSE to prioritize non-scalable NAL units within a GOP and schedule them in the decreasing order of priority. The CMSE-based scheme on scalable video is similar to [1,112]. Our proposed algorithm trades off the importance of the NAL units with their deadlines and determines the appropriate transmission order for the NAL units in the sliding-window. It significantly reduces whole frame losses and improves received video quality.

4.5.1 Simulation Setup

This section evaluates the performance of the EDF-based, CMSE-based, and our proposed scheduling schemes. Two 480p (720×480) resolution video sequences, Table Tennis and Stefan, are used in our experiments. They are encoded using H.264/SVC JSVM 9.8 reference software [122] at a frame rate of 30 fps, for a GOP length of 8 frames, using hierarchical prediction with a structural encoding/decoding delay of zero as shown in Figure 4.1. A GOP size of 8 gives four temporal layers, $\bar{\mathbf{T}} = \{T_0, T_1, T_2, T_3\}$. MGS is enabled to achieve a fine level of SNR quality and the integer transform coefficients of every 4×4 transform block are split into three additional layers by using the MGS vector $[3, 3, 10]$ suggested in [52,122]. Hence, we get four SNR quality layers $\bar{\mathbf{Q}} = \{0, 1, 2, 3\}$. Decoding all the temporal and quality layers in Table Tennis and Stefan results in PSNR values of 35.2 dB and 34.8 dB, respectively. Tables 4.1(a) and 4.1(b) show the cumulative bit rates of the sub-streams in Table Tennis and Stefan. For example, the bit rate of the BL in temporal layer T_1 in Table Tennis is 468 Kbps, and it includes the BL of temporal layer T_0 from which it is temporally predicted. Similarly, the bit rate for the first quality enhancement layer of temporal layer T_2 in Table Tennis is 1138 Kbps which includes its own BL

Table 4.1: Bit rates (Kbps) of sub-streams of 720×480 resolution (a) Table Tennis and (b) Stefan.

(a)

$(q \in \bar{\mathbf{Q}} \quad t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	406	468	525	593
$q = 1$	763	946	1138	1371
$q = 2$	952	1177	1410	1691
$q = 3$	1156	1422	1697	2022

(b)

$(q \in \bar{\mathbf{Q}} \quad t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	506	697	901	1069
$q = 1$	893	1324	1833	2354
$q = 2$	1081	1594	2185	2785
$q = 3$	1199	1743	2360	2995

as well as the BL and first quality enhancement layers of temporal layers T_0 and T_1. The video playout rate at the receiver is fixed at 30 fps. Four different pre-roll delay values of 0.1, 0.2, 0.3, and 0.4 seconds are considered corresponding to 3, 6, 9, and 12 frames allowed to be initially buffered at the receiver before starting decoding. Each temporal layer has four NAL units corresponding to the four quality enhancement layers. The NAL unit sizes vary depending on the temporal and quality layers and the video content. Generally, the NAL unit size decreases from temporal layer T_0 to T_1, T_2, and T_3. Tables 4.2 and 4.3 show the average NAL unit sizes and average CMSE values.

The wireless channel is modeled as an ergodic Markov chain with three states good, medium, and bad. The state transition probability matrix $\bar{\mathbf{P}} = [p_{i,j}] = \begin{bmatrix} 5/6 & 1/6 & 0 \\ 1/6 & 1/2 & 2/6 \\ 0 & 1/3 & 2/3 \end{bmatrix}$ with $i, j \in \{1, 2, 3\}$, s_1 being the bad channel state, and s_3 being the good channel state [52, 54, 114, 123]. We assume that transitions only happen between adjacent states, i.e., $p_{i,j} = 0$, if $|i - j| > 1$. The state probability vector $\bar{\mathbf{p}}[l] = [p_{s_1}[l] \ p_{s_2}[l] \ p_{s_3}[l]]^T$ at index l is computed using the recursive Chapman-Kolmogorov equation as $\bar{\mathbf{p}}^T[l] = \bar{\mathbf{p}}^T[l-1] \times \bar{\mathbf{P}}$. The steady-state vector $\bar{\pi} = \lim_{l \to +\infty} \bar{\mathbf{p}}[l]$ is computed by solving the system of equations $\bar{\pi}^T = \bar{\pi}^T \times \bar{\mathbf{P}}$ [129]. The steady-state probabilities of the three channel states are all equal to $\frac{1}{3}$.

The frames are read into the post-encoder buffer at 30 fps. The TTI value of the channel is set to 100 ms which is equal to a window of approximately 3 frames. The supported outgoing video bit rates, R_i, corresponding to the good, medium, and bad channel states for Stefan are 3000, 2100, and 1200 Kbps, and for Table Tennis are 2025, 1400, and 800 Kbps. Monte-Carlo simulations were performed for 120 random

Table 4.2: (a) Average NAL unit sizes (bytes) and (b) average CMSE values of Table Tennis.

(a)

$(q \in \bar{\mathbf{Q}} \;\; t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	13522	1942	892	568
$q = 1$	11907	3933	2221	1374
$q = 2$	6302	1336	673	405
$q = 3$	6791	1328	676	368

(b)

$(q \in \bar{\mathbf{Q}} \;\; t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	5845	2582	851	298
$q = 1$	308	152	75	36
$q = 2$	230	98	48	23
$q = 3$	190	91	46	23

Table 4.3: (a) Average NAL unit sizes (bytes) and (b) average CMSE values of Stefan.

(a)

$(q \in \mathbf{Q} \;\; t \in \mathbf{T})$	T_0	T_1	T_2	T_3
$q = 0$	16882	6222	3354	1401
$q = 1$	12903	7955	5027	2941
$q = 2$	6265	2716	1342	659
$q = 3$	3942	967	422	299

(b)

$(q \in \bar{\mathbf{Q}} \;\; t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	10591	5626	2585	1022
$q = 1$	288	175	95	44
$q = 2$	211	106	52	26
$q = 3$	177	92	48	25

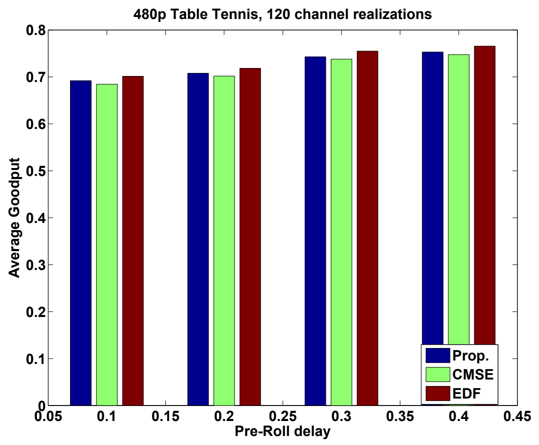
channel realizations. Each channel realization contains multiple channel states of TTI duration. To verify that 120 random channel realizations are a sufficient nmber, we generated two additional sets of 120 realizations each and verified that the average output results were within 0.0005%. The EDF-based, CMSE-based, and our proposed scheduling schemes are depicted in the figures as 'EDF', 'CMSE', and 'Prop.'.

4.5.2 Evaluation of Average Goodput and Percentage of Expired Whole Frames

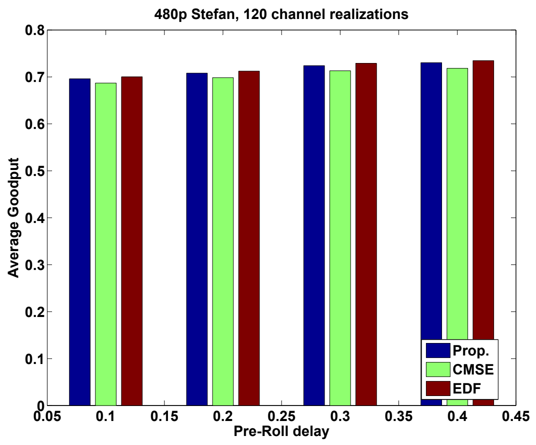
We first compute the goodput for all the scheduling schemes as

$$\text{Goodput} = \frac{\text{Total video bits received}}{\text{Total video bits in sequence}} \tag{4.5}$$

Figures 4.7(a) and 4.7(b) show the average goodput evaluated over 120 different channel realizations for Table Tennis and Stefan. The average goodput values for the EDF-based, CMSE-based, and proposed schemes differ only within 1.2%. Also the



(a)



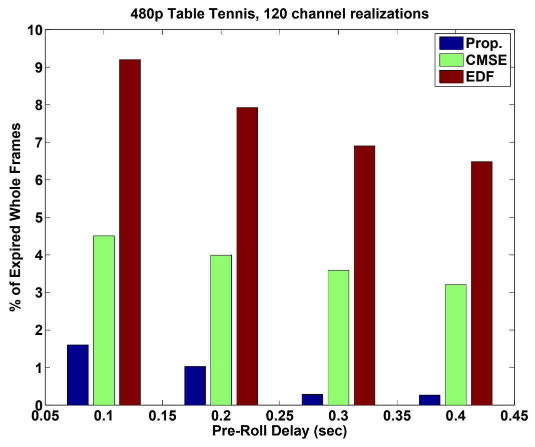
(b)

Figure 4.7: Average goodput of the EDF-based, CMSE-based, and proposed scheduling schemes.

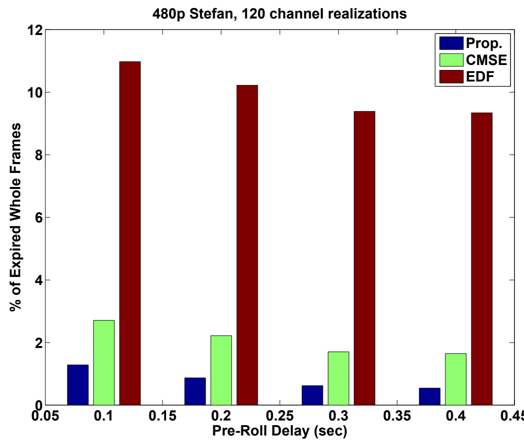
average goodput increases with pre-roll delay because more frames are allowed to be buffered at the receiver which increases the frame deadlines and TTE values of the NAL units in the post-encoder buffer.

A frame is completely lost if its BL NAL unit expires. Figures 4.8(a) and 4.8(b) show the percentage of expired whole frames averaged over 120 channel realizations, for Table Tennis and Stefan in the EDF-based, CMSE-based, and proposed schemes at different pre-roll delays. The expired whole frames are discarded from the post-encoder buffer and concealed at the decoder by using frame copy. The CMSE-based scheme sends the most important NAL units belonging to the lower temporal and SNR layers and hence incurs a lower percentage of expired whole frames compared to the EDF-based scheme. However, it ignores the frame deadlines causing frames in higher temporal layers (e.g., T_3) to expire. The proposed scheme achieves a very low percentage of whole frame losses because it considers the TTE value, CMSE contribution, and the sizes of the NAL units in the frame window. As the pre-roll delay increases, the percentage of expired whole frames decreases in all three schemes. As discussed earlier, a higher pre-roll delay results in higher frame deadlines and NAL unit TTEs. This reduces the number of NAL units that expire, due to the increased transmission delays during bad channel conditions.

Figures 4.9(a) and 4.9(b) show the percentage of expired whole frames from

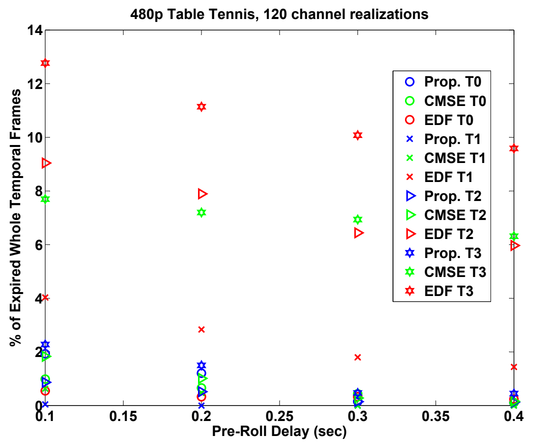


(a)

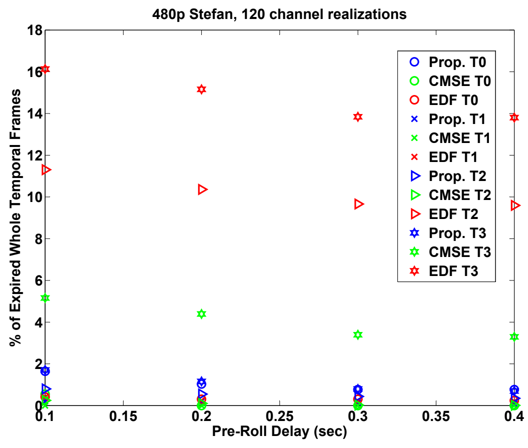


(b)

Figure 4.8: Percentage of expired whole frames in EDF-based, CMSE-based, and proposed schemes over 120 random channel realizations.



(a)



(b)

Figure 4.9: Percentage of expired whole frames in different temporal layers of EDF-based, CMSE-based, and proposed schemes over 120 random channel realizations.

Table 4.4: Average normalized CMSE values of 720×480 resolution (a) Table Tennis and (b) Stefan.

(a)

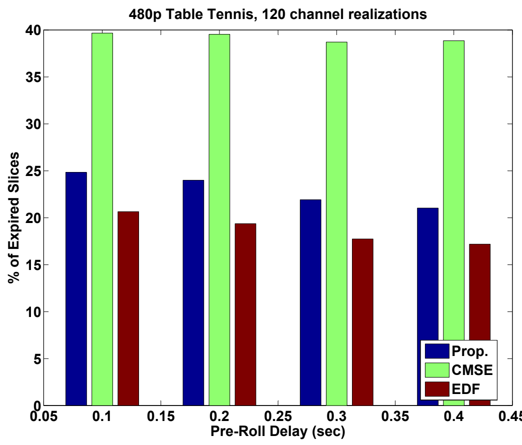
$(q \in \bar{\mathbf{Q}} \quad t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	0.43	1.33	0.95	0.5
$q = 1$	0.03	0.04	0.03	0.03
$q = 2$	0.04	0.07	0.07	0.06
$q = 3$	0.03	0.07	0.07	0.06

(b)

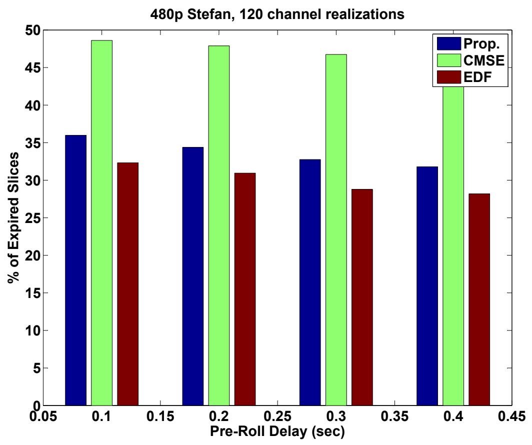
$(q \in \bar{\mathbf{Q}} \quad t \in \bar{\mathbf{T}})$	T_0	T_1	T_2	T_3
$q = 0$	0.63	0.91	0.77	0.73
$q = 1$	0.02	0.02	0.02	0.02
$q = 2$	0.03	0.04	0.04	0.04
$q = 3$	0.05	0.1	0.1	0.1

different temporal layers in Table Tennis and Stefan, computed as a ratio of the number of frames in a temporal layer whose BL NAL unit has expired to the total number of frames in that layer, averaged over 120 random channel realizations. The EDF-based scheme discards a significantly higher percentage of frames belonging to the higher temporal layers T_2 and T_3 as compared to the CMSE-based and proposed schemes. Since the EDF-based scheme considers only the TTE values of the NAL units during scheduling, transmission of the significantly larger frames belonging to T_0 and T_1 cause the smaller sized frames belonging to T_2 and T_3 to expire. The CMSE-based scheme ignores the frame deadlines and only considers the CMSE values of the NAL units. From Tables 4.2 and 4.3, we observe that this scheme transmits the larger BL NAL units of lower temporal layers in the window causing more NAL units belonging to higher temporal and SNR layers to expire. Though the proposed scheme considerably reduces the total number of expired whole frames, it incurs a slightly higher percentage of expired frames from T_0 as compared to the CMSE-based and EDF-based schemes. Though the CMSE distortion contributed by a NAL unit in T_0 is large, sometimes its size is also large causing its normalized CMSE to become smaller than other contending NAL units. This causes it to lose out to NAL units from higher temporal layers while contending for a scheduling spot. Table 4.4 shows the average normalized CMSE values for Table Tennis and Stefan derived from Tables 4.2 and 4.3, respectively.

Next, we look at how the expired NAL units are distributed among the different temporal and SNR layers.



(a)



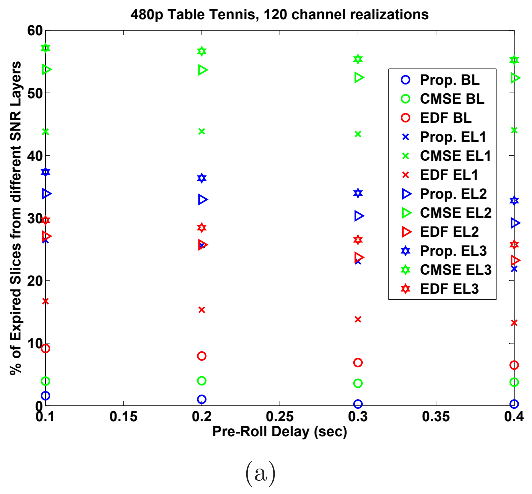
(b)

Figure 4.10: Total percentage of expired NAL units in 120 random channel realizations for EDF-based, CMSE-based, and proposed schemes.

4.5.3 Evaluation of Expired NAL Units

Figures 4.10(a) and 4.10(b) illustrate the percentage of NAL units expired in Table Tennis and Stefan, averaged over 120 random channel realizations. The percentage of expired NAL units decreases with increasing pre-roll delay in the three schemes. At every pre-roll delay, more NAL units are discarded in the proposed scheme than in the EDF-based scheme, for both the sequences. The CMSE-based scheme has the highest percentage of expired NAL units among the three schemes. However, the goodput is almost the same for the three schemes as shown in Figures 4.7(a) and 4.7(b). In fact, more higher SNR layer NAL units expire in CMSE-based and our proposed schemes, which is discussed in the next paragraph. As shown in Tables 4.2(a) and 4.3(a), these higher SNR layer NAL units are much smaller in size than the BL NAL units. Since both the CMSE-based scheme and our proposed scheme schedule the larger BL NAL units from the frame window more often, the NAL units belonging to higher SNR layers expire.

Figures 4.11(a) and 4.11(b) show the percentage of expired NAL units belonging to different SNR layers in Table Tennis and Stefan. Here, the second, third, and fourth quality enhancement layers are denoted as EL1, EL2, and EL3. Our proposed scheme has significantly reduced the percentage of expired BL NAL units and hence, also significantly reduced the distortion caused by complete frame loss as compared to



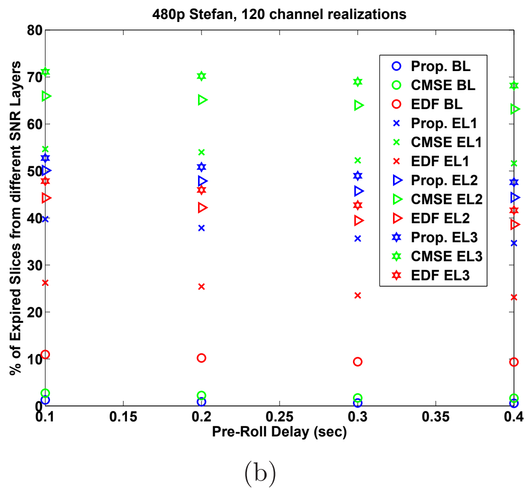
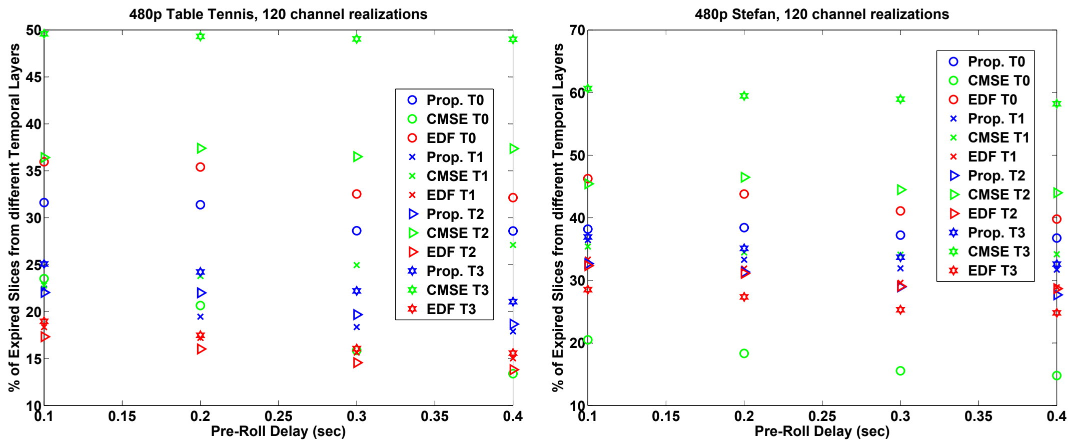


Figure 4.11: Percentage of expired NAL units in different SNR quality layers from 120 random channel realizations of EDF-based, CMSE-based, and proposed schemes.

the EDF-based scheme. However, this is achieved at the expense of more smaller-sized NAL units belonging to the higher quality enhancement layers. In the CMSE-based scheme, more NAL units in EL1, EL2, and EL3 expire than in our proposed scheme. This is because during every TTI the smaller-sized NAL units of higher SNR layers in the window fall behind in the scheduling order. For example, at a pre-roll delay of 0.2 seconds, almost 58% of NAL units in T_3 expire in the CMSE-based scheme compared to 38% in our proposed scheme and 29% in the EDF-based scheme. The discarded NAL units belonging to the higher SNR layers EL1, EL2, and EL3 also include the events where they were discarded because the BL of that frame had expired. For example, at a pre-roll delay of 0.2 seconds for Table Tennis in Figure 4.11(a), 1% of the EL3 NAL units were discarded in our proposed scheme because the BL NAL units expired, an additional 25% were discarded when EL1 NAL units expired, and additional 8% were discarded when EL2 NAL units expired. Finally, only 3% had actually contended for a scheduling spot but failed.

Figures 4.12(a) and 4.12(b) show the percentage of expired NAL units from different temporal layers averaged over 120 random channel realizations. We observe that a greater percentage of NAL units expire from T_0 in both the EDF-based and proposed schemes compared to the CMSE-based scheme. However, as shown in Figures 4.9(a) and 4.9(b) very few whole frames in T_0 expire in all the three schemes,



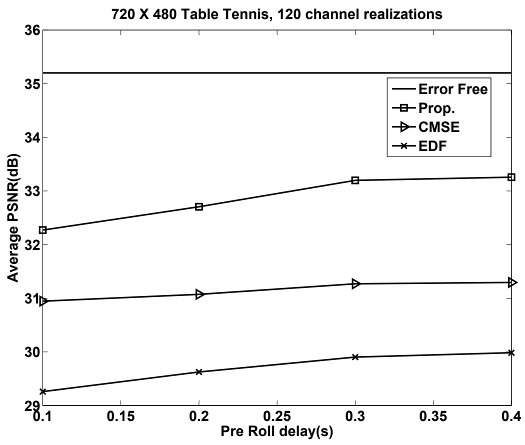
(a) (b)

Figure 4.12: Percentage of expired NAL units in different temporal layers from 120 random channel realizations of EDF-based, CMSE-based, and proposed schemes.

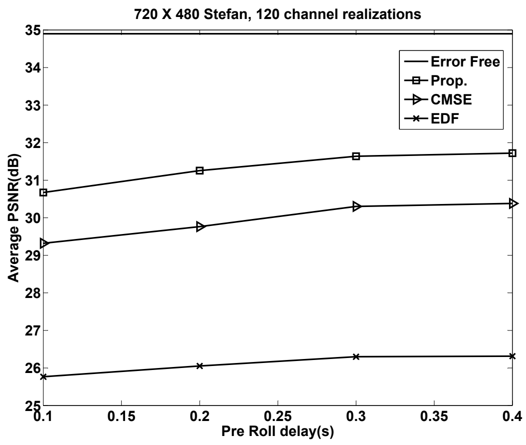
indicating that the expired NAL units belong to higher SNR layers of T_0. On the other hand, a higher percentage of NAL units expire from T_2 and T_3 temporal layers in the CMSE-based scheme. Figures 4.11(a) and 4.11(b) show that for all temporal layers, the expired slices are comprised of few BL NAL units and significantly more NAL units belonging to the higher SNR layers. Tables 4.2(a) and 4.3(a) show that NAL units in T_0 are much larger in size than the NAL units in T_1, T_2, and T_3 layers and therefore, require more time to be transmitted. Also from Table 4.4 their average normalized CMSE values are usually smaller compared to the NAL units in T_1, T_2, and T_3. Overall, our proposed scheme achieves a trade-off by discarding fewer frames from lower temporal layers and relatively more frames from higher temporal layers. Similarly, it discards fewer BL NAL units and relatively more NAL units from higher SNR layers.

4.5.4 Evaluation of Video Quality

Figures 4.13(a) and 4.13(b) show the average video PSNR for Table Tennis and Stefan, computed over 120 different channel realizations for each pre-roll delay. Our proposed scheme achieves a PSNR gain of 3.3 dB (for Table Tennis) at pre-roll delays of 0.3 and 0.4 seconds and 5.4 dB (for Stefan) at a pre-roll delay of 0.4 seconds, over the EDF-based scheme. It also achieves PSNR gains of 2 dB (for Table Tennis)



(a)



(b)

Figure 4.13: Average video PSNR of the EDF-based, CMSE-based, and proposed schemes over 120 random channel realizations.

at pre-roll delays of 0.3 and 0.4 seconds, and 1.5 dB (for Stefan) at a pre-roll delay of 0.2 seconds, over the CMSE-based scheme. The poor video quality of the EDF-based scheme is primarily attributed to the whole video frames being discarded in close proximity. To illustrate this, we plot the frame to frame performance of the EDF-based and proposed schemes in one of the 120 channel realizations.

Figures 4.14, 4.15, and 4.16 show the number of SNR quality layers received for every video frame in the proposed, EDF-based, and CMSE-based schemes for a pre-roll delay of 0.1 seconds. Figures 4.17, 4.18, and 4.19 show the same for a pre-roll delay of 0.4 seconds. If the reference frames belonging to T_0, T_1, and T_2 layers are affected by expired NAL units, the distortion propagates to other frames in the GOP. When the number of SNR layers on the y-axis is zero, it indicates that the whole frame has expired. Frames which only play out the BL, show only one SNR quality layer on the y-axis.

It is evident from the figures that for both pre-roll delays, the EDF-based scheme drops many frames in close proximity causing larger quality degradation. Also the EDF-based scheme shows large fluctuations in video quality, because some frames within the same GOP are completely discarded whereas some other frames have higher SNR layers scheduled. In our proposed scheme, for those GOPs in which complete frames are discarded, very few frames have their higher SNR layers scheduled. The

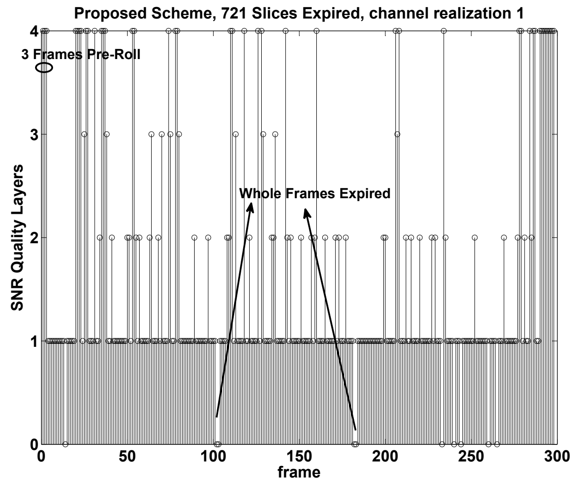


Figure 4.14: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.1s.

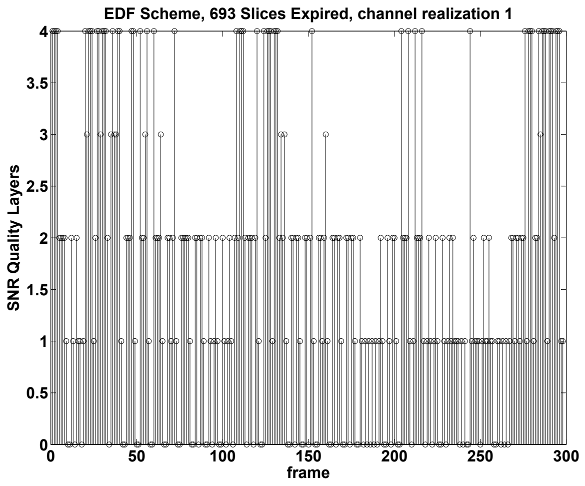


Figure 4.15: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.1s.

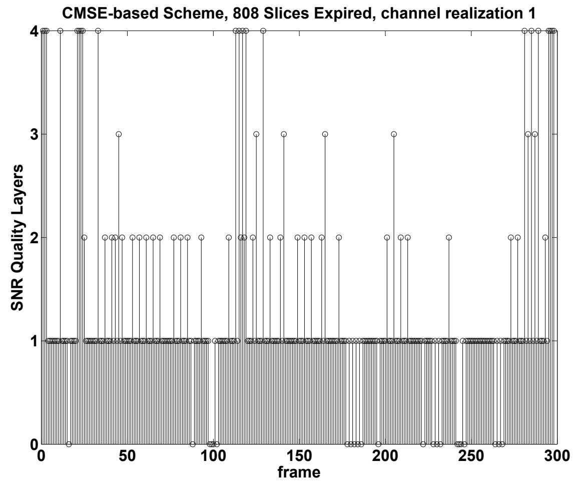


Figure 4.16: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.1s.

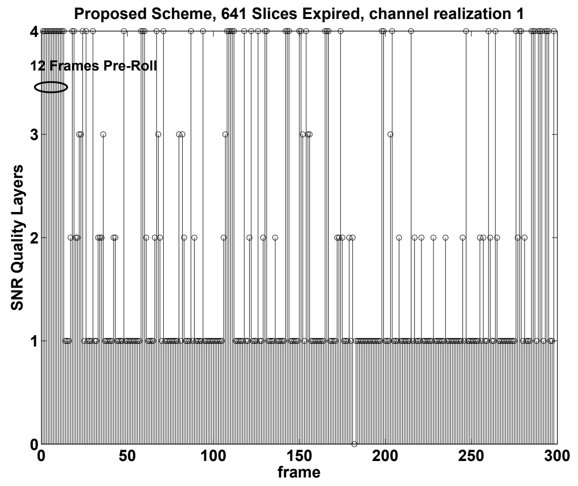


Figure 4.17: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.4s.

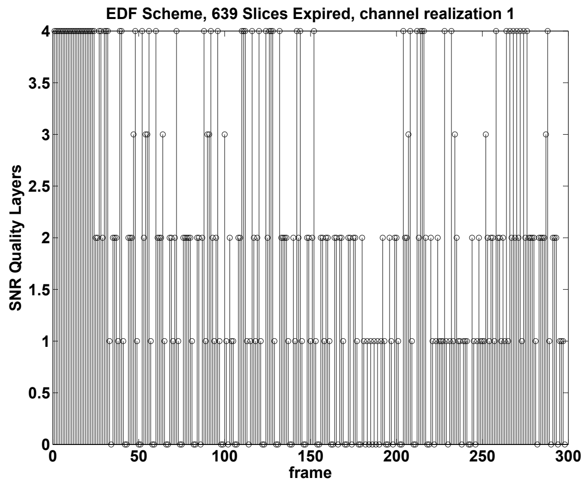


Figure 4.18: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.4s.

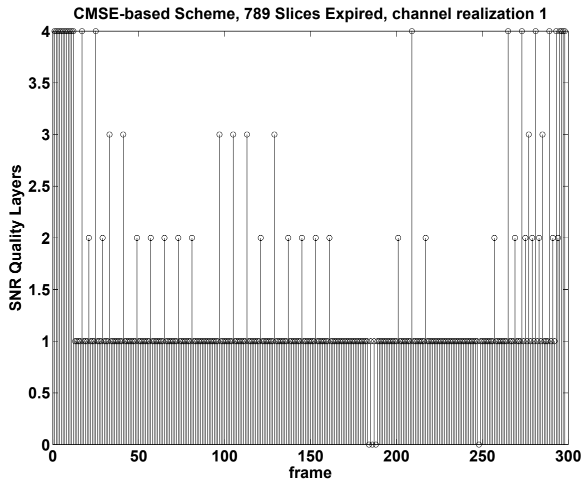


Figure 4.19: Per-frame video quality comparison between the proposed, EDF-based and CMSE-based schemes for Stefan at pre-roll delay of 0.4s.

CMSE-based scheme has less fluctuation in video quality as compared to the EDF-based scheme but it still discards some whole frames in close proximity. Finally, we also observe that as we go from a lower to higher pre-roll delay, more higher quality SNR layers are delivered for the frames.

4.6 Acknowledgement

Chapter 4 of this dissertation contains material that will appear in conference and/or journal publications. I am the primary author and the co-authors Dr. Pamela Cosman and Dr. Sunil Kumar directed and supervised the research which forms the basis for Chapter 4.

Chapter 5

Conclusion

In this dissertation, we investigated the impact on expected goodput and expected video distortion of (a) video packet prioritization and priority-aware scheduling at the APP layer, (b) packet size adaptation in terms of priority-aware MAC layer fragmentation and priority-adaptive slice aggregation at the APP layer, and finally (c) UEP at the PHY layer. The goal was to improve the received video quality in unreliable wireless networks with a cross-layer design.

In Chapter 2, a priority-aware MAC layer packet fragmentation scheme was proposed to improve the quality of pre-encoded H.264 bitstreams (measured in terms of PSNR and VQM) over error-prone wireless links. Video slices were prioritized in four classes based on their CMSE. The optimal fragment sizes for the respective priority levels were derived using the BnB technique combined with multi-dimensional arithmetic interval methods. The performance in terms of expected received video quality was compared to (a) the traditional baseline model where each packet is transmitted onto the channel at the network limited MTU size, and (b) priority-agnostic fragmentation using a single optimal fragment size. It was shown that maximizing the expected goodput or expected weighted goodput provides large gains in received video quality. The cross-layer priority information exchange between the APP and MAC layers allowed us to design a slice discard scheme which enabled us to reduce the impact of lost slices on the received video quality. The fact that these gains are achieved without error correction techniques makes it all the more interesting to evaluate the above strategies in conjunction with UEP at the PHY layer for different priority levels.

In Chapter 3, an efficient joint optimization algorithm for packet formation and optimal RCPC code rate allocation was proposed to improve the quality of H.264/AVC bitstreams transmitted over noisy channels. The proposed algorithm used a cross-layer information exchange between the PHY, MAC, and APP layers. A DP approach was used where packets were formed through slice aggregation and the optimal RCPC packet code rates were determined recursively over a GOP. The options of not coding or discarding some less important packets were exploited to reduce the expected received video distortion by increasing protection to more important packets. The proposed scheme outperformed EEP schemes as well as the Dual15 scheme in [43], providing significantly better video quality for different sequences. The DP approach was also extended to work on each frame instead of the entire GOP in order to enable live streaming with low computational complexity. The frame bit budget prediction was obtained from a GLM model developed using three factors - normalized compressed frame bit budget, normalized frame CMSE, and channel SNR over a database of videos. Our proposed DP approach showed reasonable gains in PSNR and VQM in videos spanning low, medium, and high motion. Our proposed schemes can work well with current wireless network standards such as IEEE 802.11n with MTU packet size restrictions. It would be interesting to evaluate the proposed schemes along with adaptive modulation and coding for time-varying link conditions and channel bit rates.

In Chapter 4, a sliding-window based flow control algorithm for scheduling H.264/SVC compressed video was proposed to improve the quality of streaming applications over a time-varying channel. Our scheduling algorithm considered the relative importance of the contending NAL units belonging to different temporal and SNR layers in terms of (*i*) the CMSE distortion contributed to the received video quality, (*ii*) the size of the NAL units in bits, and (*iii*) the TTE of the NAL unit in seconds. The scheduling problem of determining the appropriate order of transmission was formulated as a 0-1 knapsack problem and a DP solution was proposed which runs in polynomial time. Our proposed scheduling approach significantly reduced the number of whole frames discarded as compared to (a) a CMSE-based scheme which considers the relative importance of the NAL units only in terms of their CMSE contribution, and (b) the EDF-based scheme which minimizes the dwelling time of the NAL units in the post-encoder buffer. Our proposed scheme showed significant PSNR gains over the CMSE-based and EDF-based schemes for different video sequences and at different pre-roll delays. Our scheduling algorithm could be easily incorporated into the current RTP or HTTP streaming protocols.

5.1 Future work

There are various avenues for future work in cross-layer video packet prioritization, packet scheduling, adaptive packetization, and protection schemes. In this dissertation, we evaluated different cross-layer approaches on pre-encoded H.264 bitstreams and dropped low priority slices/packets whenever necessary to provide more protection to the higher priority packets. This assumes that the concealment techniques at the video client can efficiently reduce the error propagation and improve the perceptual video quality. One could extend the adaptive packetization and UEP to work with the R-D module in the video encoder to determine the optimal quantization parameter such that the network layers need to discard very few packets. Reinforcement learning with a focus on on-line performance can also be employed to explore the variations in channel/network conditions and derive a proper feedback that allows the video encoder to adapt the video bit rate appropriately.

Our cross-layer prioritized packet fragmentation at the MAC layer can be extended to work with UEP at the PHY layer and the performance can be evaluated for multipath fading channels. Further, Type I and Type II hybrid adaptive repeat request (HARQ) are used in wireless networks to deal with packet losses. A HARQ scheme can be designed such that video packets are re-transmitted based on their priority.

Multiple users share the resources provided by wireless networks. Video clients in the same network could have completely different channel conditions depending on their location, distance from the base station, power level, and multipath fading. Current resource allocation models are largely disassociated with the QoS requirements of the video clients in terms of their acceptable perceptual quality. Given the acceptable video quality levels, each video client could derive the PER bound depending on its channel condition and periodically communicate it to the base station. The base station could then use cross-layer adaptive packetization and UEP to achieve the required PER bounds and allocate the network resources appropriately.

In Chapter 4, we proposed a transmitter-driven scheduling scheme with a constant playout rate of 30 fps at the receiver. One could also employ a joint playout adaptation at the receiver in order to avoid playout buffer underflow during poor channel conditions which cannot support the compressed video bit rate. The video client needs to feed back the modified playout rate in order for the transmitter to compute the new frame deadlines and update the TTE values of the NAL units. It would also be worthwhile to evaluate the performance of scheduling at the APP layer along with retransmissions at the MAC and FEC at the PHY layer. We have considered only a single hop streaming scenario in this dissertation. In ad-hoc wireless networks, it is normal to stream the video over multiple hops. The scheduling algorithm at the transmitter and routers could take into account the buffer backlog of the intermediate routers, and the channel conditions of the intermediate links to minimize the number of NAL units timing out before reaching the receiver.

Bibliography

[1] K. K. R. Kambhatla, S. Kumar, S. Paluri, and P. C. Cosman, “Wireless H.264 video quality enhancement through optimal prioritized packet fragmentation,” *IEEE Trans. Multimedia*, vol. 14, no. 5, pp. 1480–1495, October 2012.

[2] T. Wiegand, G. J. Sullivan, G. Bjntegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, July 2003.

[3] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the scalable video coding extention of the H.264/AVC standard,” *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 17, no. 9, pp. 1103–1120, September 2007.

[4] S. Kumar, L. Xu, M. K. Mandal, and S. Panchanathan, “Error resiliency schemes in H.264/AVC standard,” *Elsevier J. Vis. Commun. Image R., Special issue on Emerging H.264/AVC Video Coding Standard*, vol. 17, no. 2, pp. 183–185, April 2006.

[5] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, “H.264/AVC in wireless environments,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 657–673, July 2003.

[6] Q. Zhang, W. Zhu, and Y.-Q. Zhang, “End-to-end QoS for video delivery over wireless internet,” *Proc. IEEE*, vol. 93, no. 1, pp. 123–134, January 2005.

[7] X. Zhu and B. Girod, “Video streaming over wireless networks,” in *15th EUSIPCO*, Sept. 3 - 7 2007, pp. 1462 – 1466.

[8] Y.-M. Hsiao, J.-F. Lee, J.-S. Chen, and Y.-S. Chu, “H.264 video transmissions over wireless networks: Challenges and solutions,” *Elsevier J. Comput. Commun.*, vol. 34, pp. 1661 – 1672, April 2011.

[9] A. S. Tanenbaum, *Computer Networks.* Prentice Hall, 2002.

[10] B. A. Forouzan, *Data Communications and Networking.* McGraw-Hill Forouzan Networking Series, 2007.

[11] P. Lettieri and M. B. Srivastava, "Adaptive frame length control for improving wireless link throughput, range, and energy efficiency," in *IEEE INFOCOM*, vol. 2, 1998, pp. 564–571.

[12] E. Modiano, "An adaptive algorithm for optimizing the packet size used in wireless ARQ," *Wirel. Netw. J.*, vol. 5, no. 4, pp. 279 – 286, July 1999.

[13] Y. Wang, K. Yu, Y. Liu, and H. Zhang, "Effects of MAC retransmission on TCP performance in IEEE 802.11 based ad-hoc networks," in *59th IEEE VTC*, vol. 4, February 2004, pp. 2205–2209.

[14] J. Korhonen and Y. Wang, "Effect of packet size on loss rate and delay in wireless links," in *IEEE WCNC*, vol. 3, 13 - 17 March 2005, pp. 1608 – 1613.

[15] Y. Sun, I. Sheriff, E. M. Belding-Royer, and K. C. Almeroth, "An experimental study of multimedia traffic performance in mesh networks," in *ACM WiTMeMo*, 2005, pp. 25–30.

[16] C.-W. Lee, C.-S. Yang, and Y.-C. Su, "Adaptive UEP and packet size assignment for scalable video transmission over burst-error channels," *EURASIP J. App. Signal Process.*, vol. 2006, pp. 1 – 9, January 2006.

[17] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," Microsoft Research, Tech. Rep. MSR-TR-2001-35, February 2001.

[18] T. Wiegand, N. Farber, K. Stuhlmuller, and B. Girod, "Error-resilient video transmission using long-term memory motion-compensated prediction," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 1050–1062, 2000.

[19] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Image Commun., Special Issue on Real-time video over the Internet*, vol. 15, no. 1-2, pp. 77–94, September 1999.

[20] J. Chakareski and P. Chou, "Application layer error-correction coding for rate-distortion optimized streaming to wireless clients," *IEEE Trans. Commun.*, vol. 52, no. 10, pp. 1675–1687, October 2004.

[21] S. Shakkotai, T. Rappaport, and P. Karlsson, "Cross-layer design for wireless networks," *IEEE Commun. Mag.*, vol. 41, no. 10, pp. 74–80, October 2003.

[22] M. van der Schaar and N. S. Shankar, "Cross-layer wireless multimedia transmission: Challenges, principles, and new paradigms," *IEEE Wireless Commun. Mag.*, vol. 12, no. 4, pp. 50–58, August 2005.

[23] E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod, "Cross-layer design of ad-hoc networks for real-time video streaming," *IEEE Wireless Commun.*, vol. 12, no. 4, pp. 50–58, 2005.

[24] J. Yin, X. Wang, and D. P. Agrawal, "Optimal packet size in error-prone channel for IEEE 802.11 distributed co-ordination function," in *IEEE WCNC*, 2004, pp. 1654–1659.

[25] B. S. Kim, Y. Fang, and T. F. Wong, "Throughput enhancement through dynamic fragmentation in wireless LANs," *IEEE Trans. on Veh. Technol.*, vol. 54, no. 4, pp. 1415–1425, July 2005.

[26] F. Zheng and J. Nelson, "Cross-layer adaptive design for the frame length of IEEE 802.11 networks," in *IEEE WiOPT*, 2008, pp. 437–442.

[27] P. R. Jelenkovic and J. Tian, "Dynamic packet fragmentation for wireless channels with failures," in *ACM MobiHoc*, May 2008, pp. 73–82.

[28] X. Wang, J. Yin, and D. P. Agrawal, "Analysis and optimization of the energy efficiency in the 802.11 DCF," *Springer Science J. Mobile Netw. and Appl.*, vol. 11, pp. 279–286, 2006.

[29] D. Rajan and C. Poellabauer, "Adaptive fragmentation for latency control and energy management in wireless real-time environments," in *IEEE WASA*, 2007, pp. 158–168.

[30] D. Qiao, S. Choi, and K. G. Shin, "Goodput analysis and link adaptation for IEEE 802.11a wireless LANs," *IEEE Trans. Mobile Comput.*, vol. 1, no. 4, pp. 278–292, Oct. - Dec. 2002.

[31] Y. Chang, B. Kwon, and J. A. Copeland, "Dynamic optimal fragmentation for goodput enhancement in WLANs," in *IEEE TRIDENTCOM*, 2007, pp. 1–9.

[32] Y. P. Fallah, K. Darrell, S. Avideh, K. Faizal, and P. Nasiopoulos, "A cross-layer optimization mechanism to improve H.264 video transmission over WLANs," in *IEEE CCNC*, 2007, pp. 875–879.

[33] A. T. Connie, P. Nasiopoulos, V. C. M. Leung, and Y. P. Fallah, "Video packetization techniques for enhancing H.264 video transmission over 3G networks," in *IEEE CCNC*, 2008, pp. 800–804.

[34] A. Ksentini and M. Naimi, "Toward an improvement of H.264 video transmission over IEEE 802.11e through a cross-layer architecture," *IEEE Commun. Mag.*, vol. 44, no. 1, pp. 107–114, January 2006.

[35] Y. P. Fallah, P. Nasiopoulos, and H. Alnuweiri, "Efficient transmission of H.264 video over multirate IEEE 802.11e WLANs," *EURASIP J. Wirel. Comm.*, pp. 1–14, January 2008.

[36] S. Choudhury and J. D. Gibson, "Payload length and rate adaptation for multimedia communications in wireless LANs," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 796 – 807, May 2007.

[37] B. Y. Choi, S. Song, Y. Wang, and E. K. Park, “Using RTT variability for adaptive cross-layer approach to multimedia delivery in heterogeneous networks,” *IEEE Trans. Multimedia*, vol. 11, no. 6, pp. 1194 – 1203, October 2009.

[38] C.-W. Lee, C.-S. Yang, and Y.-C. Su, “Low-complexity adaptive packet size assignment schemes for real-time scalable video transmission over WLANs,” in *ACM IWCMC*, Leipzig, Germany, 21 - 24 June 2009, pp. 1062 – 1066.

[39] C.-H. Shih, “Adaptive forward error correction combined with packet size control for wireless video,” in *IEEE Sixth IIH-MSP*, Darmstadt, Germany, 15 - 17 October 2010, pp. 256 – 259.

[40] H. Lin, T.-Y. Wu, and C.-Y. Huang, “Payload length adaptation for wireless video transmission in multicarrier systems,” in *IEEE VTC*, San Francisco, CA, USA, 5 - 8 September 2011, pp. 1 – 5.

[41] M.-F. Tsai, N. Chilamkurti, and C.-K. Shieh, “An adaptive packet and block length forward error correction for video streaming over wireless networks,” *Springer Wireless Pers. Commun.*, vol. 56, no. 3, pp. 435 – 446, February 2011.

[42] C.-H. Shih, “Enhancing packet-level forward error correction for streaming video in wireless networks,” *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 5, pp. 146–155, September 2012.

[43] T.-L. Lin and P. C. Cosman, “Efficient optimal RCPC code rate allocation with packet discarding for pre-encoded compressed video,” *IEEE Signal Process. Lett.*, vol. 17, no. 5, pp. 505–508, May 2010.

[44] T. Yoo, R. J. Lavery, A. Goldsmith, and D. J. Goodman, “Throughput optimization using adaptive techniques,” *IEEE Commun. Lett.*, pp. 1–7, 2006.

[45] S. Choudhury and J. D. Gibson, “Throughput optimization for wireless LANs in the presence of packet error rate constraints,” *IEEE Commun. Lett.*, vol. 12, no. 1, pp. 11 – 13, January 2008.

[46] T.-Y. Wu, T.-T. Chuang, and C. Y. Huang, “Optimal transmission of high definition video transmission in wimedia systems,” *Wirel. Netw. J.*, vol. 17, no. 2, pp. 291 – 303, February 2011.

[47] K. K. R. Kambhatla, S. Kumar, and P. Cosman, “Prioritized packet fragmentation for H.264 video,” in *IEEE ICIP*, 11-14 September 2011, pp. 3233–3236.

[48] K. K. R. Kambhatla, S. Kumar, P. C. Cosman, and J. Matyjas, “H.264/AVC video packet aggregation and unequal error protection for noisy channels,” in *IEEE ICIP*, Sept. 30 - Oct. 3 2012, pp. 1649–1652.

[49] K. K. R. Kambhatla, S. Paluri, S. Kumar, P. C. Cosman, and J. Matyjas, "Cross-layer prioritized H.264 video packetization and error protection over noisy channels," *EURASIP J. Wirel. Comm. (in review)*, 2014.

[50] S. Paluri, K. K. R. Kambhatla, S. Kumar, B. Bailey, P. Cosman, and J. Matyjas, "Predicting slice loss distortion in H.264/AVC video for low complexity data prioritization," in *IEEE ICIP*, Sept. 30 - Oct. 3 2012, pp. 689 – 692.

[51] L. Georgiadis, R. Guerin, and A. Parekh, "Optimal multiplexing on a single link: Delay and buffer requirements," *IEEE/ACM Trans. Netw.*, vol. 43, no. 5, pp. 1518–1535, Sept. 1997.

[52] A. A. Khalek, C. Caramanis, and R. W. Heath, "Joint source-channel adaptation for perceptually optimized scalable video transmission," in *IEEE Globecom*, Houston, TX, December 2011, pp. 1–5.

[53] T.-Y. Hung, Z. Chen, and Y.-P. Tan, "Playout adaptation based packet scheduling for scalable video delivery over wireless networks," in *IEEE ICME*, 19-23 July 2010, pp. 1022–1027.

[54] C. Chen, R. W. Heath, A. C. Bovik, and G. de Veciana, "Adaptive policies for real-time video transmission: A Markov decision process framework," in *IEEE ICIP*, 11-14 September 2011, pp. 2249–2252.

[55] A. A. Khalek, C. Caramanis, and R. W. Heath, "A cross-layer design for perceptual optimization of H.264/SVC with unequal error protection," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 7, pp. 1157–1171, August 2012.

[56] G. Alefeld and G. Mayer, "Interval analysis: theory and applications," *Elsevier J. of Comput. Appl. Math.*, vol. 121, pp. 421–464, August 1999.

[57] K. Ichida and Y. Fujii, "An interval arithmetic method for global optimization," *Springer-Verlag J. of Computing*, vol. 23, pp. 85–97, August 1979.

[58] H. Munack, "On global optimization using interval arithmetic," *Springer-Verlag J. of Computing*, vol. 48, pp. 319–336, April 1992.

[59] *H.264/AVC reference software JM14.2*, ISO/IEC Std. [Online]. Available: http://iphome.hhi.de/suehring/tml/download/

[60] *RoHC: Robust header Compression*, RFC:3095 Std., July 2001.

[61] S. Wenger, M. M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, *RFC 3894 - RTP Payload Format for H.264 Video*, Network Working Group Std., February 2005.

[62] L. Superiori, O. Nemethova, and M. Rupp, "Performance of a H.264/AVC error detection algorithm based on syntax analysis," in *Int. Conf. on Advances in Mobile Computing and Multimedia*, December 2006.

[63] S. Kumar, A. Janarthanan, M. M. Shakeel, S. Maroo, J. D. Matyjas, and M. Medley, "Robust H.264/AVC video coding with priority classification, adaptive NALU size, and fragmentation," in *IEEE MILCOM*, 2009, pp. 1702–1707.

[64] S. K. Bandyopadhyay, Z. Wu, P. Pandit, and J. M. Boyce, "An error concealment scheme for entire frame losses for H.264/AVC," in *IEEE Sarnoff*, March 2006, pp. 1–4.

[65] Z. Wu and J. M. Boyce, "An error concealment scheme for entire frame losses for H.264/AVC," in *IEEE ISCAS*, May 2006, pp. 4463–4466.

[66] S. Wolf and M. H. Pinson, "Video quality measurement techniques," National Telecommunications and Information Administration (NTIA), U.S. Department of Commerce, Tech. Rep., June 2002.

[67] M. H. Pinson and S. Wolf, "A new standardization method for objectively measuring video quality," *IEEE Trans. Broadcast.*, vol. 50, no. 3, pp. 312–322, 2004.

[68] *H.264/AVC reference software JM18.5*, ISO/IEC Std. [Online]. Available: http://iphome.hhi.de/suehring/tml/download/

[69] F. Li and G. Liu, "Compressed -domain-based transmission distortion modeling for precoded H.264/AVC video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 12, pp. 1908 – 1914, December 2009.

[70] J. Hagenauer, "Rate-compatible punctured convolutional (RCPC) codes and their applications," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, April 1988.

[71] *Local and metropolitan area networks-Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard for Information Technology-Telecommunications and information exchange between systems Std., June 2007.

[72] H. Luo, S. Ci, and D. Wu, "A cross-layer design for the performance improvement of real-time video transmission of secondary users over cognitive radio networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 8, pp. 1040 – 1048, August 2011.

[73] A. Vosoughi, V. Testoni, P. Cosman, and L. Milstein, "Joint source-channel coding of 3D video using multiview coding," in *IEEE ICASSP*, 2013, pp. 2050 – 2054.

[74] D. Li and X. Sun, *Nonlinear Integer Programming.* International Series in Operations Research and Management Science, Springer, USA, 2006.

[75] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge Univ. Press, Cambridge, U.K., 2004.

[76] Y. Z. Huang and J. G. Apostolopoulos, "A joint packet selection/omission and FEC system for streaming video," in *IEEE ICASSP*, Honolulu, HI, 15 - 20 April 2007, pp. I–845 – I–848.

[77] P. McCullagh and J. A. Nelder, *Generalized Linear Models*, 2nd ed., ser. Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, 1989.

[78] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*, 2nd ed., ser. Probability and Statistics. John Wiley & Sons, 2000.

[79] G. Schwarz, "Estimating the dimension of a model," *The Ann. Stat.*, vol. 6, no. 2, pp. 461–464, 1978.

[80] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Autom. Control*, vol. AC-19, no. 6, pp. 716–723, 1974.

[81] ——, "Canonical correlation analysis of time series and the use of an information criterion," *R. K. Mehra and D. G. Lainotis (eds.), System Identification: Advances and Case Studies, Academic Press, New York*, pp. 52–107, 1976.

[82] E. J. Hannan and B. G. Quinn, "The determination of the order of an autoregression," *J. Royal Stat. Soc., B*, vol. 41, no. 2, pp. 190–195, 1979.

[83] The R Project for Statistical Computing. Vienna, Austria. [Online]. Available: http://www.r-project.org/

[84] N. Farber and B. Girod, *Compressed Video over Networks.* Marcel Dekker, 2000, ch. Wireless Video, pp. 465–512.

[85] J. R. Ohm, "Advances in scalable video coding," *Proc. IEEE*, vol. 93, no. 1, pp. 42–56, January 2005.

[86] Y. Wang and S. Lin, "Error-resilient video coding using multiple description motion compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 438–452, August 2002.

[87] C. I. Lin, W. N. Lie, and C. W. Lin, "Enhancing video error resilience by using data embedding techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 2, pp. 300–308, February 2006.

[88] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: An overview," *IEEE Signal Process. Mag.*, vol. 20, no. 2, pp. 18–29, March 2003.

[89] C. W. Lin, J. Xin, and M. T. Sun, "Digital video transcoding," *Proc. IEEE*, vol. 93, no. 1, pp. 84–97, June 2005.

[90] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, April 2006.

[91] M. Kalman, E. Steinbach, and B. Girod, "Adaptive media plyout for low-delay streaming over error-prone channels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 841–851, June 2006.

[92] Y. F. Su, Y. H. Yang, and H. Chen, "Smooth control of adaptive media playout for video streaming," *IEEE Trans. Multimedia*, vol. 11, no. 7, pp. 1331–1339, November 2009.

[93] Y. Li, A. Markopoulou, J. Apostolopoulos, and N. Bambos, "Content-aware playout and packet scheduling for video streaming over wireless links," *IEEE Trans. Multimedia*, vol. 10, no. 5, pp. 885–895, August 2008.

[94] H. Chiang, C. Y. Huang, and T. Chiang, "Content-aware adaptive media playout controls for wireless video streaming," *IEEE Trans. Multimedia*, vol. 9, no. 6, pp. 1273–1283, September 2007.

[95] X. Liu, E. K. P. Chong, and N. B. Shroff, "A framework for opportunistic scheduling in wireless networks," *Elsevier J. Comput. Netw.*, vol. 41, no. 4, pp. 451–474, March 2003.

[96] L. Georgiadis, M. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–144, 2006.

[97] B. Girod, J. Chakareski, M. Kalman, Y. J. Liang, E. Setton, and R. Zhang, "Advances in network-adaptive video streaming," in *IWDC 2002*, Capri, Italy, September 2002.

[98] B. Girod, M. Kalman, Y. J. Liang, and R. Zhang, "Advances in channel-adaptive video streaming," in *IEEE ICIP*, Rochester, NY, September 2002.

[99] M. Kalman and B. Girod, *Multimedia Networking and Communication.* Elsevier, 2007, ch. Network-Adaptive Media Streaming, pp. 223–232.

[100] N. G. Duffield, K. K. Ramakrishnan, and A. R. Reibman, "SAVE: an algorithm for smoothed adaptive video over explicit rate networks," in *IEEE INFOCOM*, San Francisco, CA, USA, March 1998, pp. 1093–1102.

[101] T. Ott, T. V. Lakshman, and A. Tabatabai, "A scheme for smoothing delay sensitive traffic offered to ATM networks," in *IEEE INFOCOM*, Florence, Italy, 4-8 May 1992, pp. 776–785.

[102] Z. L. Zhang, J. Kurose, J. Salehi, and D. Towsley, "Smoothing, statistical mulitplexing, and call admission control for stored video," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 6, pp. 1148–1166, August 1997.

[103] M. Kalman, E. Steinbach, and B. Girod, "Adaptive media playout for low delay video streaming over error-prone channels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 841–851, June 2004.

[104] M. Kalman and B. Girod, "Rate-distortion optimized video streaming with multiple deadlines for low latency applications," in *Packet Video Workshop*, Irvine, CA, December 2004.

[105] J. Chakareski, J. G. Apostolopoulos, S. Wee, W. T. Tan, and B. Girod, "Rate-distortion hint tracks for adaptive video streaming," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 10, pp. 1257–1269, October 2005.

[106] T. Stockhammer, H. Jenkac, and G. Kuhn, "Streaming video over variable bit-rate wireless channels," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 268–277, April 2004.

[107] Z. Chen and Y. Reznik, "Analysis of video codec buffer and delay under time-varying channel," in *IEEE VCIP*, 27-30 Nov. 2012, pp. 1–6.

[108] Q. Zhang and S. A. Kassam, "Finite-state Markov model for Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 47, no. 11, pp. 1688 – 1692, November 1999.

[109] P. Sadeghi, R. A. Kennedy, P. B. Rapajic, and R. Shams, "Finite-state Markov modeling of fading channels," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 57 – 80, September 2008.

[110] H. C. Chuang, C. Y. Huang, and T. H. Chiang, "Content-aware adaptive media playout controls for wireless video streaming," *IEEE Trans. Multimedia*, vol. 9, no. 6, pp. 1273 – 1283, Sept. 2007.

[111] Y. F. Su, Y. H. Yang, M. T. Lu, and H. H. Chen, "Smooth control of adaptive media playout for video streaming," *IEEE Trans. Multimedia*, vol. 11, no. 7, pp. 1331 – 1339, Nov. 2009.

[112] S. H. Kang and A. Zakhor, "Packet scheduling algorithm for wireless video streaming," in *International Packet Video Workshop*, 2002, pp. 1–11.

[113] Z. Jiang and L. Kleinrock, "A packet selection algorithm for adaptive transmission of a smoothed video over a wireless channel," *Elsevier J. Parallel Distr. Com.*, vol. 60, no. 4, pp. 494–509, April 2000.

[114] T. Schierl, Y. S. de la Fuente, R. Globisch, C. Hellge, and T. Wiegand, "Priority-based media delivery using SVC with RTP and HTTP streaming," *Springer Multimed. Tools Appl.*, vol. 55, no. 2, pp. 227–246, September 2011.

[115] S. Xiao, H. Wang, and C.-C. J. Kuo, "A practical bit stream organization algortihm for robust H.264/SVC transmission," *Elsevier J. Vis. Commun. and Image R.*, vol. 21, no. 8, pp. 871–879, November 2010.

[116] Z. Chen and D. Wu, "Rate-distortion optimized cross-layer rate control in wireless video communication," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 3, pp. 352–365, March 2012.

[117] Z. He and S. K. Mitra, "Optimum bit allocation and accurate rate control for video coding ρ-domain source modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 10, pp. 840–849, October 2002.

[118] A. Dua, C. W. Chan, N. Bambos, and J. Apostolopoulos, "Channel, deadline and distortion (CD^2) aware scheduling for video streams over wireless," *IEEE Trans. Wireless Commun.*, vol. 9, no. 3, pp. 1001–1011, March 2010.

[119] A. Combernoux, C. Delestre, N. Changuel, B. Sayadi, and M. Kieffer, "Cross-layer optimization of a multimedia streaming system via dynamic programming," in *IEEE ICIP*, Sept. 30 - Oct. 3 2012, pp. 2261–2264.

[120] Y. Zhang, F. Fu, and M. van der Schaar, "On-line learning and optimization for wireless video transmission," *IEEE Trans. Signal Process.*, vol. 58, no. 6, pp. 3108–3124, June 2010.

[121] *Advanced Video Coding for Generic Audio Visual Services*, ITU-T H.264 — ISO/IEC 14496-10 Std., January 2012.

[122] JSVM 9.8. JSVM SVC reference software manual. [Online]. Available: http://ube.ege.edu.tr/∼ boztok/JSVM/SoftwareManual.pdf.

[123] Y. Sanchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. D. Vleeschauwer, W. V. Leekwijck, and Y. L. Louedec, "Efficient HTTP-based streaming using scalable video coding," *Elsevier Signal Process. - Image*, vol. 27, no. 4, pp. 329–342, April 2012.

[124] J. McDougall and S. Miller, "Sensitivity of wireless network simulations to a two-state Markov model channel approximation," in *IEEE Globecom*, vol. 2, San Francisco, CA, Dec. 1-5 2003, pp. 697–701.

[125] S. Karande, M. K. S. Khayam, and H. Radha, "Analysis and modeling of errors at the 802.11b link layer," in *IEEE ICME*, vol. 1, 6-9 July 2003, pp. 673–676.

[126] J. Hartwell and A. Fapojuwo, “Modeling and characterization of frame loss process in IEEE 802.11 wireless local area networks,” in *IEEE VTC ’04*, vol. 6, Los Angeles, CA, Sept. 26-29 2004, pp. 4481–4485.

[127] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementation*. John Wiley and Sons, 1990.

[128] L. Caccetta and A. Kulanoot, “Computational aspects of hard Knapsack problems,” *Nonlinear Anal. - Theor.*, vol. 47, no. 8, pp. 5547–5558, 2001.

[129] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*, W. Stephen, Ed. McGraw-Hill Science/Engineering/Math, December 2001.