The Graduate University for Advanced Studies

Doctoral Thesis

Error Resilient Multi-view Video Streaming

Author:
Zhi LIU

Supervisor:
Yusheng JI

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Informatics

in

The Graduate University for Advanced Studies

July 2014

To my parents and sister

Abstract

Video is becoming more and more popular. Traditional video streaming only allows users passively receive and playback the video, without any freedom to select the viewing angles. But users tend to change the viewing angles during the video playback especially when watching the sports events, etc. How to provide users the freedom to select favorite viewing angles becomes an important issue. *Interactive multi-view video streaming* (IMVS), which offers viewers the freedom to switch to a desired neighboring captured view periodically, and *free viewpoint video streaming* (FVV), which offers viewers the freedom to choose any viewing angle between the left-most and right-most camera are two types of multi-view video streaming, which could satisfy users' view switch requests. By providing users this freedom, multi-view video is fast becoming the core technology for a number of emerging applications such as free viewpoint TV and is expected to be the next generation of visual communication. One fundamental research problem to achieve these applications is how to deliver the video contents over the networks. The challenges exist in the network channels' unavoidable packet losses, video's playback deadline, the impracticality of retransmission on a per packet, per client basis besides how to satisfy users' view switch requests in an efficient manner.

This dissertation focuses on the application-level streaming optimization and discusses how to deliver the multi-view video over lossy networks considering video encoding methods, frame error concealments which are the works in the signal processing community at the application layer and streaming protocols, cooperative strategies which are the works in the communication community at the network transport layer. And this work is denoted as `error resilient multi-view video streaming`.

Specifically, to alleviate the wireless packet loss problem, *cooperative peer recovery (CPR) for multi-view video multicast* was proposed, where the decision process for individual peers during CPR for recovery of multi-view video content was optimized using *Markov decision process* (MDP) as a mathematical formalism, so that a loss-stricken peer can then either recover using received CPR packets of the same view, or using packets of two adjacent views and subsequent view interpolation via image-based rendering. In addition to the loss-resilient aspect during network streaming, we also address how to design efficient coding tools and optimize frame structure for transmission to facilitate view switching and contain error propagation in differentially coded video due to packet losses by using a new *unified distributed source coding* (uDSC) frame. After inserting uDSC-frames into the coding structure, we schedule packets for network transmission in a rate-distortion optimal manner for both wireless multicast and wired unicast streaming scenarios.

During FVV wireless network transmission, burst packet losses can corrupt the transmitted texture and depth videos and degrade the synthesized view quality at the client. Since two network paths suffer burst packet losses at the same time is very unluckily, we propose a system for multiple description coding of free-viewpoint video transmitted over two network paths and joint interview / temporal frame recovery scheme using frames in the received description. Near-optimal source and channel coding rates for each description are selected using a branch-and bound method, for the given transmission bandwidth on each path, as introduced in *multi-path free viewpoint video streaming.*

Extensive experiments for these three proposed solutions were conducted. The simulation results showed the proposed schemes' outperformance over the competing schemes in typical network scenario, which indicates that the proposed error-resilient multi-view video streaming schemes enable users to enjoy better multi-view video services in term of received video quality.

Acknowledgements

First, I would like to thank my advisors Prof. Yusheng Ji and Assoc. Prof. Gene Cheung for their continuous support during my Ph.D study and research. Their patience, enthusiasm, knowledge and guidance help me during all the time of research and life in Japan. I could not have imagined having better advisors and mentors for my Ph.D study.

I would like to thank the rest of my committee: Prof. Noboru Sonehara, Prof. Shigeki Yamada, Assoc. Prof. Kensuke Fukuda and Assoc. Prof. Keita Takahashi for their insightful comments and advises.

My sincere thanks also goes to Prof. Sherman Shen from University of Waterloo, Assoc. Prof. Pascal Frossard from Swiss Federal Institute of Technology in Lausanne (EPFL), Assoc. Prof. Jie Liang and Ivan Bajic from Simon Fraser University, Assit. Prof. Jacob Chakareski from University of Alabama, for offering me the short-visit opportunities in their groups and the fruitful discussions. Their research attitudes, knowledge motivates me. I also want to thank Prof. Jiannong Cao from The Hong Kong Polytechnic University, Prof. Antonio Ortega from University of Southern California, Prof. Fuqiang Liu from Tongji University and Prof. Baohua Zhao from University of Science and Technology of China, et al. The conversations with them inspire me greatly. I also thank all the adminstration staffs in National Institute of informatics (NII) and The Graduate University for Advanced Studies (Sokendai), the secretaries of Prof. Ji. Their guidance and help made my life in Japan much more easier.

I thank my friends in NII: Liping Wang, Lei Zhong, Kien Nguyen, Yunlong Feng, Ruijian An, Saran Tarnoi, Yu Mao, Shuyu Shi, Jingyun Feng, et al. The visiting researcher or interns: Yu Gu, Stephan Sigg, Kalika Suksomboon, Hao Zhou, Junfeng Jin, Wei Luo, Bo Hu, Haidi Yue, Lin Su, et al. I thank them for the stimulating discussions, for the countless get-togethers during the past five years. The list is so long that I can not write down all their names.

Last but not the least, I would like to thank my family: my parents and my sister. Their spirit support throughout my life is my most precious treasure.

Contents

List of Figures

Abbreviations

IMVS	interactive multi-view video streaming
FVV	free viewpoint video streaming
CPR	cooperative peer-to-peer repair
MDP	Markov decision process
uDSC	unified distributed source coding
HTTP	hypertext transfer protocol
HLS	HTTP Live Streaming
DASH	dynamic adaptive streaming over HTTP
TCP	transmission control protocol
UDP	user datagram protocol
FEC	forward error correction
ECC	error-correcting code
HEVC	high efficiency video coding
DIBR	depth image based rendering
WWAN	wireless wide area network
SNC	structured network coding
PSNR	peak signal-to-noise ratio
MVV	multi-view video Coding
RFS	reference picture selection
ARQ	automatic retransmission request
NC	network coding
UEP	unequal error protection
GOP	group of pictures
RaDio	rate-distortion optimized streaming
TO	transmission opportunity

iid	independent & identically distributed
GE	Gilbert-Elliot
WLAN	wireless local area network
RD	rate-distortion
POMDP	partially observable MDP
MDC	multiple description coding
TSR	temporal super-resolution
ME	motion estimation
MC	motion compensation
BME	bidirectional ME
OMC	overlapped motion compensation
IBR	image-based rendering
MBMS	multimedia broadcast/multicast service
ACK	acknowledgement
UNC	unstructured Network Coding
MRU	maximum transport unit
NAK	negative acknowledgement
SI	side information
DE-DSC	drift-elimination distributed source coding
MP-DSC	multiple-predictor distributed source coding
MVs	motion vectors
DCT	discrete cosine transform
LDPC	low-density parity check
DP	dynamic programming
QP	quantization parameters
MSE	mean square error
BB	branch-and-bound
SAD	sum of absolute differences
WMF	weighted mode filter
VQ	vector quantization

Chapter 1

Introduction

This chapter provides an brief overview of the video streaming and highlights the importance of multi-view video streaming over lossy networks. The author's contributions are listed and the dissertation organization is introduced at the end of this chapter.

1.1 Background & Motivation

Video, as an electronic medium for the recording, copying and broadcasting of moving, is becoming more and more popular. Nowadays, people are benefiting various video application, such as video teleconferencing [1, 2], live broadcasting [3, 4], video on demand [5, 6], etc. The video network transmission is one important part of all these applications. The challenges exist in the unavoidable packet losses and tight playback deadline of the video services. The packet losses of wired network channels may happen because of the congestion. The unavoidable packet losses of wireless channels come from shadowing, channel fading and inter-symbol interference.

Recent measurement studies have shown that a significant fraction of commercial streaming traffic uses *hypertext transfer protocol* (HTTP) [7]. For example, *HTTP live streaming* (also known as HLS) [8] is an HTTP-based media streaming communications protocol implemented by Apple Inc. as part of their QuickTime and iOS software. HLS breaks the video stream into a sequence of small HTTP-based file downloads for downloading. It downloads a playlist containing the metadata for the various sub-streams, which are

available at the start of the streaming session. During the video playback, the user may select from a number of different alternate streams of the same video content but encoded at a variety of data rates, adapting to the available data rate. *Dynamic adaptive streaming over HTTP* (DASH), also known as *MPEG-DASH*, is an adaptive streaming technique in term of bit rates that enables high quality video streaming over the networks using existing HTTP web server infrastructure, where the HTTP web server infrastructure was proposed for essentially all World Wide Web content' delivery. DASH is the first international standard in the filed of adaptive HTTP-based video streaming and is a technology related to *Adobe HTTP Dynamic Streaming* [1], *Apple Inc. HTTP Live Streaming* [8] and *Microsoft Smooth Streaming* [9]. The idea of MPEG-DASH is similar to Apple's HLS. MPEG-DASH also breaks the video content into multiple small HTTP-based file segments. The video contents are encoded at different bit rates. Then the user can select from the alternatives to download and playback based on his/her network conditions as the video is played back. The user chooses the segment with the largest bit rate possible that could be transmitted in time for playback without *stalls* or *rebuffering* events during playback. To handle packet losses over the networks, these HTTP/TCP(*transmission control protocol*)-based strategies adopt persistent packet retransmission. This will lead to video pause during multiple retransmission attempts, which degrades the quality of video service greatly [10, 11] and hence not desirable.

User datagram protocol (UDP) offers shorter latency since there is no retransmission and no handshaking dialogues. Hence it is more suitable for time sensitive applications e.g. VoIP [12] and vide streaming [13–15]. The problem with UDP is that it provides no guarantee of the data delivery. Moreover, UDP is an unresponsive protocol, which means it does not reduce its data rate when there is a congestion. Given the number of streaming flows is expected to grow rapidly, UDP will introduce more congestion. Response to congestion is very important for the health of the Internet. To avoid this congestion collapse, more recent transport protocols uses the congestion control for video streaming [15, 16], which means they are *TCP-friendly* [17].

Orthogonally, in telecommunication, information theory and coding theory, *forward error correction* (FEC) [18, 19] is widely used to tolerant errors in data transmission over noisy communication channels including the video streaming [20, 21]. The general idea is that server transmits the data in a redundant way by using an *error-correcting code*

[1] http://www.adobe.com/products/hds-dynamic-streaming.html

(ECC) [22]. Given that the wireless channels are burst-loss prone [23] and there is stringent playback deadline for video frames, FEC does not work efficiently in video streaming over wireless networks. Deploying sufficient amount of FEC even for the worst-channel peer translates to a large overhead, leaving preciously few bits out of a finite transmission budget for source coding to fight quantization noise, resulting in poor video quality.

In this dissertation, what we focus is the application-level streaming optimization, which unlike TCP, has knowledge about what's inside the packet payload so smarter decisions can be made regarding what to transmit, how to transmit and when. The lost frames could also be recovered to some extend.

Figure. 1.1 illustrates one typical video system, which is composed of the video capturing part, video transmission part and users side. As consumer-level cameras are becoming cheaper and cheaper, the scene of interest can now be captured by multiple cameras simultaneously from different viewpoints. Depth maps, which record the distances between the scene and cameras at pixel level, can also be captured besides the texture maps if necessary. Server encodes the captured video using the state-of-the-art encoding method such as H.264 [24], *High Efficiency Video Coding* (HEVC) [25] first before transmission. The technical problems of the video encoding are how to encode them in a coding efficient manner [24–27] or loss resilient manner [28, 29], etc. Then upon a user's request (or live broadcasting), the video contents are delivered to users via the network (core wired network or wireless network). The data transmission problems include how to jointly do the source/chnanel coding problem [30–37], etc. After receiving the frames, the client can decode the view or recover the lost frames if there are and then playback the video. How to perform the frame error concealment [38, 39] when loss happens is one important issue for video streaming.

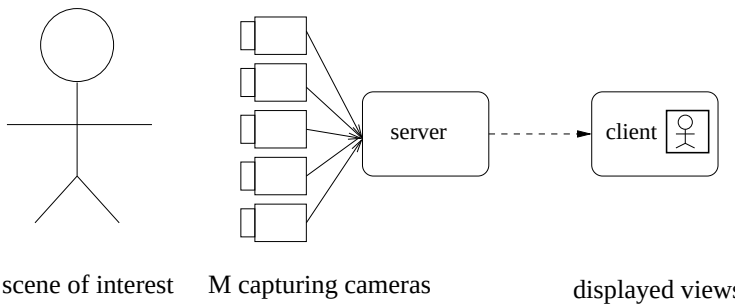


FIGURE 1.1: Overview of core components in a general video system.

Please note that the traditional video streaming systems provide no interaction between users and the video systems, i.e. users just passively receive and playback the video contents. While users tend to change the viewing angles during the playback especially when watching the sports events, dancing, etc. How to provide users the freedom to select the viewing angles freely becomes an important issue. IMVS [40–46] and FVV [47–51] are recently widely studied. In IMVS, a user observes one view at a time, but can periodically switch to a desired neighboring captured view as the video is played back in time. In FVV, by transmitting texture and depth videos captured from two nearby camera viewpoints, a client can synthesize via *depth image based rendering* (DIBR) any freely chosen intermediate virtual view of the 3D scene, enhancing the user's perception of depth.

By providing users the freedom to select the viewing angles, multi-view video is fast becoming the core technology for a number of emerging applications such as free viewpoint TV, immersive video conferencing, multi-view version of YouTube and is expected to be the next generation of visual communication. How to transmit them over the lossy network is one fundamental problems to achieve these applications. Comparing with the single view video streaming, the additional challenges are how to utilize the multi-view information besides the view itself for better performance. In this dissertation, we assume the multi-view video have already been captured, and there is no retransmission for lost packets over the wireless network channels, where the network are with limited bandwidth constraints and could be simulated using state-of-the-art channel loss models. We focus on the problem of how to deliver the video in IMVS and FVV scenarios including the video encoding, video transmission and lost frames' error concealment. We designed the coding methods which are the work in the single processing community at the application layer; and proposed streaming protocols and cooperative strategies which are the work in the communication community at the network transport layer, the cross-layer, and cross-community solution makes it challenging, effective and impressive.

Specifically, to alleviate individual *wireless wide area network* (WWAN) packet losses, a cooperative local recovery scheme was proposed, where we optimized the decision process for individual peers during CPR for recovery of multiview video content in IMVS. This work is denoted as *cooperative peer recovery for multi-view video multicast*.

In addition to the loss-resilient aspect during network streaming, we also designed a new

uDSC frame for periodic insertion into the multi-view frame structure to facilitate view-switching and contain error propagation for IMVS. After inserting uDSC-frames into the coding structure, we schedule packets for network transmission in a rate-distortion optimal manner for both wireless multicast and wired unicast streaming scenarios. This work is called *unified distributed source coding for interactive multi-view video streaming.*

The wireless network channels are burst-loss prone, which makes the traditional channel coding scheme (such as adding FEC) inefficient, but two independent paths suffer the burst loss at the same time is very unlikely. We proposed to encode the texture and depth signals of two camera captured viewpoints into two independently decodeable descriptions for transmission over two disjoint wireless network paths. We designed a novel missing frames recovery scheme using frames in the received description by exploiting the temporal and inter-view correlation of the transmitted viewpoints. Near-optimal source and channel coding rates for each description were selected using a branch-and-bound method, for the given transmission bandwidth on each path. We name this work as *multi-path free viewpoint video streaming.*

1.2 Contributions

In *cooperative peer recovery for multi-view video multicast* for IMVS multicast, local repair was introduced to alleviate the wireless packet loss problem using *distributed Markov decision process.* We optimized the decision process for individual peers during CPR for recovery of multi-view video content in IMVS, so that a loss-stricken peer can then either recover using received CPR packets of the same view, or using packets of two adjacent views and subsequent view interpolation via image-based rendering. In particular, for each available transmission opportunity, a peer decides—using MDP as a mathematical formalism—whether to transmit, and if so, how the CPR packet should be encoded using *structured network coding* (SNC). The experimental results showed that the proposed distributed MDP can outperform random schemes by at least 1.8dB in term of the received video quality in typical network scenarios evaluated using *peak signal-to-noise ratio* (PSNR).

In *unified distributed source coding for interactive multi-view video streaming*, we used the distributed source coding as a tool to halt the video error propagation and provide the view switch. The contributions are listed as follows:

- Leveraging on previous work on DSC, we propose a new uDSC frame to simultaneously facilitate view switching and halt error propagation in differentially coded video, and periodically insert them into a pre-encoded IMVS frame structure.
- Given inserted uDSC frames in coding structure, at streaming time we optimize transmission for wireless multicast and wired unicast scenario, so that uDSC is decoded with high probability and the visual quality of the decoded video is maximized.
- We conducted extensive experiments to show that our frame structures using uDSC frames with optimized transmission can outperform other competing coding schemes by up to 2.8 and 11.6dB for the multicast and unicast scenarios, respectively.

In the chapter of *multi-path free viewpoint video streaming*, we propose to transmit the FVV through two paths since FEC does not work efficiently over the burst-loss prone wireless channels, but two paths suffer from the burst loss at the same time is very unlikely. The contributions of this work are as follows:

- We propose to encode the texture and depth signals of two camera captured viewpoints into two independently decodeable descriptions for transmission over two disjoint wireless network paths. We designed a novel missing frames recovery scheme using frames in the received description by exploiting the temporal and inter-view correlation of the transmitted viewpoints
- Near-optimal source and channel coding rates for each description are selected using a branch-and-bound method, for the given transmission bandwidth on each path.
- Extensive experimental are conducted and the results show that our system can outperform a traditional single-description / single-path transmission scheme by up to 5.5dB in PSNR of the synthesized intermediate view at the client

1.3 Dissertation Organization

The reminder of this dissertation is organized as follows:

Chapter 2 introduces this dissertation's related work.

Chapter 3 describes how we optimized the decision process for individual peers during CPR for recovery of multi-view video content in IMVS without asking the sever to retransmit the lost packets.

Chapter 4 introduces how we construct the uDSC frame and how we optimize the uDSC frame insertion, the wireless multicast and wired unicast transmission. The simulation results and the chapter summary are included at the end of this chapter.

Chapter 5 explains how we encode the two nearest neighboring views' texture and depth maps into two independently decodeable descriptions for transmission over two disjoint wireless network paths. Then the frame recovery scheme exploring the temporal and inter-view correlation of the transmitted viewpoints, and the video data transmission optimization are introduced. The experiments results and the conclusion are written at the end of this chapter.

Chapter 6 concludes this dissertation, discussed the solutions proposed and introduces some research issues that have not been well addressed.

Chapter 2

Background

This chapter introduces the related researches of IMVS/FVV video streaming. This chapter's organization and why these related works are introduced are explained in Section 2.1.

2.1 Overview

This chapter overview all the related researches in the filed of IMVS/FVV video streaming. For both IMVS/FVV, the captured video need to be encoded before transmission, Section 2.2 introduces the multi-view video coding methodology. This dissertation is all about video streaming, an overview about the video streaming is given in Section 2.3. In this dissertation, we use the state-of-the-art network channel loss models to simulate the network channels and the related researches in the filed of network channel loss models are explained in Section 2.4. Chapter 3 discusses how to use MDP for the cooperative local repair, hence the related work in the filed of cooperative local repair and MDP are introduced in Section 2.5 and Section 2.6, respectively. Chapter 4 introduces how we optimally use the DSC as a tool to provide the timely view switch and stop the video error propagation for IMVS. The DSC researches in literature are introduced in Section 2.7. Chapter 5 is about the FVV multipath transmission. In order to transmit the FVV over multiple paths, we need to divide the video into multiple descriptions, hence multiple description coding is used and the related work is introduced in Section 2.8. When one path enters the 'bad' state, frame recovery is conducted exploiting the temporal and

inter-view correlation of the transmitted viewpoints. The literature work about how to utilize the temporal correlation of transmitted viewpoints is introduced in Section 2.9.

2.2 Multi-view Video Coding

Multi-view Video Coding (MVC) [52] is an extension of the single-view video coding standard H.264/AVC [24], where multiple texture maps from closely spaced capturing cameras are encoded into one bitstream. Much research in MVC has been focused on compression of all captured frames across time and view, exploiting both temporal and inter-view correlation to achieve maximal coding gain [52–54]. Though suitable for compact storage of all multi-view data (e.g., on a DVD disc), for our intended IMVS application [55] where only a single view requested by client is needed at a time, complicated inter-frame dependencies among coded frames across time and view reduce the random decodability of the video stream. In other words, inter-dependencies among frames of different views mean that, often, more than one video view/frame must be transmitted in order to correctly decode and display a single view/frame, leading to an unnecessary increase in streaming rate.

Given the limitation of MVC frame structures for IMVS, one previous work [56] proposed to encode each video view in the multi-view content into three streams to allow for three kinds of view interactions. [55] focused on the design of good redundant frame structure to facilitate periodic view-switches, where by pre-encoding more likely view navigation paths *a priori*, expected IMVS transmission rate can be traded off with storage size of the coding structure. To avoid decoder complexity of performing DIBR for view synthesis of intermediate virtual views, error propagation in differentially coded video frames stemming from irrecoverable packet losses in the communication networks, however, was not considered in these works. In contrast, we propose a new frame type (uDSC frame) that offers both periodic view-switches and error resiliency for insertion into a multi-view coding structure, and optimize its transmission for two practical IMVS streaming scenarios, which will be introduced in Chapter 4.

The texture-plus-depth format of free viewpoint video [57] is another multi-view representation that encoded texture and depth maps associated with multiple captured viewpoints, so that a user can also choose intermediate virtual viewpoints between every

two captured view, for an enhanced 3D viewing experience. Depth maps possess unique piecewise smooth signal characteristics that can be exploited for coding gain [58–60]. Given that temporal redundancy has already been exploited via MC, and neighboring temporal frames tend to be more similar than neighboring inter-view frames due to the typically high frame rate of captured videos, it was shown that additional coding gain afforded by disparity compensation is noticeable, but not dramatic (around 1dB in PSNR [52]). Since we focus on error-resilient streaming of free viewpoint video, for simplicity, we employ the standard H.264 video codec for coding texture and depth maps. Using more advanced coding tools is left as future work.

2.3 Error Resilient Networked Video Streaming

Error resilient networked video streaming is a well studied topic. At the application layer, *reference picture selection* (RPS) has been proposed as a way to mitigate error propagation in differentially encoded video frames, by predicting from frames further in the past that have higher probability of correct decoding [61, 62]. At the transport layer, different combinations of FEC and *automatic retransmission request* (ARQ) have been proposed for different network settings [63, 64]. More recently, *network coding* (NC) has been used as a FEC tool for *unequal error protection* (UEP) of video packets of different importance [65–67]. Our optimized transport of multi-view frames for wireless multicast also employs UEP and applies FEC of different strengths to protect motion and residual packets in a *group of pictures* (GOP) unequally. This is done so that our proposed uDSC frames are correctly decoded with high probability to halt error propagation. Further, unlike RPS, our proposed insertion of uDSC frames into coding structures does not require real-time video encoding.

Among works that proposed transport layer protection like FEC and ARQ, *Rate-distortion optimized streaming* (RaDio) [37] studied more methodically what optimal packet transmission decision should be made at sender at each *transmission opportunity* (TO) using MDP as a mathematical formalism.

While the problem of error-resilient streaming of single-view video has been extensively studied, error-resilient streaming of free viewpoint video is an emerging topic. In [68], a scheme to minimize the expected synthesized view distortion based on RFS [62] at the

block level was proposed for depth maps only. In a follow-up work, [69] extended the idea proposed in [68] to encoding of both texture *and* depth maps. Lastly, in [70] the work is extended to the case where optimization of source coding rate (via an optimal selection of *quantization parameters* (QP)) is included into the error-resilient streaming framework. However, in [68–70] a simple *Independent and identically distributed* (*i.i.d.*) packet loss model is adopted, while in wireless networks it is more common to observe burst packet loss events [23].

2.4 Network Loss Models

Packet losses in network channels can be modeled by *i.i.d.* [71], *2-state Markov* [72] and *Gilbert-Elliot* (GE) [23]. Since i.i.d. model assumes packet losses occur independently, it is not suitable for wireless channels, which are burst-loss prone. The channel models by *2-state Markov* and GE can explain the burst loss by describing packet losses in a simple and idealized way. GE model is a commonly used model for wireless losses [23], which has packet loss probabilities g and b for each of *good* and *bad* state, and state transition probabilities p and q to move between states. *2-state Markov* is a special case of GE, where $g = 0, b = 1$. In this dissertation, we use GE model and i.i.d. model to model packet losses in IMVS wireless multicast and wired unicast, respectively.

2.5 Cooperative Local Repair

Because the probability of packet loss in WWAN can be substantial, conventional WWAN video multicast schemes [73] involve a large overhead of FEC packets. Previous work on CPR [65] differs from these traditional approaches by utilizing a secondary ad hoc *wireless local area network* (WLAN) network for local recovery of packets lost in the primary network exploiting peers' cooperation. [65] has shown substantial gain in visual quality using CPR over non-cooperative schemes.

We stress that the assumption of multi-homing capable devices (each has multiple network interfaces to connect to orthogonal delivery networks simultaneously) is a common one in the literature [74–76], where different optimizations are performed exploiting the

multi-homing property. [74] shows that aggregation of an ad hoc group's WWAN bandwidths can speed up individual peer's infrequent but bursty content download like web access. [75] proposes an integrated cellular and ad hoc multicast architecture where the cellular base station delivered packets to proxy devices with good channel conditions, and then proxy devices utilized local ad hoc WLAN to relay packets to other devices. [76] shows that smart striping of FEC-protected delay-constrained media packets across WWAN links can alleviate single-channel burst losses, while avoiding interleaving delay experienced in a typical single-channel FEC interleaver. [65] extended this body of multi-homed literature to WWAN video broadcast. We further improve [65] via formal optimization of the packet selection decision at each peer using distributed MDP.

2.6 Markov Decision Process

MDP was first used for packet scheduling in a *rate-distortion* (RD) optimal manner in the seminal work RaDio [37]. Using *partially observable MDP* (POMDP), [77] extended the RaDiO work and considered scheduled transmissions of coded packets by a single sender using network coding when observations of client states are not perfect. [78] addressed the problem of selecting transmission video rates to neighboring peers in a P2P streaming scenario, where each peer must make its own decision in a distributed manner. Our work introduced in Chapter 3 also leverages on the power of MDP in decision making; in particular, we tailor MDP to the WWAN video broadcast with CPR scenario, where the decision a peer must make is whether to transmit, and if so, which SNC type to encode a repair packet for local WLAN transmission.

2.7 Distributed Source Coding

Much of the early DSC work for video coding, based on information theory developed by Slepian-Wolf [79] for the lossless case and Wyner-Ziv [80] for the lossy case, has focused on the reduction of encoder complexity, where the computation cost of motion estimation is shifted to the decoder [81, 82]. Beyond encoder complexity reduction, DSC can also be used for efficient encoding of correlated media data that requires random access during navigation, without resorting to large independently coded I-frames. Examples include light field navigation [83], flexible playback of single-view video [84], and aforementioned

IMVS [55, 85]. See [86] for a more extensive overview on coding for random access in media navigation. None of these works, however, considered network packet losses and their impact on visual quality during media navigation.

In an orthogonal development, [29] proposed a DSC-based tool to halt error propagation in single-view video at the periodic DSC-frame boundary. In Chapter 4, we combine the advantages of previous DSC usages via the construction of a single unified DSC frame that can halt error propagation *and* facilitate view-switching. This is the first notable difference from previous work. Second, [29] did not discuss how the transmission of the proposed DSC frame can be optimized in a real network streaming scenario. In Chapter 4, a frame structure is designed using the proposed unified DSC frames, and then its transport is optimized for two network streaming scenarios: wireless IMVS multicast and wired IMVS unicast. This is the second major difference from [29].

2.8 Multiple Description Coding

Multiple description coding (MDC) has been proposed for multi-path streaming of single-view video [87–91]. In particular, in [87] the even and odd frames of a video are encoded separately into two descriptions; we follow the same paradigm in our MDC design as well. However, the recovery of a lost description in [87] relies on conventional block-based ME using temporal neighboring frames [92], which does not result in accurate recovery of the motion field per pixel. In contrast, this dissertation proposes a sub-block-based ME scheme, where a block can potentially be divided into foreground sub-block and background sub-block using available depth information. As a result, we can recover more accurate per-pixel motion information at comparable complexity.

Note that in the multiple description literature for single-view video, there exist studies [90, 93] that generalize the number of descriptions to $N > 2$ sent over disjoint network paths. However, it has been shown [89, 90] that video coding performance of a system based on $N > 2$ descriptions drops dramatically, due to the inefficiency of motion compensated video coding when the temporal distance between the target frame and the predictor frame is larger than two [28, 94]. This will hold true for free viewpoint video

as well, given that the prediction structure in our coding scheme is similar to the single-view video case. Thus, though in theory employing $N > 2$ descriptions is possible, we encode only two descriptions in our proposed system.

The work in [95] exploited a hierarchical B-frame structure to construct multiple descriptions. In contrast, in our work only I- and P-frames are considered, which has the advantage of minimum decoding delay[1]—important for free viewpoint video streaming, where a user can interactively switch views in real-time as the video is played back. Moreover, [95] studied the single-view video scenario instead of free viewpoint, and in their context the focus is on reconstruction of the single-view video at higher quality, when multiple frame-subsampled versions of the same content are received. In contrast, in our MDC work we focus on how a lost description can be recovered by exploiting inter-view and temporal correlation in the received description.

2.9 Temporal Super Resolution

temporal super-resolution (TSR) interpolates frame $\mathbf{x}_t$ at time t using its two temporal neighbors $\mathbf{x}_{t-1}$ and $\mathbf{x}_{t+1}$, by exploiting their temporal correlation. TSR is used in applications such as temporal down-sampling for low-bitrate video streaming [96]. The most common method for TSR remains block-based *motion estimation* (ME) and *motion compensation* (MC). For example, [97] proposed to perform forward ME from frame $\mathbf{x}_{t-1}$ to $\mathbf{x}_{t+1}$ and backward ME from $\mathbf{x}_{t+1}$ to $\mathbf{x}_{t-1}$, and then selects the better option. The shortcoming of [97] is that it cannot guarantee at least one candidate per missing pixel in the target frame. In our MDC scheme, because DIBR does not provide inter-view recovery candidates for all missing pixels (due to disocclusion, out-of-view problems, etc.), we must construct a temporal recovery candidate per-pixel in the missing frame. We thus elect the bidirectional ME (BME) approach taken in [98], described in Section 5.3. Note, however, that we perform sub-block ME and *overlapped motion compensation* (OMC) using the available depth information, which is not considered in [92].

[1] A B-frame is correctly decoded only after the past and future predicted frames are correctly decoded, resulting in decoding delay.

Exploiting spatiotemporal correlation in the context of stereoscopic video coding using distributed source coding principles have been examined in [99–101], where side information video frames are generated by combining disparity-based and temporal-based data recovery.

Chapter 3

Cooperative Peer Recovery for Multi-view Video Multicast

This chapter introduces how we optimized the decision process for individual peers during CPR for the recovery of multi-view video content. In particular, for each available transmission opportunity, a peer decides—using Markov decision process as a mathematical formalism—whether to transmit, and if so, how the CPR packet should be encoded using SNC. A loss-stricken peer can then either recover using received CPR packets of the same view, or using packets of two adjacent views and subsequent view interpolation via image-based rendering.

3.1 Introduction

Packet losses over wireless channels happen due to the shadowing, channel fading and inter-symbol interference. A conventional approach [73] to tackle the unavoidable packet losses and protect source pacekts is to use FEC packets. However, deploying sufficient amount of FEC even for the worst-channel peer in the multicast group translate to a large overhead, leaving preciously few bits out of a finite WWAN transmission budget for source coding to fight quantization noise, resulting in poor video quality. One alternative solution to alleviate the wireless packet loss problem is CPR [65]. CPR, exploiting the "uncorrelatedness" of neighboring peers' channels to the same WWAN source (hence

unlikely for all peers to suffer bad channel fades at the same time), calls for peers to locally exchange received WWAN packets via a secondary network such as an ad hoc WLAN. Further, it was shown [65] that instead of exchanging raw received WWAN packets from the server, a peer can first encode a SNC repair packet using received source packets before sharing the encoded repair packet to further improve packet recovery.

In this chapter, we proposed to locally exchange received WWAN packets via a secondary network like ad hoc WLAN to alleviate individual WWAN packet losses. In particular, we optimized the decision process for individual peers during CPR for recovery of multi-view video content in IMVS. In particular, for each available transmission opportunity, a peer decides—using Markov decision process as a mathematical formalism—whether to transmit, and if so, how the CPR packet should be encoded using SNC. A loss-stricken peer can then either recover using received CPR packets of the same view, or using packets of two adjacent views and subsequent view interpolation via *image-based rendering* (IBR) [102]. The proposed MDP is fully distributed and peer-adaptive, so that state transition probabilities in the MDP can be appropriately estimated based on observed aggregate behavior of neighboring peers. Experiments show that decisions made using our proposed MDP outperforms decisions made by a random scheme by at least 1.8dB in PSNR in received video quality.

3.2 System Overview

3.2.1 WWAN Multiview Video Multicast with CPR

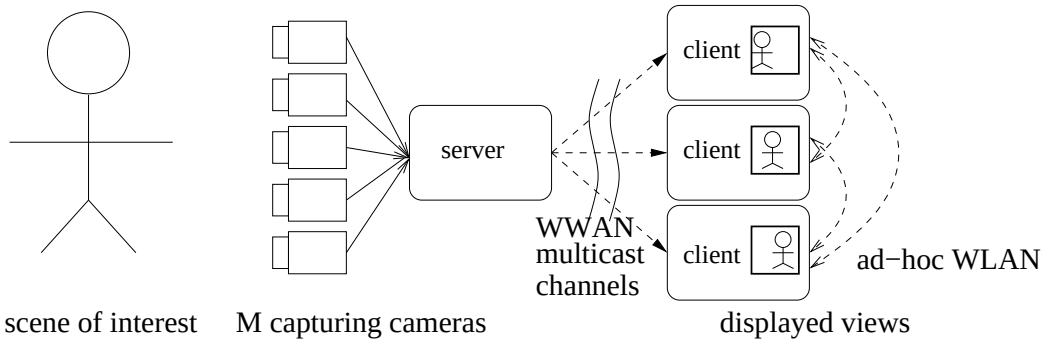


FIGURE 3.1: Overview of WWAN Multiview Video Multicast System

The components of our proposed WWAN multiview video multicast system, shown in Fig. 3.1, are as follows. M cameras in a one-dimensional array capture a scene of interest

from different viewpoints. A server compresses the M different views into M individual streams and transmits them, synchronized in time, in different WWAN multicast channels such as *Multimedia Broadcast/Multicast Service* (MBMS) in 3GPP [103].

A peer interested in a particular view subscribes to the corresponding channel and can switch to an adjacent view interactively by switching multicast channels every T/FPS seconds (an *epoch*), where FPS is the playback speed of the video in frames per second.

Peers are also connected to their neighbors via ad hoc WLAN, providing a secondary network for potential CPR frame recovery by relaying each peer's own received packets. If a neighbor to a peer is watching the same view v, then she can assist in frame recovery of same view v by relaying her own received packets via CPR. If neighbor is watching a different view v', then she can still help to partially recover lost frames via view interpolation since the views are correlated.

The WWAN server first multicasts one epoch worth of video to peers. During WWAN transmission of the next video epoch, cooperative peers will exchange received packets or decoded frames of the first video epoch. When the server multicasts the third epoch, peers exchange video packets in the second epoch, and video in the first epoch is decoded and displayed. View-switching delay is hence two epochs $2T/FPS$.

3.2.2 Source and Network model

3.2.2.1 Source Model

We assume M views are captured by closely spaced cameras so that strong inter-view correlation exists among them. We assume a GOP of a given view, transmitted in one epoch duration T/FPS, is composed of a leading I-frame followed by $T-1$ P-frames; each P-frame F_k is differentially coded using the previous frame F_{k-1} as predictor. A frame F_k is correctly decoded if it is correctly received *and* its predictor (if any) is correctly decoded. A correctly decoded frame F_k reduces visual distortion by d_k. Each frame F_k is divided into r_k packets, $p_{k,1}, \ldots, p_{k,r_k}$, for transmission. The total number of source packets in a GOP is then $R = \sum_{k=1}^{T} r_k$.

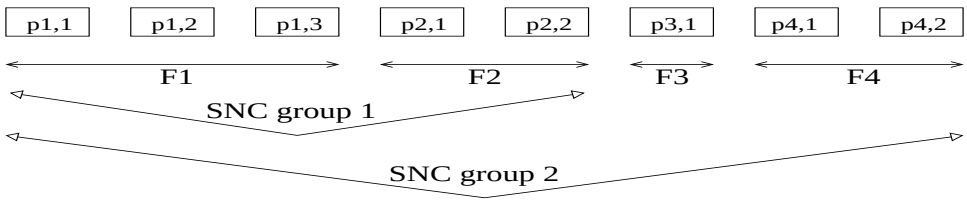


FIGURE 3.2: Example of structured network coding (SNC) for a 4-frame GOP and two SNC groups: $\Theta_1 = \{F_1, F_2\}$, $\Theta_2 = \{F_1, \ldots, F_4\}$.

3.2.2.2 WWAN & Ad hoc WLAN Channel Models

Burst packet losses are common in wireless links due to shadowing, slow path fading, and interference [23]. To model WWAN packet losses, we use the *Gilbert-Elliot* (GE) model with *independent & identically distributed* (iid) packet loss probabilities g and b for each of 'good' and 'bad' state, and state transition probabilities p and q to move between states, as illustrated in Fig. 3.3.

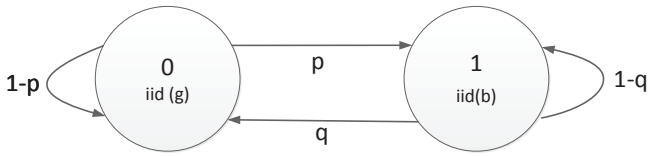


FIGURE 3.3: Gilbert-Elliott packet loss model: transitions between the two states (good - 0 and bad - 1) with probabilities p and q. The packet loss probabilities in good and bad states are g and b, respectively.

We assume packets are lost in the ad hoc WLAN due to in-air collision from hidden terminals. Denote by $\gamma_{n,m}$ the probability of a transmitted packet by peer n being lost to a one-hop receiving peer m. For simplicity, we assume they are known and unchanging for the duration of a repair epoch.

Transmissions in the ad hoc WLAN are scheduled according to the 802.11 MAC layer protocol. When the right to send is granted by the MAC layer, a TO becomes available to the peer. The peer then decides whether to send and what packet to send during this TO. We assume an *acknowledgement* (ACK) control packet is broadcasted after receiving a CPR packet and is transmitted without loss.

3.2.3 Network Coding for CPR

In order to improve CPR packet recovery efficiency, it has been proposed [65] that each peer should encode received packets into a coded packet using NC [104] before

performing CPR exchange. More specifically, at a particular TO for peer n, she has received set $\mathcal{G}_n$ of source packets from WWAN streaming source via WWAN and set $\mathcal{Q}_n$ of NC repair packets from neighboring peers via ad hoc WLAN. Peer n can NC-encode a CPR packet, q_n, as a randomized linear combination of packets in $\mathcal{G}_n$ and $\mathcal{Q}_n$:

$$q_n = \sum_{p_{i,j} \in \mathcal{G}_n} a_{i,j} p_{i,j} + \sum_{q_l \in \mathcal{Q}_n} b_l q_l \tag{3.1}$$

where $a_{i,j}$'s and b_l's are random coefficients for the received source and CPR packets, respectively. This approach is called *Unstructured Network Coding* (UNC). The advantage of UNC is that *any* set of R received *innovative*[1] packets—resulting in R equations and R unknowns—can lead to full recovery of all packets in the GOP. The shortcoming of UNC is that if a peer receives fewer than R innovative packets, then this peer cannot recover *any* source packets using the received CPR packets.

3.2.3.1 Structured Network Coding for CPR

To address UNC's shortcoming, one can impose structure in the random coefficients $a_{i,j}$'s and b_l's in (3.1) when encoding a CPR packet, so that partial recovery of important frames in the GOP at a peer when fewer than R innovative packets are received is possible. Specifically, we define X *SNC groups*, $\Theta_1, \ldots, \Theta_X$, where each Θ_x covers a different subset of frames in the GOP and $\Theta_1 \subset \ldots \subset \Theta_X$. Θ_1 is the most important SNC group, followed by Θ_2, etc. Corresponding to each SNC group Θ_x is a *SNC packet type* x. Further, let $g(j)$ be the index of the smallest SNC group that contains frame F_j.

With the definitions above, a SNC packet $q_n(x)$ of type x can now be generated as follows:

$$q_n(x) = \sum_{p_{i,j} \in \mathcal{G}_n} \mathbf{1}(g(i) \leq x)\ a_{i,j} p_{i,j} + \sum_{q_l \in \mathcal{Q}_n} \mathbf{1}(\Phi(q_l) \leq x)\ b_l q_l, \tag{3.2}$$

where $\Phi(q_l)$ returns the SNC type of packet q_l, and $\mathbf{1}(\mathsf{c})$ evaluates to 1 if clause c is true, and 0 otherwise. In words, (3.2) states that a CPR packet $q_n(x)$ of type x is a random linear combination of received source packets of frames in SNC group Θ_x and received

[1] A new packet is innovative for a peer if it cannot be written as a linear combination of previously received packets by the peer.

CPR packets of type $\leq x$. Using (3.2) to generate CPR packets, a peer can now recover frames in SNC group Θ_x when $|\Theta_x| < R$ innovative packets of types $\leq x$ are received.

More specifically, we can define the necessary condition to NC-decode a SNC group Θ_x at a peer as follows. Let c_x be the sum of received source packets $p_{i,j}$'s such that $g(i) = x$, *and* received CPR packets of SNC type x. Let C_x be the number of source packets in SNC group x, i.e. $C_x = \sum_{F_k \in \Theta_x} r_k$. We can then define the number of *type x innovative packets* for SNC group Θ_x, I_x, recursively as follows:

$$\begin{aligned} I_1 &= \min(C_1, c_1) \\ I_x &= \min(C_x, c_x + I_{x-1}) \end{aligned} \tag{3.3}$$

(3.3) states that the number of type x innovative packets I_x is the smaller of i) C_x, and ii) c_x *plus* the number of type $x-1$ innovative packets I_{x-1}. A SNC group Θ_x is decodable only if $I_x = C_x$.

3.3 Formulation

We now address the packet selection problem when a peer is granted a TO by the MAC layer: should she send a CPR packet, and if so, which SNC type should the packet be NC-encoded in? We discuss how we solve this problem via a carefully constructed MDP with finite horizon in this section.

3.3.1 Preliminaries

We assume that at the start of the CPR repair epoch of T/FPS seconds, each peer n already knows who are her 1-hop neighbors, and who lost what source packets during WWAN transmission in the last epoch. Among her 1-hop neighbors, those who did not have all their source packets received in the last WWAN broadcast epoch are marked *target receivers*. At each MDP invocation (when a peer is granted a TO by the MAC layer), one peer m out of the pool of marked target receivers is selected (in a round robin fashion for fairness). The objective of the MDP is to maximize the expected distortion reduction of the selected target receiver m. At each TO, a peer can estimate

the number of TOs she has left until the end of the repair epoch based on the observed time intervals between previous consecutive TOs, and the amount of time remaining in the repair epoch. Let the estimated number of remaining TOs until the end of the repair epoch be H; hence H is also the finite horizon for the constructed MDP.

Finally, we assume that when a peer received a CPR packet from her neighbor, she immediately transmits a *rich* ACK packet, revealing her current state (the number of CPR packets she has received in each SNC type).

3.3.2 State & Action Space for MDP

In a nutshell, a MDP of finite horizon H is a H-level deep recursion, each level t is marked by its states s_t's and actions a_t's. Each state s_t represents the state of the target receiver m $t-1$ TOs into the future, and a_t's are the set of feasible actions that can be taken by the sender at that TO given receiver's state s_t. The solution of a MDP is a *policy* π that maps each s_t to an action a_t, i.e., $\pi : s_t \rightarrow a_t$.

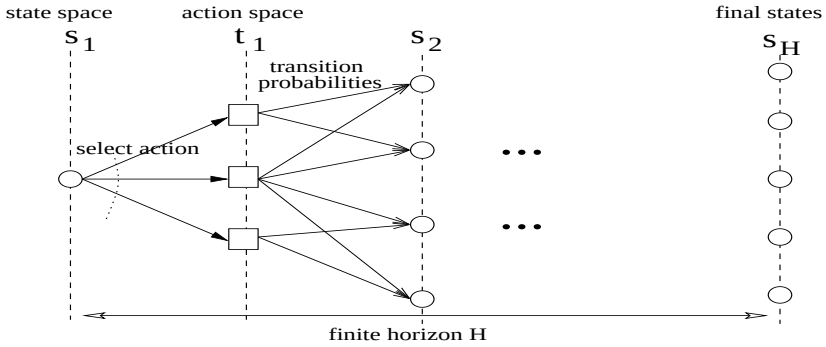


FIGURE 3.4: Example of Markov Decision Process

We first define states s_t's for our MDP construction for SNC packet selection. Let $s_t(I_1, ..., I_X, I_1^{'}, ..., I_X^{'}, I_1^{''}, ...I_X^{"})$ be a feasible state for target receiver m at TO t, where I_x is the number of type x innovative packets of the same view as the target receiver. $I_x^{'}$and $I_x^{''}$ are the number of type x innovative packets of the left and right adjacent views to the target receiver. Given $I_x \leq C_x$, the size of the state space is bounded by $O((\prod_{x=1}^{X} C_x)^3)$. In practice, the number of SNC groups X is small, hence the state space size is manageable.

Given each peer receives one view from the WWAN source, the action space for each sender is: i) no transmission ($a_t = 0$), and ii) transmission of CPR packet of type x

($a_t = x$) of the sender's selected view. A type x CPR packet will not be transmitted if there are already sufficient packets to decode SNC group x of that view at receiver. Thus, an action $a_t = x$ is feasible iff the following two conditions are satisfied:

1. There exists source packets $p_{i,j} \in \mathcal{G}_n$ such that $g(i) = x$ and/or $q_m \in \mathcal{Q}_n$ such that $\Phi(q_m) = x$.
2. $I_x < C_x$.

The first condition ensures there is new information pertaining to SNC group Θ_x that can be encoded, while the second condition ensures the encoded packet of type x is not unnecessary.

3.3.3 State Transition Probabilities for MDP

State transition probabilities is the likelihood of reaching state s_{t+1} in next TO $t+1$ given state s_t and action a_t at current TO t. Here, we arrive at the "distributed" component of the packet selection problem: the probability of arriving in state s_{t+1} depends not only on the action a_t taken by this peer n at this TO t, but also actions taken by other peers transmitting to the target receiver m during the time between TO t and TO $t+1$. However, given packet selection is done by individual peers in a distributed manner (as opposed to centralized manner), how can this peer know what actions will be taken by other transmitting peers in the future?

Here, we leverage on previous work in distributed MDP [78] that utilizes the notion of users' aggregate *behavior patterns* in normal-form games. The idea is to identify the patterns of users' tendencies to make decisions (rather than specific decisions), and then make prediction of users' future decisions based on these patterns. For our specific application, first, we assume the number of received packets of the same, left and right adjacent views at target receiver m from other transmitting peer(s) between two TOs are L, $L^{'}$ and $L^{''}$, respectively. These can be learned from target receiver m's ACK messages overheard between the sender's consecutive TOs.

For given L, $L^{'}$ or $L^{''}$ received packets of the same, left or right adjacent view from other sender(s), we identify the corresponding SNC packet types by considering the following two aggregate behavior patterns. First is *pessimistic* and assumes the aggregate of other

transmitting peers of this view always transmit innovative packets of the smallest SNC groups possible. This pattern is pessimistic because it seeks immediate benefit as quickly as possible, regardless of the number of TOs available in the finite horizon of H levels. The second is *optimistic* and assumes the aggregate of other transmitting peer(s) will always transmit innovative packets of the largest SNC group Θ_X. This is optimistic because it assumes R innovative packets for the largest SNC group Θ_X will be received by the target receiver m, so that the entire GOP can be correctly decoded.

Let λ, $\lambda^{'}$ and $\lambda^{''}$ be the probabilities that a peer uses pessimistic pattern when selecting a SNC packet type of the same, left and right adjacent views, respectively. The probability that L packets of the same view are divided into k packets of pessimistic and $L-k$ packets of optimistic patterns is:

$$P(k|L) = \left(\begin{array}{c} L \\ k \end{array} \right) \lambda^k (1-\lambda)^{L-k} \tag{3.4}$$

(3.4) can also be used for the probability $P(k|L^{'})$ or $P(k|L^{''})$ that $L^{'}$ or $L^{''}$ packets are divided into k pessimistic and $L^{'}-k$ or $L^{''}-k$ optimistic packets, with $\lambda^{'}$ or $\lambda^{''}$ replacing λ in (3.4).

Initially, we do not know which pattern is more likely, and we assume they are equally likely with probability 1/2. The probabilities of pessimistic pattern for the three views, λ, $\lambda^{'}$ and $\lambda^{''}$, will be learned from ACK messages from target receiver m as the CPR process progresses, however.

To derive state transition probabilities, we first define G to be a mapping function that, given state s_t, maps k pessimistic and $L-k$ optimistic packets of view v ($v \in \{s, l, r\}$ to denote same, left and right adjacent view of the receiver) into corresponding SNC packet difference vector $\Delta = \{\delta_1, \ldots, \delta_X\}$, i.e. $G(s_t, v) : (k, L-k) \longrightarrow \Delta$. In general, there can be multiple pessimistic / optimistic combinations $(k, L-k)$'s that map to the same Δ. Let $\Delta = s_{t+1}(I_1, \ldots, I_X) - s_t(I_1, \ldots, I_X)$ be the SNC type-by-type packet count difference between state s_{t+1} and s_t for the same view. Similarly, let $\Delta^{'}$ and $\Delta^{''}$ be the type-by-type packet count difference between s_{t+1} and s_t for the left and right adjacent views. Further, let $\Delta^{+} = s_{t+1} - s_t - \{a_t\}$, which is like Δ, but also accounting for the CPR packet of the same view transmitted by this sender's current action $a_t = x$.

Assuming action of the same view $a_t = x$, $x \geq 1$, the state transition probability $P(s_{t+1}|s_t, a_t = x)$ can now be written:

$$\gamma_{n,m}\left(\sum_{k|(k,L-k)\overset{G(s_t,s)}{\longrightarrow}\Delta} P(k|L)\right)\left(\sum_{k|(k,L'-k)\overset{G(s_t,l)}{\longrightarrow}\Delta'} P(k|L')\right)\left(\sum_{k|(k,L''-k)\overset{G(s_t,r)}{\longrightarrow}\Delta''} P(k|L'')\right) \quad (3.5)$$
$$+(1-\gamma_{n,m})\left(\sum_{k|(k,L-k)\overset{G(s_t,s)}{\longrightarrow}\Delta^+} P(k|L)\right)\left(\sum_{k|(k,L'-k)\overset{G(s_t,l)}{\longrightarrow}\Delta'} P(k|L')\right)\left(\sum_{k|(k,L''-k)\overset{G(s_t,r)}{\longrightarrow}\Delta''} P(k|L'')\right)$$

(3.5) states that to arrive at state s_{t+1}, the L, $L^{'}$, $L^{''}$ packets received from other senders of the same, left and right adjacent views must lead to packet difference vectors Δ, $\Delta^{'}$ and $\Delta^{''}$ if transmitted packet of the same view by this peer n is lost (with probability $\gamma_{n,m}$), or lead to packet difference vectors Δ^{+}, $\Delta^{'}$ and $\Delta^{''}$ if transmitted packet by this peer n is delivered successfully (with probability $1-\gamma_{n,m}$). Similar expression can be derived for the state transition probability if the sender is transmitting CPR packets of left or right adjacent view.

3.3.4 Finding Optimal Policy for MDP

The optimal policy π^* is one that leads to the minimum expected distortion (maximum distortion reduction) at the end of the H-level horizon. More specifically, denote by $\pi_t^*(s_t)$ the maximum expected distortion reduction at the end of H-level horizon, given state at TO t is s_t. $\pi_t^*(s_t)$ can be defined recursively: a chosen action a_t at TO t leads to state s_{t+1} with probability $P(s_{t+1}|s_t, a_t)$, as defined in Section 3.3.3, and assuming optimal policy $\pi_{t+1}^*(s_{t+1})$ is performed at TO $t+1$, we have expected benefit $P(s_{t+1}|s_t, a_t)\pi_{t+1}^*(s_{t+1})$. $\pi_t^*(s_t)$ exhaustively searches for the optimal action a_t^* given state s_t:

$$\pi_t^*(s_t) = \begin{cases} \max_{a_t} \sum_{s_{t+1}} P(s_{t+1}|s_t, a_t)\pi_{t+1}^*(s_{t+1}) & \text{if } t < H \\ d(s_t) & \text{o.w.} \end{cases} \quad (3.6)$$

If state s_t is at the end of the H-level horizon, then no more actions can be taken, and $\pi_t^*(s_t)$ in (4.18) simply returns the distortion reduction $d(s_t)$ given state s_t. $d(s_t)$ is defined as follows:

$$\sum_{k=1}^{T} d_k \, \mathbf{1}\left(\bigcup_{x=g(k)}^{X} (I_x = C_x)\right) + d'_k \, \mathbf{1}\left(\bigcap_{x=g(k)}^{X} (I_x < C_x)\right) \mathbf{1}\left(\bigcup_{x=g(k)}^{X} (I'_x = C_x) \cap (I''_x = C_x)\right) \tag{3.7}$$

(3.7) states that frame F_k can be recovered from CPR packets of the same view (with distortion reduction d_k), if one of SNC groups $\Theta_{g(k)}, \ldots, \Theta_X$ can be correctly NC-decoded. If all SNC groups $\Theta_{g(k)}, \ldots, \Theta_X$ of the same view fail, then F_k can still be partially recovered (with distortion reduction $d'_k < d_k$) from CPR packets of adjacent views, if one of SNC groups $\Theta_{g(k)}, \ldots, \Theta_X$ of *both* left and right adjacent views can be NC-decoded. (4.18) can be solved efficiently using dynamic programming as done in [78]. Note that the complexity is determined by the finite horizon H and the size of the state space.

3.4 Experimentation

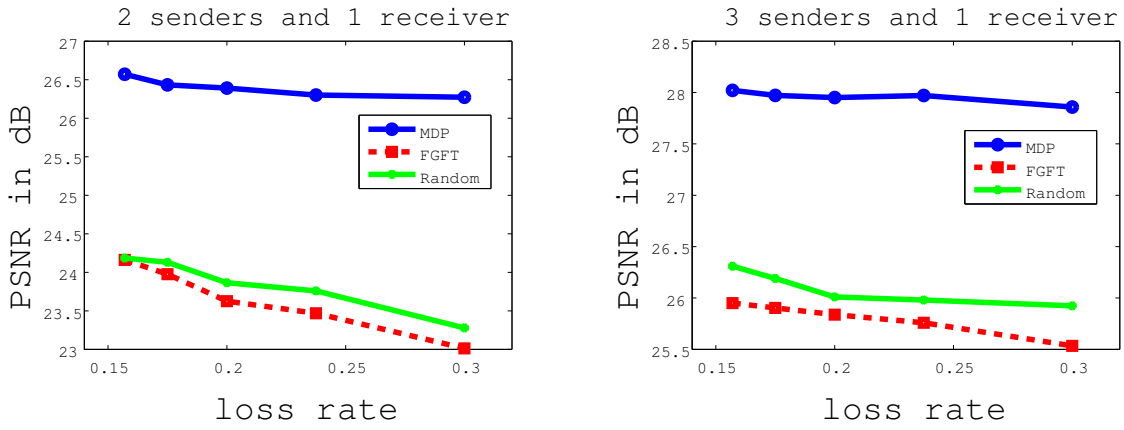
3.4.1 Experimental Setup

For our experiments, we used three views of the standard multiview video test sequence `akko` at 640×480 resolution. Each client randomly chose one view for each transmission epoch. A GOP was a leading I-frame plus 9 P-frames. We fixed the quantization parameters for I- and P-frames so that the resulting visual quality in PSNR was roughly 32.5dB. *Maximum transport unit* (MTU) were assumed to be 1250 bytes, bandwidths of the WWAN broadcast and WLAN were assumed to be 400 and 260kbps, respectively. We assumed there were only two globally defined SNC groups: $\Theta_1 = \{F_1, F_2\}$ and $\Theta_2 = \{F_1, \ldots, F_{10}\}$.

3.4.2 Experimental Results

We focused on the case when there was one target receiver and more than one neighboring peer helping to recover lost packets. For comparison, `FGFT` instructs each peer to send CPR packets of SNC group Θ_x only after sending sufficient number of type $x-1$ packets. `random` instructs each peer to randomly choose an SNC type for encoding for each available TO.

In Fig. 3.5(a), there were two senders and one receiver, and WWAN loss rate (x-axis) was changed by varying loss rate b. y-axis is the resulting video quality in PSNR in dB. We



(a) $\alpha = 0.1$, $a = 0.04$, $\beta = 0.85$

(b) $\alpha = 0.01$, $a = 0.05$, $\beta = 0.8$

FIGURE 3.5: PSNR comparison of proposed MDP-based decision making and two other schemes for a three/four-node network topology.

can see that `MDP` outperformed `random` and `FGFT` by 2.5dB and 2.5dB, respectively. The main reason for the improvement is that `MDP` maximized expected distortion reduction for the entire repair epoch, taking other transmitting peers' actions into consideration through observed behavior patterns. That means peers can avoid sending duplicate packets. In the case when WLAN loss rates were set to 0.05 and 0.06, one sender could also learn that the other sender had lower WLAN loss rate than herself, and subsequently elected not to transmit during available TOs for better overall system performance. In Fig. 3.5(b), we change the WWAN GE model parameters to be $\alpha = 0.01$, $a = 0.05$, $\beta = 0.8$ for a four-peer topology. WLAN loss rate was set to be 0.05. We see here that `MDP` outperformed `random` and `FGFT` by 1.8dB and 2.1dB, respectively.

3.5 Summary

In this chapter, we introduced the *cooperative peer recovery for multi-view video multicast*. We optimized the decision process for individual peers during CPR for recovery of multiview video content in IMVS. Based on the MDP, at each TO a peer decides whether to transmit, and if so, how the CPR packet should be encoded using SNC. It is fully distributed, computationally scalable and peer-adaptive. Experiments showed that decisions made using our proposed MDP improved the received video quality in typical network scenario by at least 1.8dB in PSNR over the random scheme.

Chapter 4

Unified Distributed Source Coding for Interactive Multi-view Video Streaming

This chapter introduces how we provided users the ability to switch to other views and evade the error propagation at the same time. We first designed a new unified distributed source coding (uDSC) frame for periodic insertion into the multi-view frame structure. After inserting uDSC-frames into the coding structure, we scheduled packets for network transmission in a rate-distortion optimal manner for both wireless multicast and wired unicast streaming scenarios.

4.1 Introduction

Multi-view video [105] is recorded by an array of closely spaced cameras from different viewing angles; i.e., at video frame rate, images of the same scene are captured synchronously by M cameras from different viewpoints. During video playback, although a viewer is typically observing only one of M captured views at a time on a conventional 2D display, multi-view video offers a viewer the freedom to switch to a desired neighboring captured view periodically (once every T frames). This *inter-view interactivity*

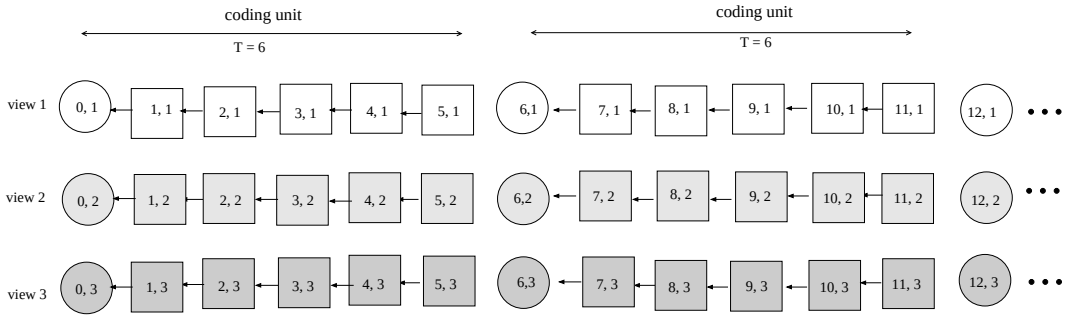


FIGURE 4.1: Example of coding structure with periodic I-frames inserted for view-switching and mitigating error propagation, for $M = 3$ views and coding unit size $T = 6$. Circles and squares are I-, P-frames. Each frame $F_{i,v}$ is labeled by its time index i and view v.

(random access in view) can be triggered, for example, by viewer's detected head movements, enhancing a depth perception in the observed 3D scene via *motion parallax* [106]. Inter-view interactivity is typically invoked by users on the order of tens to hundreds of milliseconds, contrasting with *temporal interactivity* (random access in time) available in conventional single-view video (i.e., fast forward and rewind of the video in time), which is typically invoked on the order of tens of seconds or minutes.

Multi-view videos with captured views from up to one hundred cameras [107] are clearly much larger in size than single-view videos, even if state-of-the-art multi-view video coding (MVC) techniques [52, 53] are employed to exploit inter-view and temporal correlation to reduce the overall bit-rate. In an interactive multi-view video streaming (IMVS) scenario [55], in order to minimize streaming rate, only the desired views currently requested by clients are transmitted by servers. In IMVS multicast, that means a client subscribes to only one multicast channel that distributes his current desired view, and re-subscribes to a different channel that distributes a neighboring view during a view-switch. In IMVS unicast, it means the server streams only one single-view video to a client that corresponds to his most recently chosen viewpoint. In either case, the technical challenge remains the same: how to flexibly pre-encode the multi-view video content *a priori*, so that at streaming time, a server can transmit compact versions of the content that facilitate clients' periodic view-switching? Previous IMVS works [40, 41, 55] focus on efficient compression techniques that facilitate interactive view-switching without sacrificing coding efficiency in differentially coded video (such as video compression standards H.263 [108] and H.264 [24]).

In this chapter, given unavoidable packet losses in best-effort communication networks,

we address *in addition the loss-resilient aspect* during network streaming: at encoding time, how to design efficient coding tools and optimize frame structure, so that at streaming time, pre-encoded structure can facilitate interactive view-switching *and* contain error propagation in differentially coded video in the face of packet losses, at the same time? One naïve way to solve both the interactive view-switching problem and the error propagation problem is to insert an independently coded I-frame for every view v at multiples of T frames. Thus, a client can switch from P-frame of view u to a desired neighboring view v at *view-switching boundary* nT, and have I-frame of instant nT and view v correctly decoded. Further, error propagation due to a loss-corrupted differentially coded P-frame will terminate at the next view-switching boundary. See Fig. 4.1 for an example IMVS frame structure using periodic I-frames (drawn as circles) for view-switching and error resiliency. However, independently coded I-frame can be up to 10 times larger than differentially coded P-frame, and given application-specific view-switching period T can be quite small (e.g., T was set to 5 frames in [55]), frequent insertion of large I-frames into the structure is not practical.

Towards a more efficient solution to both the interactive view-switching and error propagation problems, we first provide a novel alternative to large I-frames: uDSC frame. Our work leverages on previous work in DSC [29, 85], where the key idea is to treat the *source coding* problem with uncertainty as a *channel coding* problem instead. In our IMVS scenario, there are two uncertainties when coding an uDSC frame $W_{nT,v}$ of instant nT and view v: i) which frame $P_{nT-1,u}$ will be available during IMVS streaming at the decoder buffer as side information (SI) for decoding of $W_{nT,v}$ when a client switches from neighboring view u to v; and, ii) how many and which preceding P-frames will be lost by the time uDSC frame $W_{nT,v}$ is decoded. We first interpret and model each uncertainty as "channel noise" in the SI for decoding of uDSC frame $W_{nT,v}$.

We then statistically bound the maximum noise level for the two uncertainties, called *view-switching noise* and *error propagation noise*. Finally, we apply channel code of sufficient strength to overcome the *larger* of the two noise terms in the SI for reconstruction of DSC frame $W_{nT,v}$. Thus, the same constructed uDSC frame can facilitate view-switching *and* halt error propagation, unifying capabilities of previous DSC proposals that only facilitates view-switching [85] or only halts error propagation [29].

Inserting uDSC frames into a coding structure are effective only if accompanied packet

scheduling is optimized to exploit uDSC frames' unique loss-resilient characteristics. Thus, after pre-encoding and inserting uDSC frames into IMVS coding structures, at streaming time, we optimize frame transmission for two different network settings: i) IMVS multicast over wireless channels with Gilbert-Elliott packet losses, and ii) IMVS unicast over wired networks with tight playback deadline and iid packet losses. For IMVS wireless multicast, we optimize packetization, UEP and packet reordering for each GOP, so that important packets containing motion information necessary for uDSC frame decoding can be protected more heavily than packets containing prediction residual information. This ensures uDSC frames can be correctly decoded with high probability to halt possible error propagation. For IMVS wired unicast, we optimize packet scheduling of a group of T frames with uDSC frames inserted using MDP, so that the expected visual distortion is minimized.

4.2 Interactive Multiview Video Streaming System

4.2.1 IMVS System Overview

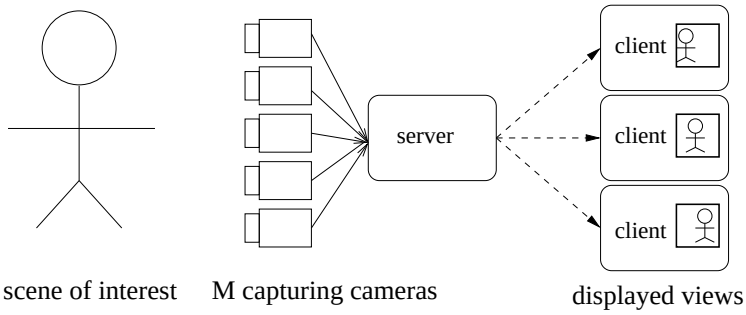


FIGURE 4.2: Overview of core components in a general IMVS system.

A general IMVS system is composed of the following basic components as illustrated in Fig. 4.2. A scene of interest is captured simultaneously by a 1D array of M closely spaced cameras from different viewing angles. The number of capturing cameras can be large; up to 100 time-synchronized cameras were used in [107]. The multi-view videos from M viewpoints are pre-encoded into a frame structure (to be discussed in Section 4.3) and stored *a priori* in a server—targeting store-and-playback streaming scenarios that require no real-time video encoding. At stream time, a client requests and watches a single view v, $v \in \{1, \ldots, M\}$, at a time on a conventional 2D display. Further, the client can periodically switch to a neighboring view u, $u = \{\max(1, v-1), v, \min(M, v+1)\}$,

every T-frame interval. Thus, the frame structure must be constructed to facilitate view-switches to neighboring views every T frames. This *interactive streaming* paradigm [86], where a client interactively navigates subsets of available high-dimensional data for observation, is in stark contrast with *non-interactive streaming* like single-view video broadcast, where clients passively consume every stored and transmitted media bit without data selection. This system model is similar to the system introduced in Chapter 3, but the network channels here could be wired or wireless and there are no links connecting users.

The structure must also provide error resiliency, so that if a frame is lost during transmission, error propagation (decoding error in the subsequent differentially coded frames) can be halted in a timely manner. *Designing and optimizing transmission of an IMVS frame structure that facilitates interactive view-switching and provides error resiliency for a particular network scenario is the main challenge.*

4.2.1.1 IMVS Wireless Multicast

In IMVS wireless multicast, each captured video view v is transmitted in a separate multicast channel like Multimedia Broadcast/Multicast Service (MBMS) in 3GPP [103]. A client interested in a view v only subscribes to a single multicast channel that transmits view v. When switching to a neighboring view u after T frames, a client can re-subscribe to a different multicast channel that transmits view u. Due to the broadcast nature of the wireless medium, multicasting a video view v means a single allocated wireless channel can satisfy multiple clients requesting the same view at the same time, resulting in efficiency of wireless resource allocation.

However, because of the well known NAK (*negative acknowledgment*) implosion problem in large group multicast [109]—the undesirable situation where the server is overwhelmed by a large number of requests for individual packet retransmission from multiple clients—typical loss protection strategies for multicast employs FEC like Reed-Solomon codes [110] or network coding [111] to pro-actively protect source packets before packet losses happen. We discuss how FEC can be optimally applied to protect the IMVS structure with uDSC frames in wireless multicast in Section 4.4.

4.2.1.2 IMVS Wired Unicast

In contrast, in IMVS wired unicast the server can tailor transmission of a single video view v for each client. Because the aforementioned NAK implosion problem does not applied to unicast, ARQ [112], where a lost packet is detected and reported from client to server to request a packet retransmission, is more often used. We optimize packet scheduling via MDP for an IMVS structure with uDSC frames in wired unicast in Section 4.5.

4.2.2 Packet Loss Models

We use two models to model packet losses in IMVS wireless multicast and wired unicast. For wireless multicast, we use the GE model, as introduced in Chapter 3. GE model has iid packet loss probabilities g and b for each of *good* and *bad* state, and state transition probabilities p and q to move between states.

Fore wired unicast, we use the simple iid packet loss model with packet loss parameter β.

4.3 Frame Structure for IMVS

We first overview different frame types for coding of individual video frames. We then design coding structures using the discussed frame types for wireless IMVS multicast and wired IMVS unicast.

4.3.1 Distributed Source Coding Frames for IMVS

We first overview conventional I- and P-frames used in video coding standards like H.263 [108] and H.264 [24]. We then discuss *Drift-Elimination* (DE) DSC frame and *Multiple-Predictor* (MP) DSC frame that are useful in an IMVS scenario. We create our proposed uDSC frame by combining the functionalities of DE-DSC and MP-DSC into a single frame.

4.3.1.1 Conventional I- and P-frames

I-frame, denoted as $I_{i,v}$ for frame at instant i and view v, is an *intra*-coded frame and can be decoded independently from other frames. *P-frame*, denoted as $P_{i,v}$, is *inter*-coded via motion and/or disparity compensation. In other words, using another encoded frame $F_{j,u}$ as predictor, only the *frame differential* $\Delta_{i,v}$ between target $P_{i,v}$ and predictor $F_{j,u}$—block-by-block *motion vectors* (MVs) and quantized motion prediction residuals—are encoded [108], resulting in a frame size much smaller than an I-frame. However, correct decoding of a P-frame $P_{i,v}$ requires first the correct decoding of predictor $F_{j,u}$ at the decoder. We will assume correct decoding of a frame $F_{i,v}$ leads to a reduction in signal distortion by $d_{i,v}$.

4.3.1.2 Drift-Elimination DSC Frames

DE-DSC frame [29], denoted as $W^1_{i,v}$, is designed to halt error propagation (also known as *coding drift*) due to *prediction mismatch* between encoder and decoder at the DE-DSC frame boundary. Mismatch happens when there are irrecoverable packet losses in the transmission network, resulting first in an erred reconstructed frame $\hat{F}_{i,v}$ of instant i at decoder that is different from coded $F_{i,v}$ at encoder. Due to differential coding, subsequent reconstructed P-frames $\hat{P}_{j,v}$'s, $j > i$, in the prediction chain stemming from erred predictor $\hat{F}_{i,v}$ will also be incorrect, even if frame differentials $\Delta_{j,v}$'s are correctly delivered, resulting in error propagation. DE-DSC frame $W^1_{l,v}$ halts this error propagation—i.e., restore $\hat{F}_{l,v}$ at decoder back to encoded $F_{l,v}$ at encoder at a later instant l.

For implementation, we first assume motion prediction residuals of a given frame are block-by-block transformed using *Discrete Cosine Transform* (DCT), with the resulting DCT coefficients quantized. If the magnitude of reconstruction error ("channel noise") due to error propagation in different bit-planes of the quantized coefficients can be bounded statistically, then DE-DSC frame can deploy sufficient amount of channel codes (*low-density parity check* (LDPC) code was used in [29] and [85]) for each bit-plane to remove the noise, given the noise statistics of that bit-plane. We retain the assumption in [29] that MVs in frame differentials $\Delta_{j,v}$'s between two DE-DSC frames are correctly

delivered, and only prediction residuals can be lost during transmission; this assumption helps limit the noise level in bit-planes to a manageable amount.

More specifically, we generate a DE-DSC frame $W^1_{i,v}(k)$ capable of halting propagated error due to lost prediction residuals of *any* k or fewer preceding P-frames since the previous DE-DSC frame as follows. Encoding target for $W^1_{i,v}$ is the I-frame $I_{i,v}$. Using the last k frame differentials $\Delta_{j,v}$'s created during encoding, $j \in \{i-k, \ldots, i-1\}$, we reconstruct k preceding P-frames $\hat{P}_{j,v}$'s *using MVs only* (no motion prediction residuals) in frame differentials $\Delta_{j,v}$'s. Doing so, we claim that $\hat{P}_{i-1,v}$ represents the predictor with the largest possible channel noise, assuming k or fewer prediction residuals have been lost since the last DE-DSC frame. This is true because lost prediction residuals from a different set of k P-frames earlier in the sequence will in general result in a smaller noise term by instant $i-1$, since error propagation tends to dissipate slowly over time due to occasional intra-block coding [55]. Given worst-case P-frame $\hat{P}_{i-1,v}$, we first apply frame differential $\Delta_{i,v}$ (both MVs and prediction residuals) for target DE-DSC frame, and then determine the sufficient amount of LDPC code needed to overcome the observed noise to recover target $I_{i,v}$. This LDPC code is the additional encoded bits for $W^1_{i,v}$ (beyond frame differential $\Delta_{i,v}$) in order to reconstruct target I-frame $I_{i,v}$ given noisy predictor (side information) $\hat{P}_{i-1,v}$.

4.3.1.3 Multi-Predictor DSC Frames

Multi-Predictor DSC (MP-DSC) frame [85], denoted as $W^2_{i,v}$, generalizes the single-predictor motion compensation paradigm in P-frame by employing *multiple* predictors at encoder. At decoder, only *one* in the encoder set of predictors needs to be available for the MP-DSC frame to be correctly decoded. For IMVS, we use MP-DSC frames for view-switching; for example, MP-DSC frame $W^2_{i,v}$ can be encoded using predictor frames $F_{i-1,u}$'s of previous instant, where view $u \in \{\max(1, v-1), \ldots, \min(M, v+1)\}$. A client of view u can thus switch to view v and decode frame $W^2_{i,v}$ correctly, using $F_{i-1,u}$ in his buffer as predictor.

For implementation, MP-DSC frames can be encoded similarly to DE-DSC frames. To overcome the uncertainty of which predictor will be available at decoder, a MP-DSC frame first encodes multiple sets of MVs, one for each predictor frame. Then, the resulting quantized DCT coefficients of the prediction residual for each predictor are

compared against the coefficients of the target frame $I_{i,v}$ to compute the noise statistics in each coefficient bit-plane. Appropriate amount of LDPC codes are then deployed in each bit-plane to overcome the *largest* noise of all prediction residuals for that plane [85].

Complexity of encoding a MP-DSC frame is only roughly k times as complex as a P-frame, where k is the number of predictor frames (due to repeated motion compensation performed for different predictor frames). Decoding of a DSC frame is only slightly more complex than a conventional P-frame (due to additional channel decoding of LDPC in different bit-planes).

We now encode MP-DSC frame $W^2_{i,v}$ so that it can also halt error propagation in the same view, as done in DE-DSC; new frame will be called uDSC frame $W^u_{i,v}$. In addition to the noise statistics of different prediction residuals from multiple predictors, we consider the computed noise statistics for a DE-DSC frame $W^1_{i,v}(k)$ of the same view also when deciding the amount of LDPC code used for each bit-plane. Note that doing so means that *the overhead of a uDSC frame is not the sum of overheads from both a DE-DSC and a MP-DSC frame, but only the larger of the two.* This is the key in creating a coding-efficient uDSC frame. However, this also mean a client cannot halt error propagation *and* switch to a neighboring view at the same time and have statistical guarantee that the frame will be correctly decoded. Assuming view-switching and irrecoverable packet loss are independent events and the probability of irrecoverable packet loss is relatively small (given suitable FEC or ARQ has been applied for loss protection), then the probability of a client requesting a view-switch while experiencing irrecoverable packet losses is tolerably small.

4.3.2 Frame Structure for Wireless IMVS Multicast

We now design a frame structure for wireless IMVS multicast using the previously discussed frame types as building blocks. Moreover, we optimize packetization and packet reordering to mitigate the adverse effects of packet losses in a GE loss model.

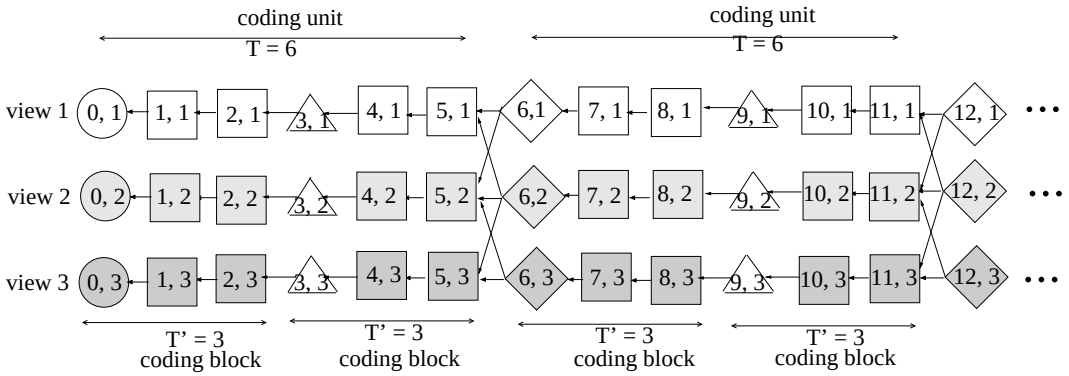


FIGURE 4.3: Example of proposed coding structure for $M = 3$ views and coding block size $T' = 3$, coding unit size $T = 6$. Circles, squares, triangles and diamonds are I-, P-, DE-DSC and uDSC frames. Each frame $F_{i,v}$ is labeled by its time index i and view v.

4.3.2.1 Constructing Frame Structure for IMVS Multicast

Given an application-specific requirement of a T-frame view-switching period, we consider coding of a GOP of τT frames, $\tau \in \mathcal{Z}^+$ (i.e., user can perform up to $\tau - 1$ view-switches in a GOP). A segment of T consecutive frames in a single view v is called a *coding unit* $U_{i,v}$, $i \in \{0, \ldots, \tau - 1\}$. Within coding unit $U_{i,v}$, T/T' *coding blocks* $L_{i,v}(j)$'s of T' frames each, $T' < T$, are coded as follows. We first encode a starting uDSC frame $W^u_{iT,v}(k)$—I-frame $I_{0,v}$ if it is the first coding block of the first coding unit—with $T' - 1$ trailing P-frames $P_{iT+l,v}$'s, $1 \leq l \leq T' - 1$, each motion-compensated using previous frame $F_{iT+l-1,v}$ as predictor, into the first coding block $L_{i,v}(1)$. We then encode a DE-DSC frame $W^1_{iT+T',v}(k)$, $k < T'$, followed again by $T' - 1$ trailing P-frames as the second coding unit $L_{i,v}(2)$. We repeat for T/T' coding blocks in the coding unit $U_{i,v}$. See Fig. 4.3 for an illustration.

If a DE-DSC $W^1_{i,v}(k)$ or uDSC $W^u_{i,v}(k)$ frame can be correctly decoded, it can mitigate error propagation due to earlier irrecoverable packet losses by serving as the perfectly reconstructed predictor for the following frames. Larger recoverability k results in a larger DE-DSC or uDSC frame, however. Optimization of structure parameters T' and k is discussed in Section 4.4.

4.3.2.2 Packetization of Encoded Bits in Coding Unit

We now discuss how we packetize encoded bits in coding unit $U_{i,v}$ into packets. As illustrated in Fig. 4.4, encoded bits in a P-frame are divided into *header*, *MVs* and

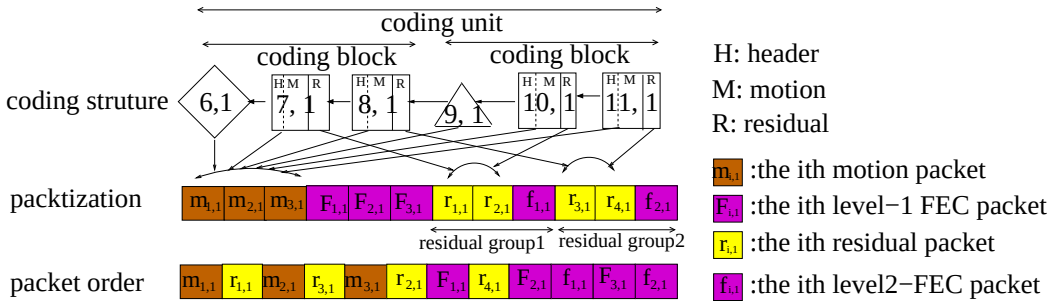


FIGURE 4.4: The three stages of the transmission scheme: i) encoding of captured images into frames in coding structure, ii) packetization of encoded bits into IP packets, and iii) ordering of generated packets for transmission. Motion, residual and FEC packets are indicated by red, yellow and blue, respectively.

prediction residuals. We group encoded bits of I-, DE-DSC and uDSC frames plus header and MVs of P-frames in coding unit $U_{i,v}$ together for packetization into $M_{i,v}$ *motion packets*, each of maximum size S bits (maximum transmission unit). These are the important packets that require more loss protection.

We next packetize encoded bits of prediction residuals in P-frames. Prediction residuals of $T'-1$ P-frames in a single coding block are packed into different *residual groups* that are channel-protected and transmitted separately, so that the likelihood of losing all P-frame prediction residuals in a block during a burst loss is small. Specifically, we gather residual bits of the lth frame of all coding blocks in a coding unit $U_{i,v}$, where $2 \leq l \leq T'$, into the lth residual group, which are then divided into packets of maximum size S bits[1]. For simplicity of discussion, let the number of *residual packets* in each residual group be $r_{i,v}$. The number of residual groups is $G = T' - 1$.

After packetization of the source bits in all blocks $L_{i,v}(j)$'s in a coding unit $U_{i,v}$, we next generate FEC packets to protect the motion and residual packets unequally. We first generate $\Phi_{i,v}$ level-1 FEC packets to protect $M_{i,v}$ motion packets in coding unit $U_{i,v}$. We then generate $\phi_{i,v}$ level-2 FEC packets to protect $r_{i,v}$ residual packets in *each* residual group. Assuming a perfect code (e.g., Reed-Solomon [110], network codes [111]) is used for FEC, $M_{i,v}$ motion packets can be correctly recovered if at least $M_{i,v}$ of $M_{i,v} + \Phi_{i,v}$ transmitted motion plus level-1 FEC packets are delivered. Similarly, each residual group can be correctly recovered if at least $r_{i,v}$ of $r_{i,v} + \phi_{i,v}$ residual plus level-2 FEC

[1]One can generalize grouping so that prediction residuals of ρ frames of each coding block $L_{i,v}(j)$ goes into one residual group. For simplicity, we will assume prediction residuals of a single frame from each coding block goes into a residual group.

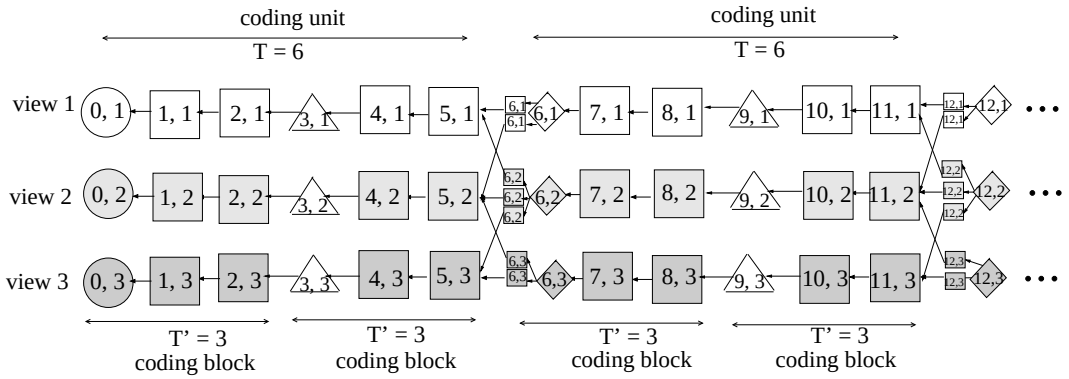


FIGURE 4.5: Example of proposed frame structure for wired IMVS unicast, for $M = 3$ views, coding block size $T' = 3$, and coding unit size $T = 6$. Circles, squares, triangles and diamonds denote I-, P-, DE-DSC and MP-DSC frames respectively. Each frame $F_{i,v}$ is labeled by its time index i and view v.

packets are delivered. We discuss how $\Phi_{i,v}$ and $\phi_{i,v}$ for each coding unit $U_{i,v}$ are selected optimally in the next section.

4.3.2.3 Packet Ordering for GE loss model

After packetizing encoded source bits of frames in a coding unit into packets and generating two levels of FEC packets, we now arrange the generated packets into a transmission order. Given the wireless loss model is a GE model, the guiding principle we use is *interleaving* [113]: space the motion information and prediction residuals apart so that the adverse effect of one traversal into the bad state in the GE model will be spread evenly across the coding unit.

Let ratio of the number of packets for motion plus level-1 FEC packets to residual plus level-2 FEC packets be $\lambda_M : \lambda_R$, where λ's are integers. We alternatively select λ_M motion & level-1 FEC packets and λ_R residual & level-2 FEC packets into a transmission order. When selecting residual packets, we select packets from different residual groups in a round-robin fashion. Doing so means that the spacings among motion packets and among residual packets are maximized. See Fig. 4.4 for an example illustration of packet reordering.

4.3.3 Frame Structure for Wired IMVS Unicast

Similarly, we now design a frame structure for wired IMVS unicast using different frame types as building blocks. An example frame structure is shown in Fig. 4.5. The

key difference from previous structure in Fig. 4.3 for wireless IMVS multicast is: at a view-switching point of instant nT and view v, there are first multiple P-frames $P_{nT,v}$'s, motion-compensated from frames $F_{nT-1,u}$'s of views u's, $u \in \{\max(1, v-1), \ldots, \min(M, v+1)\}$. Then, in addition, a uDSC frame $W^u_{nT,v}(k)$ of *the same time instant and view* is added. This $W^u_{nT,v}(k)$ is generated as follows. As done for DE-DSC frame, we first reconstruct k preceding P-frames $\hat{P}_{j,v}$'s, $j \in \{nT-k, \ldots, nT-1\}$, using MVs only in frame differentials $\Delta_{j,v}$'s. $\tilde{P}_{nT,v}$ is then constructed using $\hat{P}_{nT-1,v}$ as predictor and differential $\Delta_{nT,v}$. Using as predictors $\tilde{P}_{nT,v}$ ("noisy" P-frame motion-compensated from the same view) and $P_{nT,v}$'s ("noiseless" P-frames motion-compensated from different views u's, where $u \neq v$ and $\max(1, v-1) \leq u \leq \min(M, v+1)$), encoder determines sufficient amount of LDPC code required to overcome the largest noise in these predictors to recover target $I_{nT,v}$. The chosen LDPC code *alone* (without frame differential $\Delta_{nT,v}$) are the encoded bits for $W^u_{nT,v}(k)$. Like uDSC frame for wireless multicast, combination of P-frames $P_{nT,v}$'s and this uDSC frame $W^u_{nT,v}(k)$ can also enable view-switching *or* halt error propagation in the same view.

The reason for this new structure construction is as follows. Given the multiple predictors $P_{nT,v}$'s of uDSC frame $W^u_{nT,v}(k)$ are of the same instant and same view, $W^u_{nT,v}(k)$ does not need to perform motion prediction first, and hence there is no need to encode multiple sets of MVs for multiple predictors. Further, The largest noise among predictors $P_{nT,v}$'s of the same view are likely much smaller than the largest noise among predictors of different views (as in structure in Fig. 4.3). Thus, the resulting uDSC frame $W^u_{nT,v}(k)$ will be much smaller.

During IMVS unicast, this modified structure has the following advantage. If a client does not switch view at a view-switching boundary (the likely case), then only a small P-frame $P_{nT,v}$ and a small uDSC frame $W^u_{nT,v}(k)$ are required for transmission. If a client does switch view (the unlikely case), then a larger cross-view predicted P-frame $P_{nT,v}$ and uDSC frame $W^u_{nT,v}(k)$ are needed. On the average then, fewer transmission bits are required compared to the structure in Fig. 4.3, where a large uDSC frame $W^u_{nT,v}(k)$ must be transmitted at a view-switching boundary, no matter if a client is switching view or not.

Packetization for IMVS unicast frame structure is very simple. We simply put encoded bits in each frame $F_{i,v}$ into its own packets. Optimal packet ordering (scheduling) is

discussed in Section 4.5. We assume a frame $F_{i,v}$ is mapped to $m_{i,v}$ motion and $r_{i,v}$ residual packets.

4.4 Optimized Streaming for Wireless IMVS Multicast

We now formalize an optimization problem to find the optimal structure parameters for wireless IMVS multicast: the frequency of DE-DSC insertion T', the error recoverability k for DE-DSC $W^1_{i,v}(k)$ and uDSC $W^u_{i,v}(k)$, and the number of FEC packets for each level, $\Phi_{i,v}$ and $\phi_{i,v}$. We first define the wireless transmission constraint for each coding unit $U_{i,v}$. We then derive the probabilities that: i) an entire coding unit $U_{i,v}$ is correctly decoded, and ii) a DE-DSC or uDSC is perfectly reconstructed. We then derive the appropriate objective function for our optimization.

4.4.1 Transmission Constraint

We assume a transmission constraint in number of packets B for each coding unit $U_{i,v}$. We can write the transmission constraint as follows:

$$M_{i,v} + Gr_{i,v} + \Phi_{i,v} + G\phi_{i,v} \leq B \tag{4.1}$$

In words, (4.1) states that the total number of packets used for motion and residual packets and both levels of FEC packets cannot exceed the bandwidth of B packets for unit $U_{i,v}$.

4.4.2 Preliminaries

We first formally define mathematical quantities that are useful when dealing with a GE packet loss model. Let $P(i)$ be the probability of having *at least* i consecutive transmissions in the good state in the GE model, given transmission starts in bad state. Further, let $p(i)$ be the probability of having *exactly* i good state transmissions between two bad state transmissions, given transmission starts in bad state. We write $P(i)$ and

$p(i)$ as follows:

$$\begin{aligned} P(i) &= \begin{cases} 1 & \text{if } i = 0 \\ q(1-p)^{i-1} & \text{otherwise} \end{cases} \\ p(i) &= \begin{cases} 1-q & \text{if } i = 0 \\ q(1-p)^{i-1}p & \text{otherwise} \end{cases} \end{aligned} \tag{4.2}$$

Similarly, we define $Q(i)$ and $q(i)$ as the probability of *at least* i consecutive bad state transmissions, and the probability of *exactly* i bad state transmissions, given transmission starts in good state. Equations for $Q(i)$ and $q(i)$ will be the same as those for $P(i)$ and $p(i)$, with the parameters p and q interchanged.

We can now recursively define the probability $R(m, n)$ of *exactly* m bad state transmissions in n total transmissions, given transmission starts in bad state:

$$R(m,n) = \begin{cases} P(n) & \text{for } m = 0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} p(i)R(m-1, n-i-1) & \text{for } 1 \leq m \leq n \end{cases} \tag{4.3}$$

Similarly, the probability $S(m, n)$ of *exactly* m good state transmissions in n total transmissions, given transmission starts in good state, is written in the same form as (4.3), with $Q(i)$ and $q(i)$ replacing $P(i)$ and $p(i)$ in (4.3), respectively.

4.4.3 Correct Receive Probability of a Coding Unit

Given previous definitions, we now derive the probability $\alpha_{i,v}$ that all motion and residual packets of unit $U_{i,v}$ are correctly delivered. As previously discussed, besides source packets, two levels of FEC packets are employed to protect against packet losses. Hence, a necessary condition for recovery is to require the number of lost packets not to exceed the total number of FEC packets used: $\Phi_{i,v} + G\phi_{i,v}$.

We first write $\alpha_{i,v}$ as a weighted sum of $\alpha_{i,v}^G$ and $\alpha_{i,v}^B$, the decoding success probability of unit $U_{i,v}$ given transmission starts in good and bad state respectively:

$$\alpha_{i,v} = \left(\frac{q}{p+q}\right)\alpha_{i,v}^G + \left(\frac{p}{p+q}\right)\alpha_{i,v}^B \tag{4.4}$$

Assuming transmission starts in the good state, m of B total packets can be transmitted in good state with probability $S(m, B)$. Unit $U_{i,v}$ can be successfully received if at least $r \geq M_{i,v} + Gr_{i,v}$ packets are correctly delivered, and these r packets can be a sum of r_G and $r - r_G$ delivered packets in good and bad states respectively. Hence we can write $\alpha^G_{i,v}$ as:

$$\alpha^G_{i,v} \approx \sum_{m=0}^{B} S(m,B) \sum_{r=M_{i,v}+Gr_{i,v}}^{B} \sum_{r_G=0}^{r} P_G(r_G, m) P_B(r - r_G, B - m) \tag{4.5}$$

where $P_G(x, y)$ and $P_B(x, y)$ are the probabilities of exactly x delivered packets in y tries in good and bad state respectively:

$$P_G(x,y) = \begin{cases} \binom{y}{x} (1-g)^x g^{y-x} & \text{if } x \leq y \\ 0 & \text{o.w.} \end{cases} \tag{4.6}$$

$$P_B(x,y) = \begin{cases} \binom{y}{x} (1-b)^x b^{y-x} & \text{if } x \leq y \\ 0 & \text{o.w.} \end{cases} \tag{4.7}$$

$\alpha^B_{i,v}$ can be written similarly to $\alpha^G_{i,v}$ in (5.12) and is hence omitted.

4.4.4 Correct Decode Probability for DE-DSC / uDSC

We next derive the correct decode probability for all DSC frames (DE-DSC or uDSC) in unit $U_{i,v}$, *given* not all motion and residual packets in the unit are correctly delivered. A DSC frame is correctly decoded if: i) all motion packets between two DSC frames are correctly recovered, and ii) residual packets of at least $T' - k - 1$ preceding P-frames are correctly recovered.

We first consider the probability $\delta_{i,v}$ that the motion information of all frames in unit $U_{i,v}$ are correctly recovered. As done in (5.11) for $\alpha_{i,v}$, $\delta_{i,v}$ can also be written as a weighted sum of $\delta^G_{i,v}$ and $\delta^B_{i,v}$, depending on whether transmission starts in good or bad state. Let $\gamma_M = (M_{i,v} + \Phi_{i,v})/B$ be the fraction of bandwidth for transmission of motion and level-1 FEC packets. $M_{i,v}$ motion packets are correctly recovered if at least $r \geq M_{i,v}$ packets are correctly delivered, where again r can be a sum of r_G and $r - r_G$ packets

delivered in good and bad state, respectively. The difference from (5.12) is that for given m and $B-m$ transmissions in good and bad states, only portions $\gamma_M m$ and $\gamma_M(B-m)$ are used for transmission of motion and level-1 FEC packets. We can now write $\delta_{i,v}^G$ as follows:

$$\delta_{i,v}^G \approx \sum_{m=0}^{B} S(m,B) \sum_{r=M_i}^{M_{i,v}+\Phi_{i,v}} \sum_{r_G=0}^{r} P_G(r_G, \gamma_M m) P_B(r-r_G, \gamma_M(B-m)) \tag{4.8}$$

Next, we derive the probability $\eta_{i,v}$ that residual packets of at least $T'-k-1$ of $T'-1$ P-frames are recovered for DSC frame to be correctly decoded. Given our packetization scheme, that means at least $T'-k-1$ residual groups are correctly recovered. Because interleaving was performed to space packets in one residual group to be as far apart as possible, we can approximate packet losses within a residual group as iid losses, with probability $l = \left(\frac{p}{p+q}\right) g + \left(\frac{q}{p+q}\right) b$. The probability $\theta_{i,v}$ that a residual group is correctly recovered is hence:

$$\theta_{i,v} = \sum_{r=r_{i,v}}^{r_{i,v}+\phi_{i,v}} \begin{pmatrix} r_{i,v}+\phi_{i,v} \\ r \end{pmatrix} (1-l)^r l^{r_{i,v}+\phi_{i,v}-r} \tag{4.9}$$

In words, (4.9) states that a residual group must receive at least $r_{i,v}$ packets for the group to be correctly recovered.

Having derived $\theta_{i,v}$, we can now write $\eta_{i,v}$ as follows:

$$\eta_{i,v} \approx \sum_{j=T'-k-1}^{T'-2} \begin{pmatrix} T'-1 \\ j \end{pmatrix} \theta_{i,v}^j (1-\theta_{i,v})^{T'-1-j} \tag{4.10}$$

where the upper limit in (4.10) is $T'-2$, since by assumption not all the motion and residual packets in the coding unit are correctly recovered.

4.4.5 Objective Function

We now derive our objective function as the expected distortion reduction D in the GOP given chosen parameters for the frame structure. For a coding unit $U_{i,v}$ to be correctly decoded, each previous unit $U_{j,v}$, $j<i$, must be either fully correctly received with probability $\alpha_{j,v}$, or have all its DSC frames correctly decoded with probability

$(1-\alpha_{j,v})\delta_{j,v}\eta_{j,v}$. If the entire unit $U_{i,v}$ is correctly delivered as well, then all T frames are correctly decoded with distortion reduction $D'_{i,v}$. Otherwise, if at least the T/T' DSC frames are correctly decoded, then correct decoding of T/T' DSC frames brings distortion reduction $D''_{i,v}$, while the remaining frames' distortion reduction is assumed to be 0. $D'_{i,v}$ and $D''_{i,v}$ can be written simply as a sum of individual frames' distortion reduction $d_{i,v}$'s:

$$D'_{i,v} = \sum_{l=0}^{T-1} d_{iT+l,v} \qquad D''_{i,v} = \sum_{j=0}^{T/T'} d_{iT+jT',v} \tag{4.11}$$

The objective function D can now be written as:

$$\begin{aligned} \max \quad D &= \sum_{v=1}^{M}\sum_{i=0}^{\tau-1}\left(\alpha_{i,v}\ D'_{i,v} + (1-\alpha_{i,v})\delta_{i,v}\eta_{i,v}\ D''_{i,v}\right) Y_{i,v} \\ Y_{i,v} &= \prod_{j=1}^{i-1}\alpha_{j,v} + (1-\alpha_{j,v})\delta_{j,v}\eta_{j,v} \end{aligned} \tag{4.12}$$

The goal is to find parameters that maximize D in (5.13) for the entire GOP subject to transmission constraint (4.1) for each coding unit $U_{i,v}$.

4.4.6 Coding Structure Optimization

Having formally defined the optimization above, we now describe a simple heuristic to find good structure parameters. We observe that since DSC frames of coding units early in the GOP affect the decodability of differentially coded frames later in the GOP, early DSC frames should have high probability of correct decoding. Specifically, DE-DSC frame insertion period T' for an early coding unit should be no smaller than a later coding unit, since fewer DE-DSC frames in a unit means more bandwidth left for FEC packets, resulting in higher correct decode probability for DE-DSC frames. Further, more sparsely inserted DE-DSC frames should have larger DSC error recovery ability k, resulting in a higher tolerance for burst losses in the GE model. Note that sparse insertion of DE-DSC frames also means a longer error propagation when a P-frame is not correctly decoded. For early coding units, however, higher DE-DSC correct decoding probability is more important than short error propagation.

Based on the above observation, we optimize the structure parameters from the last coding unit $U_{\tau-1,v}$ *backward* to the first coding unit $U_{0,v}$ as follows. For $U_{\tau-1,v}$, we first

insert one DE-DSC frame into the unit ($T' = T/2$), and let DSC error recoverability $k = 1$. We then equally allocate the remaining available bandwidth to level-1 and level-2 FEC, $\Phi_{\tau-1,v}$ and $\phi_{\tau-1,v}$. Then, we incrementally increase $\Phi_{\tau-1,v}$ and decrease $\phi_{\tau-1,v}$ (since motion packets are more important) while observing bandwidth constraint, until no further gain in objective value (5.13) is possible. We subsequently increase k by 1 and perform the same FEC allocation procedure to find locally optimal $\Phi_{\tau-1,v}$ and $\phi_{\tau-1,v}$. We stop when a larger k does not lead to further gain, resulting in the locally optimal set of parameters $(k, \Phi_{\tau-1,v}, \phi_{\tau-1,v})$ for given T'. We then incrementally increase insertion of DE-DSC frames and repeat the process to search for better parameter sets, until no further gain is observed. The best parameter set $(T', k, \Phi_{\tau-1,v}, \phi_{\tau-1,v})$ is implemented for $U_{\tau-1,v}$.

Given structure parameters for $U_{i+1,v}, \ldots, U_{\tau-1,v}$ are discovered, we find parameters for $U_{i,v}$ as follows. We first set T' and k for $U_{i,v}$ to be the same as $U_{i+1,v}$, and find the optimal FEC allocation as described above. Then, we iteratively increase T' and k to find better allocation, until no further gain is observed.

4.5 Optimized Streaming for Wired IMVS Unicast

In this section, we discuss how IMVS coding structure with uDSC frames is optimized in the wired IMVS unicast scenario. We will assume iid packet losses and tight playback deadlines for each streaming client.

4.5.1 Overview of Server Packet Scheduling Optimization

For wired IMVS unicast, we assume that there are tight playback deadlines for video frames; from the moment a client has started video playback after waiting an initial buffering period of b_o seconds, each successive frame $F_{i,v}$ must arrive before its playback deadline, or decoding of $F_{i,v}$ fails. b_o is also the maximum buffering size (in time) the client can accommodate; i.e., the server cannot send more than b_o seconds worth of frames into the future beyond the frame the client is currently watching to avoid buffer overflow.

We assume that the transmission channel has bandwidth $\bar{C}$ bits per second (bps). For fixed packet size S bits and a given transmission duration of t seconds, bandwidth $\bar{C}$ translates to a fixed number of possible packet transmissions $f(t) = \lfloor \bar{C} * t/S \rfloor$. If t is the amount of time until a frame's playback deadline expires, then $f(t)$ is also the number of TO until the expiration of the frame at decoder, upon which the frame is late and decoding fails. For simplicity, we assume each packet loss is known instantly at the server with zero delay.

We optimize packet transmission one code block at a time. As discussed in Section 4.3.3, each frame F_i is divided into m_i motion and r_i residual packets[2]. We assume the following server behavior: once it has committed to transmit motion or residual packets of a given frame F_i, the m_i motion or r_i residual packets will be (re)transmitted until: i) the committed motion / residual packets are all correctly delivered, or ii) the available TOs until F_i expires have been depleted. In the latter case, it means F_i cannot be correctly decoded.

For a given MV coding structure, it is clear that *all* motion packets in a coding block must be correctly delivered for the next DE-DSC or uDSC frame to be correctly decoded. The hard decision for the server is the following: should the r_i residual packets of a frame F_i be transmitted towards the goal of having F_i itself correctly decoded, or should the m_j motion packets of a later frame F_j, $j > i$, be transmitted towards the goal of having the next DSC $W_{l,v}(k)$ correctly decoded, forsaking the correct decoding of F_i? This is the optimization we will formulate using MDP.

4.5.2 Markov Decision Process

We use MDP [37] as the mathematical tool to help the server make the best possible decisions of which motion or residual packets of which frame in a code block to transmit at any given time. We define the three major components of MDP in order: i) state & action space, ii) state transition probabilities, and iii) benefit function.

[2]For simplicity in explanation, we will drop the subscript for view v in this section.

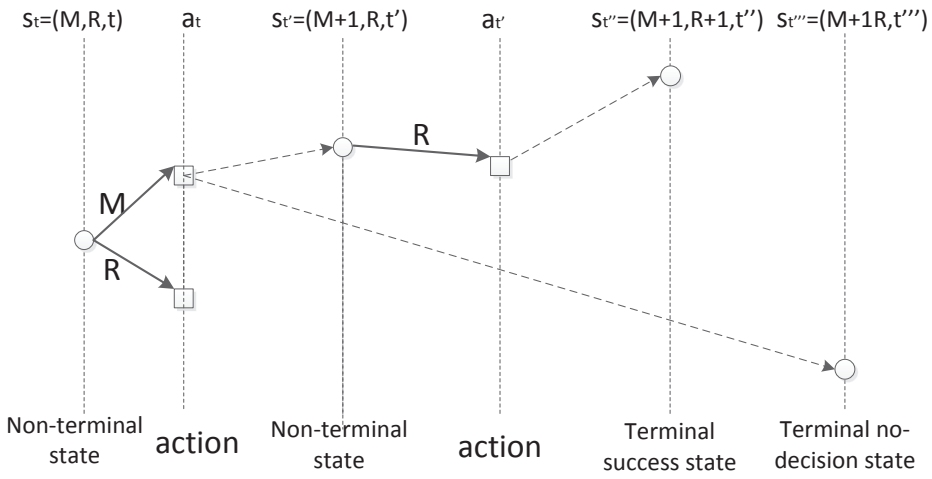


FIGURE 4.6: Example of Markov Decision Process.

4.5.2.1 State & Action Space for MDP

In a nutshell, a MDP is a finite horizon recursion, where the leaf node of the recursion tree marks the end of the decision process. Each recursive call at time t is associated with a state s_t. MDP then selects an action a_t given state s_t, which will then lead to multiple recursive calls with states $s_{t'}$'s of later time t''s, $t' > t$, with state transition probabilities $P_{(s_t,a_t)}(s_{t'})$'s. MDP will spur no more recursive calls when the state s_t is a *terminal state* (to be formally defined shortly). The solution of a MDP is a *policy* π that maps each feasible non-terminal state s_t to an action a_t, i.e., $\pi : s_t \mapsto a_t$.

Specifically, we first define the state space for our MDP for a code block of frames $F_0, \ldots, F_{T'-1}$. Let $s_t = (M, R, t)$ be a state at time t, where M is the number of consecutive frames starting from F_0 with all motion packets correctly received, and R is the number of consecutive frames starting from F_0 with all residual packets correctly received. $t = 0$ is the start of the optimization, and initial state is $s_0 = (0, 0, 0)$.

Since there are only motion and residual packets, given state $s_t = (M, R, t)$, possible actions a_t's are: i) transmit m_M motion packets for the next frame F_M with motion packets not transmitted yet (action denoted as $\mathcal{M}$), or ii) transmit r_R residual packets for the next frame F_R with residual packets not transmitted yet (action denoted as $\mathcal{R}$). Because motion packets are more important than residual packets, server will always choose to send m_i motion packets before r_i residual packets for a given frame F_i. Hence state $s_t = (M, R, t)$ will always have $M \geq R$. State s_t is a terminal state when: i) state

$s_t = (M, R, t)$ is a *success state* and has all motion and residual packets delivered, i.e., $M = R = T'$; or ii) state s_t is a *no-decision state* and indicates that residual packets of frame F_{R-1} have missed the playback deadline of F_{R-1}, i.e., $(t - b_o) * FPS > R - 1$. In the latter case, the only logical action left is to transmit the remaining packets in the code block that are necessary for correctly decoding the next DSC frame.

Figure 4.6 is a simple illustration of MDP. For a non-terminal state s_t, the corresponding action could be $\mathcal{M}$ or $\mathcal{R}$, leading to different future states. Figure shows that action $a_t = \mathcal{M}$ can lead to two possible future states: non-terminal state ($s_{t'}$) and terminal no-decision state ($s_{t'''}$). State $s_{t'''}$ means that residual packets $R - 1$ has missed frame F_{R-1}'s playback deadline. For non-terminal state $s_{t'}$, MDP will continue until a different terminal no-decision state is reached, or a terminal success state is reached.

4.5.2.2 Client Buffer Adjustment

To avoid client buffer overflow, server will avoid sending more than b_o seconds worth of frames beyond the client's current playback frame. This can be accounted for simply in the MDP. Given state $s_t = (M, R, t)$, the amount of video data delivered (in second) minus the amount already played back, $M/FPS - (t - b_o)$, must be smaller than buffer size in second b_o; i.e., $M/FPS \leq t$. Otherwise, the server must wait $M/FPS - t$ seconds before another action can be taken. In other words, state $s_t = (M, R, t)$ maps directly to state $s_{M/FPS} = (M, R, M/FPS)$ before an action is taken via MDP if $M/FPS > t$.

4.5.2.3 Transmission Probabilities for MDP

We now define the transmission probability for MDP—the likelihood of landing in future state $s_{t'}$ given the server takes action a_t at state $s_t = (M, R, t)$. Given action a_t can only be motion $\mathcal{M}$ or residual $\mathcal{R}$, the corresponding future state can only be $s_{t'} = (M+1, R, t')$ and $s_{t'} = (M, R + 1, t')$, for $t' > t$. Note that because t' is precisely the time when a discrete number of motion or residual packets have been successfully transmitted, we know t' belongs to one of the following two discrete sets:

$$t' = \begin{cases} t + (m_M + k)S/\bar{C}, & k \in \mathcal{Z}^+ \quad \text{if} \quad a_t = \mathcal{M} \\ t + (r_R + k)S/\bar{C}, & k \in \mathcal{Z}^+ \quad \text{if} \quad a_t = \mathcal{R} \end{cases} \tag{4.13}$$

where $\mathcal{Z}^+$ denotes the set of non-negative integers. In words, (4.13) says that t' depends on the number $m_M + k$ of actual transmissions used for the correct delivery of m_M motion packets if $a_t = \mathcal{M}$ (similar expression if $a_t = \mathcal{R}$).

Given iid packet loss model with loss probability β, if $a_t = \mathcal{M}$, then state transition probability $P_{s_t,a_t}(s_{t'})$ can be derived simply:

$$P_{(M,R,t),\mathcal{M}}(M+1,R,t') = \begin{pmatrix} (t'-t)\bar{C}/S-1 \\ m_M - 1 \end{pmatrix} \beta^{(t'-t)\bar{C}/S-m_M}(1-\beta)^{m_M} \tag{4.14}$$

(4.14) states that for transmission period $t'-t$, $(t'-t)\bar{C}/S-1$ transmissions were first expended to successfully delivered $m_M - 1$ motion packets, and then the last motion packet is successfully delivered with probability $1-\beta$. Similar expression can be written for $P_{(M,R,t),\mathcal{R}}(M,R+1,t')$ if $a_t = \mathcal{R}$.

4.5.2.4 Benefit of Each Action

We now assign benefit $B_{s_t,a_t}(s_{t'})$ for each state transition $(s_t, a_t) \mapsto s_{t'}$. There are only two cases where we need to assign a non-zero benefit value to a state transition. First, given $s_t = (M,R,t)$, if action $a_t = \mathcal{R}$, and residual packets of frame F_R are transmitted correctly before playback deadline of F_R, i.e. $t' - b_o < (R+1)/FPS$, then F_R can be correctly decoded on time, resulting in distortion reduction d_R. If transmitted residual packets are of the last frame $F_{T'-1}$, we assign *in addition* a large distortion reduction d_{DSC}. Since the following DSC frame in the next code block must be correctly decoded for the remaining frames in the GOP to be correctly decoded, an additional d_{DSC} for the last frame in the current block can reflect its importance for the remaining frames in the GOP.

Second, if $s_{t'} = (M',R',t')$ misses the playback deadline of frame $F_{R'-1}$, i.e., $t' - b_o > R/FPS$, then $s_{t'}$ is a terminal non-decision state. To assign a benefit for $s_{t'}$, we need to first calculate the probability P_{DSC} that sufficient motion and residual packets in this block are transmitted correctly for the following DSC frame in the next block to be correctly decoded. We then multiply P_{DSC} by d_{DSC} as benefit for $s_{t'}$.

Let the remaining number of packets necessary for decoding of the following DSC frame to be K. It is the sum of remaining motion packets in T' frames in the block, plus

remaining residual packets (if any) in $T' - k$ frames in the block:

$$K = \sum_{i=M'}^{T'-1} m_i + \sum_{j=R'}^{T'-1-k} r_i \tag{4.15}$$

Let the number of remaining TOs in the block be Q:

$$Q = \left\lfloor (T'/FPS + b_o - t')\bar{C}/S \right\rfloor \tag{4.16}$$

We can now write P_{DSC} as the probability that at least K packets are successfully transmitted in Q transmissions:

$$P_{DSC} = \sum_{i=K}^{Q} \begin{pmatrix} Q \\ i \end{pmatrix} \beta^{Q-i}(1-\beta)^i \tag{4.17}$$

4.5.2.5 Finding Optimal Policy for MDP

The optimal policy π^* is one that leads to the minimum expected distortion (maximum distortion reduction) for the lifetime of the MDP. Denote by $\pi_t^*(s_t)$ the maximum expected distortion reduction, given current state s_t. $\pi_t^*(s_t)$ can be defined recursively: a chosen action a_t at state s_t leads to state $s_{t'}$ with state transition probability $P_{s_t,a_t}(s_{t'})$ and benefit $B_{s_t,a_t}(s_{t'})$. Assuming optimal policy $\pi_{t'}^*(s_{t'})$ is performed at instant t', we have expected benefit $P_{s_t,a_t}(s_{t'})\left[\pi_{t'}^*(s_{t'}) + B_{s_t,a_t}(s_{t'})\right]$. $\pi_t^*(s_t)$ exhaustively searches for the optimal action a_t^* given state s_t.

$$\pi_t^*(s_t) = \max_{a_t} \sum_{s_{t'}} P_{s_t,a_t}(s_{t'})\left[\pi_{t+1}^*(s_{t'}) + B_{s_t,a_t}(s_{t'})\right] \tag{4.18}$$

(4.18) can be solved using *dynamic programming* (DP). Specifically, each time a subproblem to $\pi_t^*(s_t)$, where state $s_t = (M, R, t)$, is solved, the solution is placed in entry $[M, R, t * C/S]$ of a DP table[3] Π. Doing so means that the solution to a repeated subproblem $\pi_t^*(s_t)$ can be simply looked up instead of solved again, reducing computation.

[3] $t * C/S$ is an integer, since t is the amount of time transmitting an integer number of packets over a constant bandwidth channel.

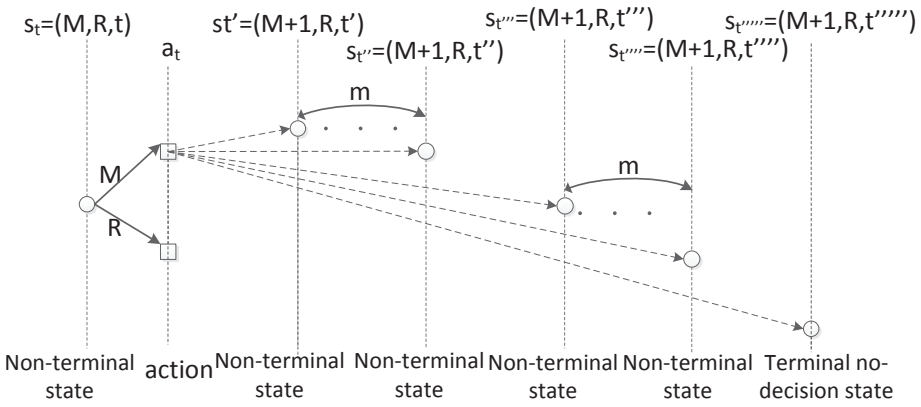


FIGURE 4.7: Example of computation reduction for Markov Decision Process.

4.5.3 Coding Structure Optimization

As done in Section 4.4.6, we use a heuristic to find locally optimal structure parameters: DE-DSC insertion period T' and DSC error recoverability k. We will again make the observation that T' and k for a coding unit early in the GOP should be no smaller than a coding unit later in the GOP. Based on this observation, we again follow similar optimization procedure, where we optimize parameters for the last coding unit $U_{\tau-1,v}$ back to the first coding unit $U_{0,v}$. For each coding unit $U_{i,v}$, we ensure that T' and k for $U_{i,v}$ are no smaller than ones in $U_{i+1,v}$.

One can reduce the computation complexity of our proposed MDP packet scheduling, with a cost in optimization accuracy, using the following method. For state s_t, after choosing action a_t ($\mathcal{M}$ or $\mathcal{R}$), the client may arrive at a set of non-terminal states with different probabilities. We propose to combine every m consecutive non-terminal states into one, as illustrated in Figure 4.7. For the new combined state s_{t^*}, time t^* will be the average time of the m original states. The probability of arriving at state s_{t^*} will be the sum of probabilities of arriving at each of the m states. Given the optimization problem is solved using DP, this method effectively reduces the computation complexity of MDP by a factor of m, by reducing the size of the DP table by the same factor.

4.6 Experimentation

In this section, we present extensive experiments conducted to demonstrate the performance advantage of our proposed optimized IMVS frame structures with uDSC frames over competing schemes without uDSC frames, for both wireless multicast and wired unicast streaming scenarios.

4.6.1 Experimental Setup

We modified DSC codec in [85]—a H.263-based[4] codec using LDPC code for channel coding to overcome noise in the side information—to generate uDSC frames as described in Section 4.3. In the first experiments, I- and P-frames were encoded using comparable H.263 tools. Multiview video test sequences used were 150-frame `Akko` [5] at 640×480 resolution and 75-frame `Breakdancers` [114] at 1024×768 resolution. We assume there are $M = 3$ captured views for each sequence. Video playback speed for `Akko` and `Breakdancers` are $30fps$ and $15fps$, with view-switching period T being 30 and 15 respectively.

We fixed the *quantization parameters* (QP) for I-, P- and DSC frames so that the resulting visual quality in PSNR for each frame due to source coding only is roughly 35.5dB for `Akko` and 37.2dB for `Breakdancers`. Given a coding unit is composed of an I-frame followed by P-frames, the source coding rates for `Akko` and `Breakdancers` are 587kbps, 788kbps respectively, when coded using H.263.

The size of DE-DSC will increase as the error recovery ability increases. When the DE-DSC frame can tolerate residual loss of 4 P-frames among 7 transmitted frames, the size of a typical DE-DSC is about 2.25 and 2.8 times of the size of a H.263 encoded P-frame for `Akko` and `Breakdancers`, respectively. We note that two multiview test sequences have very different characteristics: `Akko` has closer capturing cameras and slower video motion than `Breakdancers`.

An event-driven network simulator developed in-house was used to simulate packet losses according to the GE and iid packet loss models for wireless multicast and wired unicast.

[4] For the purpose of demonstrating the effectiveness of our proposed uDSC frame in facilitating view-switching and halting error propagation in IMVS scenarios (rather than raw coding performance), we believe DSC frames constructed using H.263 tools is sufficient.

[5] http://www.tanimoto.nuee.nagoya-u.ac.jp

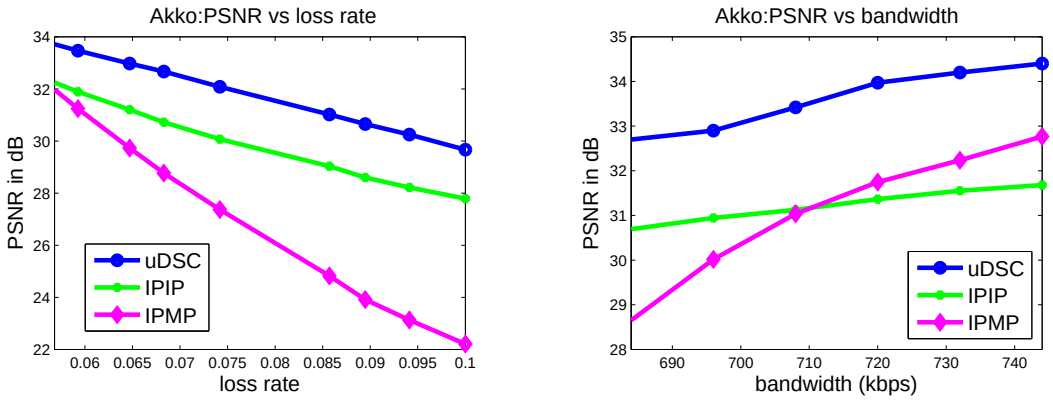
Each performance point in the network simulation experiments is the average result of over $100,000$ experimental trials. MTU was set to be 1500bytes.

We assume that for all streaming scenarios, the decoder performs *frame freeze*, which means that when a frame i cannot be correctly decoded (due to missing packets to this frame or a previous frame in its motion prediction chain at frame i's playback time), the encoder will display the last correctly decoded frame j as a replacement. Hence the *mean square error* (MSE) between the replacement decoded frame j and the original target frame i is computed as the distortion of the incorrectly decoded frame i. PSNR is then computed as a function of MSE.

4.6.2 Wireless IMVS Multicast Scenario

In this section, we present experimental results for the wireless IMVS multicast scenario. Packet losses were generated using the GE model with parameters p, q, g, b, as discussed in Section 4.2.2. We compare the performance of our optimized IMVS structure (`uDSC`) with two competing schemes. In `IPIP`, each coding unit is composed of an I-frame plus $T-1$ P-frames. At stream time, the server would vary the amount of source packets transmitted according to network conditions by dropping trailing P-frames. The selected frames would be divided into two groups, and UEP FECs were applied thereafter. We found the optimal number of frames to transmit, the division of selected frames into two groups, and UEP FEC for the two groups via exhaustive search to achieve the maximum expected receiver video quality for a given set of network conditions. `IPIP` thus represents the state-of-the-art without DSC frames. In `IPMP`, each coding unit (except the first unit, which starts with an I-frame) is composed of a MP-DSC frame plus $T-1$ P-frames. `IPMP` allocates the largest possible number of FEC packets for equal error protection of the generated source packets, given the same network bandwidth. `IPMP` represents coding structures with DSC frames but without optimized transport mechanisms.

At encoding time, parameters of our proposed structure `uDSC` were optimized assuming loss rate was $\alpha = 5.68\%$, and bandwidth is $1+\alpha$ times the source coding rate (587kbps for `Akko`) and rounded up to the nearest integer multiple of 100kbps, which is 700kbps. At stream time, the actual loss parameters of the channel were used to optimize packet transmission. Figure 4.8(a) shows the resulting PSNR of the decoded video at client



(a) bandwidth 700kbps

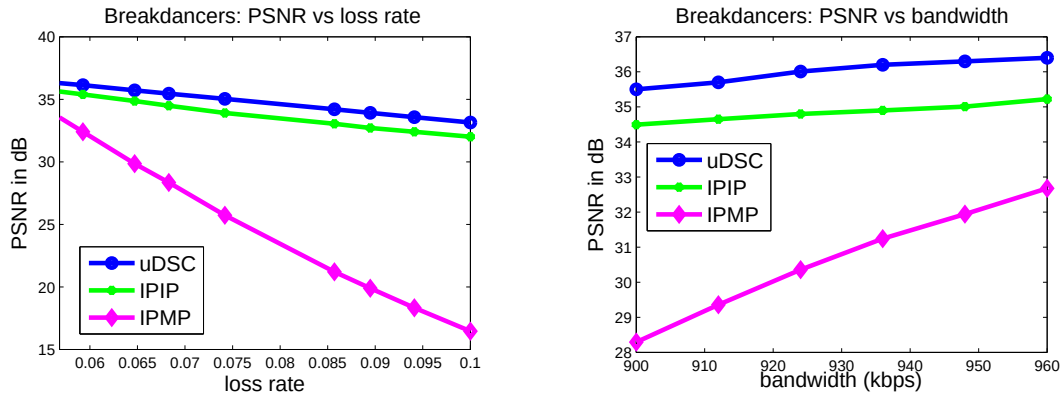
(b) loss rate 6.9%

FIGURE 4.8: PSNR versus loss rate (a) and bandwidth (b) for `Akko`. Loss rates are varied by changing p, while $q = 0.15$, $g = 0.05$ and $b = 0.8$.

using the three schemes for `Akko` at different loss rates (by varying p), where $g = 0.05$, $b = 0.8$, $q = 0.15$ and bandwidth was 700kbps. We see that `uDSC` outperformed the other competing schemes by up to 1.9dB. It is because suitable DE-DSC and uDSC insertion, in combination with optimized FEC, packetization and reordering greatly improved error resiliency. `IPIP` can also halt the error propagation, but is less effective than `uDSC` due to I-frame's larger size compared to an uDSC frame, leaving little bandwidth for transmission of trailing P-frames and FEC.

In Figure 4.8(b), we plot the resulting PSNR against bandwidth, while the loss rate is fixed at 6.9%. The parameters of the GE models were the same. We see that as the bandwidth available for transmission increases, performance for all three schemes improves, but `uDSC` still consistently outperforms `IPIP` and `IPMP`. In particular, `uDSC` outperformed `IPIP` by up to 2.8dB.

Similar experiments were conducted for `Breakdancers`, and results are shown in Figure 4.9. We see that similar trends can be observed. In Figure 4.9(a), the bandwidth was set to 900kbps, GE parameters were set as: $g = 0.05$, $b = 0.8$, $q = 0.15$. We see that the largest PSNR gain of `uDSC` over the competing schemes is 1.15dB. In Figure 4.9(b), we fixed the loss rate at 6.9% and varied the transmission bandwidth. The largest PSNR gain of `uDSC` over the other competing schemes is 1.3dB.



(a) bandwidth 900kbps

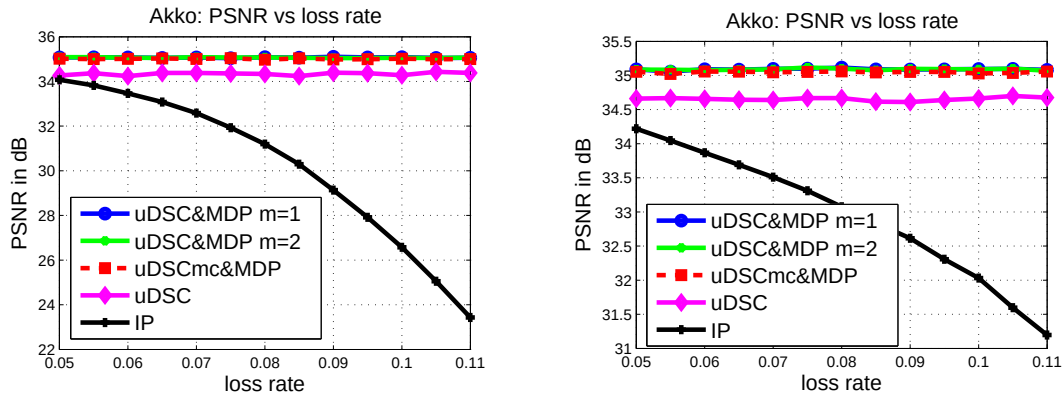
(b) loss rate 6.9%

FIGURE 4.9: PSNR versus loss rate (a) and bandwidth (b) for `Breakdancers`. Loss rates are varied by changing p, while $q = 0.15$, $g = 0.05$ and $b = 0.8$.

4.6.3 Wired IMVS Unicast Scenario

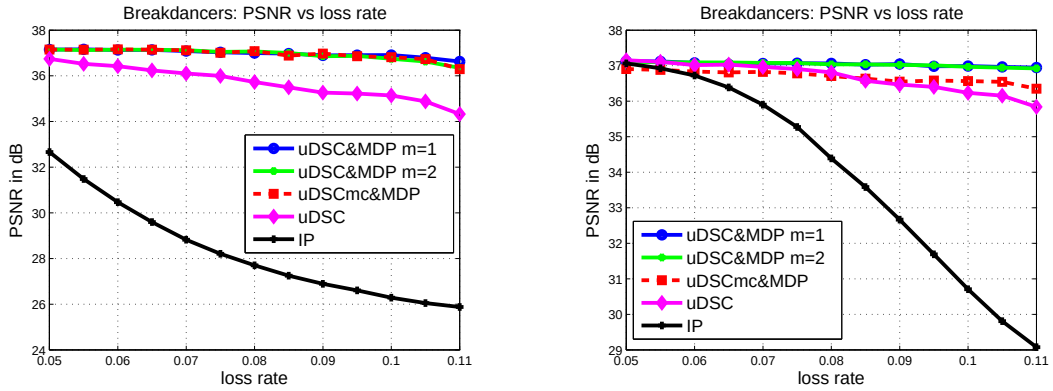
In this section, we present experimental results for the wired IMVS unicast scenario. The packet loss model used is iid with parameter β, as discussed in Section 4.2.2. We compare the performance of our optimized IMVS structure (`uDSC&MDP m=1`) with four competing schemes. `uDSC&MDP m=2` is the complexity-reduced version of `uDSC&MDP m=1` by combining two consecutive non-terminal states into one, as discussed in Section 4.5.3. `uDSCmc&MDP` uses a single uDSC frame proposed for wireless multicast as discussed in Section 4.3.2.1 for both view-switching and error resilience (as opposed to the multiple P-frames plus small uDSC frame for unicast as discussed in Section 4.3.3). As a result, `uDSCmc&MDP` must transmit a large uDSC frame whether view-switching is performed or not. `uDSCmc&MDP` performs packet scheduling using MDP. `uDSC` uses the proposed coding structure for unicast described in Section 4.3.3 but does not use MDP for packet scheduling. Instead, the server transmits motion and residual packets of frames in succession, and as soon as it fails to meet a frame's playback deadline, it transmits packets necessary to decode the next uDSC frame. In `IP`, each coding unit is composed of an I-frame plus $T - 1$ P-frames. For the competing schemes not employing MDP for packet scheduling, the server transmits motion and residual packets of frames in succession. All schemes use the same network bandwidth and the same buffer time.

The results, in PSNR of decoded video at client versus loss rate, are shown in Figure 4.10 for `Akko`. The initial buffering periods were 0.16s and 0.15s, and bandwidths were 630kbps and 670kbps for Figure 4.10(a) and (b), respectively. In Figure 4.10(a), we can



(a) bandwidth 630kbps b_0=0.16s (b) bandwidth 670kbps b_0=0.15s

FIGURE 4.10: Akko: Comparison of Received Video quality vs loss rate for different schemes.



(a) bandwidth 810kbps b_0=0.27s (b) bandwidth 850kbps b_0=0.23s

FIGURE 4.11: Breakdancers: Comparison of Received Video quality vs loss rate for different schemes.

see that `uDSC&MDP m=1` outperformed `IP` by up to 11.6dB. We see also that complexity reduction from `uDSC&MDP m=1` to `uDSC&MDP m=2` incurred a PSNR drop of at most 0.2dB. We see that the performance of all schemes dropped when the loss rate increased. Larger uDSC frames used in `uDSCmc&MDP` resulted in a lower PSNR when compared to `uDSC&MDP`. We see also the importance of using MDP for packet scheduling, as `uDSC`, which used the same structure as `uDSC&MDP`, performed poorly compared to `uDSC&MDP`.

As shown in Figure 4.10(b), when the bandwidth increased, all the schemes performed better. However, the comparative trends are the same as Figure 4.10(a). We can see the performance gain for `uDSC&MDP m=1` over `IP` is up to 3.8dB.

Similar experiments were conducted for `Breakdancers`, and results are shown in Figure 4.11. We see that similar behavioral trends can be observed. In Figure 4.11(a),

initial buffer period and bandwidth were set to be 0.27s and 810kbps respectively. We can see that the PSNR gain is up to 11dB compared to `IP`. The cost of computation complexity reduction is at most 0.2dB. We increased the bandwidth to 850kbps and show the corresponding results in Figure 4.11(b). We can see the gain is up to 7.9dB compared to `IP`.

4.7 Summary

In IMVS, a client can periodically switch to neighboring captured views as the video is played back in time. The technical challenge is to design coding structure to facilitate periodic view-switching, while providing some level of error resiliency, so that error propagation due to differentially coded frames can be mitigated. In this paper, we first propose a new frame type called uDSC frame that can both facilitate view-switching and halt error propagation. We then optimize transmission strategies for coding structures with uDSC frames for wireless IMVS multicast and wired IMVS unicast scenarios. Experimental results show that optimal transmission of coding structure with uDSC frames out-performs competing schemes by up to 2.8dB and 11.6dB for the two streaming scenarios.

Chapter 5

Multi-Path Free Viewpoint Video Streaming

This chapter introduces the system for multiple description coding and joint interview/temporal recovery of free viewpoint video transmitted over two network paths. Near-optimal source and channel coding rates for each description are selected using an efficient branch-and-bound method for the given transmission bandwidth on each path.

5.1 Introduction

The popularity of stereoscopic video, where two texture images captured from two closely collocated cameras are shown respectively to each of the viewer's eyes in order to induce a perception of depth in the 3D scene, is indisputable. However, it is known that *motion parallax* [106], where the viewer's head movement triggers a corresponding shift in the viewing angle of the observed scene, represents an even stronger stimulus of depth perception [115]. With stereoscopic video, the same two views are shown to the viewer's two eyes regardless of how much the viewer moves his head. This results in physical objects in the 3D scene appearing as unnatural flat layers, which is undesirable.

One technology to enable motion parallax is FVV [50]. At the sender, a large 1D array of closely spaced cameras synchronously captures texture and depth images[1] of the same 3D scene from slightly different viewing angles. The sender then transmits texture and depth maps of two adjacent captured views—a format known as *texture-plus-depth* [57]—that are closest to the viewer's viewing perspective of the scene, as governed by his head movement that is dynamically tracked over time [43]. (The two transmitted views are denoted as left and right views in the sequel.) The viewer can then synthesize any intermediate virtual view that corresponds to his present viewpoint of the scene via DIBR [118], using texture and depth maps of the two captured views as references. This results in an enhanced 3D depth perception via the aforementioned motion parallax.

If the communication path between the sender and receiver is over wireless links that are known to be burst-loss prone due to shadowing, slow channel fading, and interference [23], then the resulting packet losses of texture and depth video data are difficult to overcome and can severely affect the synthesized view quality. This is especially true since the interactivity of free viewpoint video mandates stringent playback deadline requirements at the receiver [49]. Therefore, packet loss recovery strategies based on automatic retransmission request [119], which exhibit round-trip-time delays, are not applicable.

To tackle this challenge, we propose a novel MDC, and joint inter-view and temporal recovery system for wireless multi-path streaming of free-viewpoint video. Specifically, we construct description D_1 to comprise four sub-streams of data that are encoded separately. They are the even frames of the texture and depth maps of the left view and the odd texture and depth frames of the right view. Similarly, the odd texture and depth frames of the left view and even texture and depth frames of the right view comprise the second description D_2. Each description is transmitted over a disjoint network path. Furthermore, appropriate QP and channel coding rates are selected for the sub-streams comprising the two descriptions using an efficient *branch-and-bound* (BB) algorithm.

Like MDC for single-view video [87], if the receiver receives one description but loses the other during transmission, the sole received description can be independently decoded, resulting in reduced, but still acceptable, video quality. Yet, unlike single-view video

[1]Texture image is a digital (color) image that includes color information (e.g., red (R), green (G), or blue (B)) for each pixel. A depth image comprises per-pixel distances between physical objects in the 3D scene and the capturing camera. It can be either captured directly via a depth sensor [116] or estimated from neighboring texture images using stereo-matching algorithms [117].

MDC [87], our MDC is carefully designed so that a lost frame in one description can be partially reconstructed using available frames in the received description, exploiting both temporal and inter-view correlation. Our recovery approach comprises two methods.

In the first method, denoted as *temporal super-resolution* (TSR), for a given lost right-view texture frame[2] $\mathbf{x}_t^r$ at time instant t, we exploit temporal correlation in received neighboring frames $\mathbf{x}_{t-1}^r$ and $\mathbf{x}_{t+1}^r$ in time to interpolate the missing pixels in $\mathbf{x}_t^r$. Yet, unlike traditional TSR methods like [98] where only block-based *motion estimation* (ME) is performed, we exploit available depth information in the corresponding depth frames $\mathbf{z}_{t-1}^r$ and $\mathbf{z}_{t+1}^r$ to partition the missing texture block into foreground and background sub-blocks for separate ME, leading to a more accurate per-pixel motion field. Finally, when copying the reference sub-blocks to reconstruct the missing block in $\mathbf{x}_t^r$, depending on the sharpness of the sub-block boundary in the reference texture block, we optionally perform *overlapped motion compensation* (OMC) to synthesize a more naturally looking image.

The second method, denoted as DIBR, exploits the inter-view correlation between the received left-view texture frame $\mathbf{x}_t^l$ and missing frame $\mathbf{x}_t^r$. Then, given that most missing pixels in $\mathbf{x}_t^r$ have two recovery candidates (TSR and DIBR), we select the better candidate for each texture pixel at a patch level, where an image patch is a neighborhood of pixels with similar depth values. This ensures consistency of selected candidates within the same object. Through extensive experimentation, we demonstrate that our system outperforms a single-description / single-path transmission scheme by up to 5.5dB in PSNR of the synthesized intermediate view at the receiving client.

5.2 Multiple-path Free Viewpoint Video System

5.2.1 Free Viewpoint Video Streaming System

Our system is illustrated in Fig. 5.1. We assume that there are two disjoint network paths available for transmission of free viewpoint video content to the client. For example, a multi-homed wireless client can have two network interfaces such as 3G cellular

[2]Frame recovery for left-view texture frame $\mathbf{x}_t^l$ can be performed similarly. Due to its piecewise smooth characteristics, recovery of depth frame $\mathbf{z}_t^r$ is done using DIBR only as described in Section 5.4.

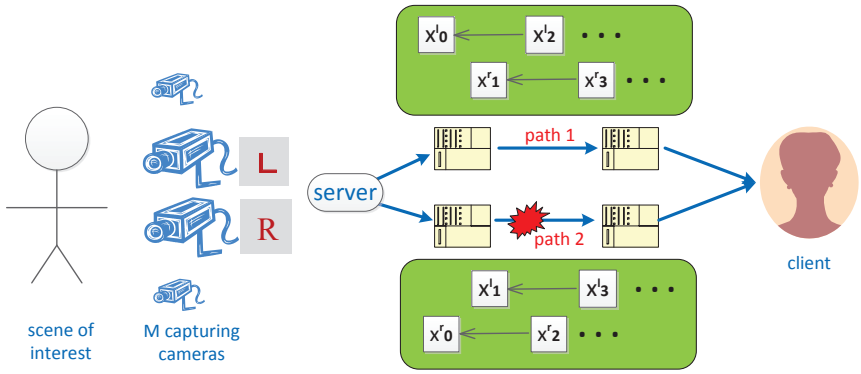


FIGURE 5.1: Overview of our streaming system for free viewpoint video encoded in two descriptions for transmission over two disjoint paths.

and 802.11 Wi-Fi that connect to two orthogonal communication networks [120]. Another example is a community of wireless clients [76] in proximity of each other that collaboratively pool their wireless network resources together for a high-priority task. Yet another example is multi-source video streaming [121], where the video content resides in both a remote server and a nearby peer who has cached the content and can help with the distribution. In any of these cases, the free viewpoint video content can be transmitted to the client simultaneously over two disjoint network paths. At the same time, we assume that the client sends periodic feedback to the sender(s) over these two paths, so that the sender(s) knows the intermediate virtual view requested at any time. The two disjoint network paths will in general be characterized by different transmission bandwidth and packet loss statistics. Since the paths are disjoint, packet loss events on one link are independent from loss events on the other.

We assume that each network path exhibits end-to-end burst packet loss characteristics modeled by a GE model [122].

5.2.2 Free Viewpoint Video Representation

We assume that the free viewpoint video content is encoded in the now popular *texture-plus-depth* format [57]. In a nutshell, an array of closely spaced cameras capture texture and depth maps (images) from different viewpoints (see [107] for an example camera setup). Depending on the intermediate virtual view currently requested by the client (based on currently tracked head position [106], for example), texture and depth maps

from the *two* nearest camera viewpoints (left and right views) will be encoded for transmission[3]. We further assume that the two transmitted views are rectified in a preprocessing step [124].

Using texture and depth maps from two captured views as references, a novel image as observed from an intermediate virtual view chosen by the client can be synthesized via DIBR. This is essentially a pixel-to-pixel mapping procedure that translates texture pixels in the reference camera views to the virtual view image, where the mapped locations are determined by known camera parameters and the corresponding depth pixel values. Spatial regions in the virtual view that are occluded by foreground objects and thus not visible in the reference views are called *disocclusion holes*. They are in general difficult to fill; there exist depth-based inpainting methods in the literature [125–127] that provide satisfactory solutions in typical cases. Using the two closest camera views as references for DIBR ensures that the sizes of the resulting disocclusion holes in the virtual image are small.

5.2.3 Multiple Description Construction

We encode texture and depth videos from the left and right views as follows. We first perform standard MC predictive video coding, such as H.264 [24], respectively on the odd and even frames of the left-view texture video, $\mathbf{x}_1^l, \mathbf{x}_3^l, \ldots$, and $\mathbf{x}_0^l, \mathbf{x}_2^l, \ldots$, thereby creating two streams $\mathbf{X}_o^l$ and $\mathbf{X}_e^l$. Similarly, we encode the odd and even frames of the left-view depth video, as well as the odd and even frames of the right-view texture and depth video, into the corresponding streams $\mathbf{Z}_o^l$, $\mathbf{Z}_e^l$, $\mathbf{X}_o^r$, $\mathbf{X}_e^r$, $\mathbf{Z}_o^r$ and $\mathbf{Z}_e^r$. This procedure of encoding even and odd frames separately into different streams is reminiscent of previous MDC schemes for single-view video [87]. Note that since the temporal distance between the consecutively coded frames is two (rather than one frame as in conventional video coding), our MDC results in a slightly larger source coding rate.

Note also that because a depth frame provides only geometric information for viewpoint image rendering and is not itself observed directly by users, how to select QPs for texture and depth maps for optimal synthesized view quality is a non-trivial problem [128]. We

[3]Using more than two captured views typically does not increase the synthesized view quality noticeably, while using only a single captured view for synthesis leaves large disocclusion holes, resulting in poor synthesized view quality, as shown for example in [123]. Thus, we also assume that two and only two captured views are transmitted.

will discuss our proposed QP selection for texture and depth videos in the left and right views in Section 5.5.

Given the encoded streams, we construct two descriptions D_1 and D_2 as follows. First, we bundle the streams $\mathbf{X}_e^l$, $\mathbf{Z}_e^l$, $\mathbf{X}_o^r$, and $\mathbf{Z}_o^r$ into description D_1; *i.e.*, D_1 is composed of the left-view even frames and right-view odd frames. Then, we bundle the remaining streams $\mathbf{X}_o^l$, $\mathbf{Z}_o^l$, $\mathbf{X}_e^r$, and $\mathbf{Z}_e^r$ into description D_2; *i.e.*, D_2 is composed of the left-view odd frames and right-view even frames. D_1 and D_2 are transmitted to the client via paths one and two, as illustrated in Fig. 5.1.

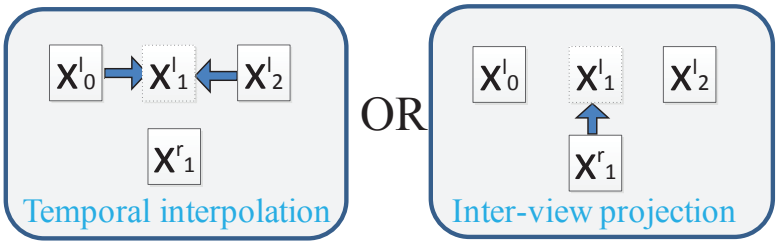


FIGURE 5.2: Illustration of the recovery procedure.

The descriptions are designed such that even if only one description is received, the client can reconstruct the missing frames of the other description by exploiting the inherent temporal and inter-view correlation that the descriptions feature. See Fig. 5.2 for an illustration. Specifically, for each pixel in a lost frame, we reconstruct two recovery candidates. The first candidate is reconstructed via TSR using neighboring temporal frames of the same view. The second candidate is reconstructed via DIBR using a frame of the same time instant in the opposing view. Given the recovery candidates, we then select the final reconstruction of the missing data at a patch level, where each image patch is a neighborhood of pixels with similar depth values. Doing so means we achieve reconstruction consistency among neighboring pixels of the same object.

5.3 Temporal Super-Resolution-based Frame Recovery

Let texture frame $\mathbf{x}_t^r$ be lost during transmission. The TSR recovery procedure comprises a number of computational steps that are outlined in Figure 5.3 and are explained next.

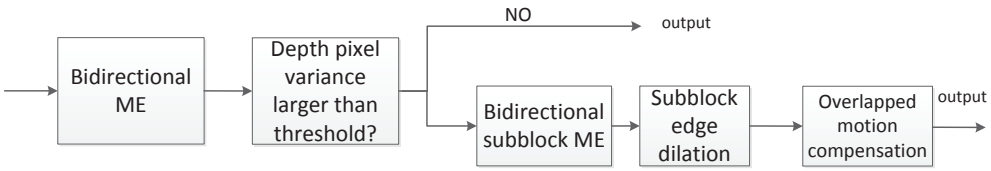


FIGURE 5.3: Flow diagram of the proposed TSR-based frame recovery method.

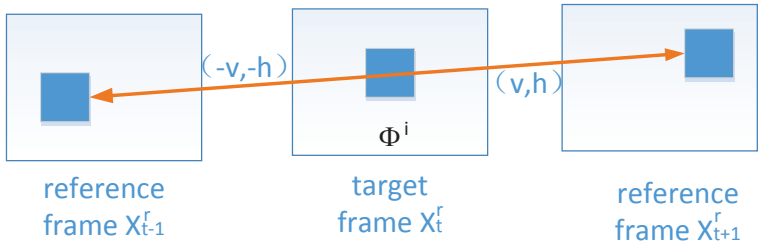


FIGURE 5.4: Bidirectional motion estimation (BME) to recover missing block in target frame $\mathbf{x}_t^r$ via block matching in neighboring temporal reference frames $\mathbf{x}_{t-1}^r$ and $\mathbf{x}_{t+1}^r$.

5.3.1 Bidirectional Motion Estimation

We first perform BME at the block level. Specifically, for each given non-overlapping $K \times K$ pixel block Φ_p, specified by its upper-left corner pixel $p = (i, j)$ in the target missing frame $\mathbf{x}_t^r$, we search for two similar blocks in the reference frames $\mathbf{x}_{t-1}^r$ and $\mathbf{x}_{t+1}^r$ at locations $(i - v, j - h)$ and $(i + v, +h)$, respectively. In other words, we search for the two best-matched blocks in $\mathbf{x}_{t-1}^r$ and $\mathbf{x}_{t+1}^r$ such that a *half* of their temporal motion vector (MV) will place the block at location p in frame $\mathbf{x}_t^r$. Fig. 5.4 shows an example of BME.

Assuming that the *sum of absolute differences* (SAD) is used as a matching criteria, the best MV (v_p, h_p) for block $\Phi_{(i,j)}$ in the target frame $\mathbf{x}_t^r$ is given by:

$$\begin{aligned}(v_p, h_p) = &\arg\min_{(v,h)} SAD\left(\mathbf{x}_{t-1}^r(\Phi_{(i-v,j-h)}),\ \mathbf{x}_{t+1}^r(\Phi_{(i+v,j+h)})\right) \\ &+\lambda\left(|v - \bar{v}_p| + |h - \bar{h}_p|\right) \end{aligned} \tag{5.1}$$

where $(\bar{v}_p, \bar{h}_p)$ is the weighted average of the MVs of the causal neighboring blocks of Φ_p. The additional regularization term in (5.1) enforces piecewise smoothness of the motion field. Note that the search is performed at 1/2-pixel precision, interpolated from

full-pixel resolution using bilinear filtering[4].

$\bar{v}_p$ is computed as

$$\begin{aligned} \bar{v}_p &= \frac{\sum_{q\in\mathcal{N}_p} w_q v_q}{\sum^{s}_{q\in\mathcal{N}_p} w_q}, \\ w_q &= \exp\left\{-\frac{|\bar{z}^r_t(\Phi_p) - \bar{z}^r_t(\Phi_q)|}{\sigma^2}\right\}, \end{aligned} \tag{5.2}$$

where $\mathcal{N}_p$ denotes the set of causal neighboring blocks of Φ_p, $\bar{z}^r_t(\Phi)$ denotes the arithmetic mean of depth values in block Φ of depth frame $\mathbf{z}^r_t$, and σ is a chosen parameter. $\bar{h}_p$ is written in the same form as $\bar{v}_p$ with h_q replacing v_q. Given unique MV (v_p, h_p) for block Φ_p in frame $\mathbf{x}^r_t$, we can compute the average of blocks $\mathbf{x}^r_{t-1}(\Phi_{i-v_p,j-v_p})$ and $\mathbf{x}^r_{t+1}(\Phi_{i+v_p,j+h_p})$, to reconstruct block Φ_p in $\mathbf{x}^r_t$.

Ideally, instead of block-level motion, *pixel-level* motion would provide more accurate information, since a given block can contain parts of more than one object with different motion vectors. However, finding pixel-level motion via optical flow techniques [129] is computationally expensive. To overcome the shortcomings of both block-based BME and optical flow, we propose an alternative *arbitrary-shaped sub-block BME* that uses the available information in depth frames $\mathbf{z}^r_{t-1}$ and $\mathbf{z}^r_{t+1}$.

Specifically, given a texture block in the reference frame $\mathbf{x}^r_{t-1}$, we first check if the variance of the corresponding depth block in depth frame $\mathbf{z}^r_{t-1}$ is large. If so, we partition the texture block into two sub-blocks along an edge similar to the corresponding depth block discontinuity. The partition edge in the reference texture block in frame $\mathbf{x}^r_{t-1}$ is then translated to a partition in the target block in missing frame $\mathbf{x}^r_t$, dividing the target block into sub-blocks. We then perform sub-block BME following the previously described BME procedure. Finally, OMC is optionally performed to avoid sharp sub-block boundaries in the reconstructed block.

5.3.2 Texture Block Partitioning

Given texture map $\mathbf{x}^r_{t-1}$ and depth map $\mathbf{z}^r_{t-1}$, block support Φ_p at pixel p—denoted by a sequence of offsets from p, *i.e.*, $(0,0), (0,1), \ldots, (K-1, K-1)$—can be partitioned into

[4] *Bilinear* interpolation is also used in H.264 [24] to increase the resolution from half-pel to 1/4-pel for a more accurate ME. For complexity reasons, we perform BME only at half-pel resolution.

two non-overlapping sub-block supports Φ_p^1 and Φ_p^2 (*e.g.*, foreground and background objects), where $\Phi_p = \Phi_p^1 \cup \Phi_p^2$ and $\emptyset = \Phi_p^1 \cap \Phi_p^2$. Hence the texture pixel block $\mathbf{x}_{t-1}^r(\Phi_p)$ is also the union set $\mathbf{x}_{t-1}^r(\Phi_p^1) \cup \mathbf{x}_{t-1}^r(\Phi_p^2)$.

The first step of macroblock partitioning is to compute the variance of the corresponding depth block $\mathbf{z}_{t-1}^r(\Phi_p)$. If the variance is smaller than a pre-defined threshold T_d (indicating how likely the block contains more than one object), the block will not be partitioned.

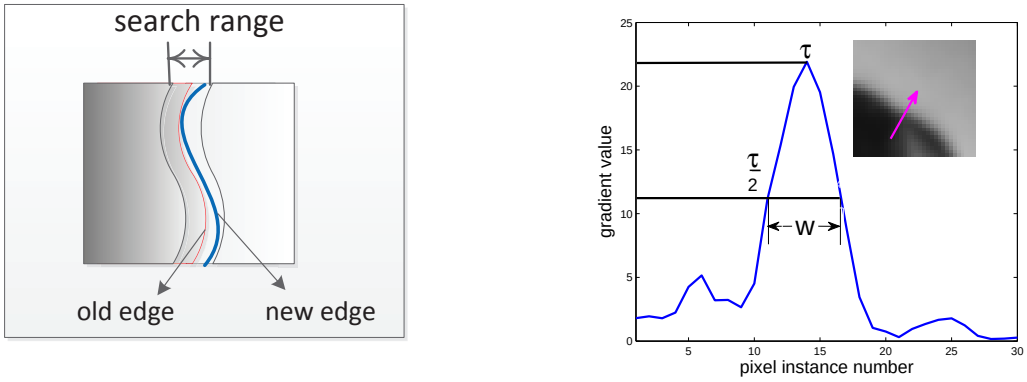
If the variance is larger than T_d, the depth block will be partitioned into two sub-blocks, each with depth pixels above and below the arithmetic mean $\bar{z}_{t-1}^r(\Phi_p)$, respectively. Assuming block $\mathbf{z}_{t-1}^r(\Phi_p)$ contains only one foreground object (small depth value) in front of a background (large depth value), this method can segment pixels into two correct sub-blocks. This statistical approach has been shown to be robust and has low complexity [130]. Finally, we perform a morphological closing to ensure that each partitioned sub-block represents a contiguous region.

(a) Kendo (b) Pantomime (C) Pantomime

FIGURE 5.5: Illustration showing texture and depth edges may not be perfectly aligned, where the depth edges (white lines) are detected using a 'Canny' edge detector.

In the ideal case, the texture map contains a superset of edges of the depth map. Thus, one can simply reuse the computed depth sub-block boundary for partitioning the texture block as well. However, a known problem in the texture-plus-depth representation [57] is that edges in texture and depth maps may not be perfectly aligned, due to noise in the depth acquisition process. Fig. 5.5 shows example spatial regions of texture maps overlaid with edges detected in the corresponding depth maps using a Canny edge detector (white lines). One can clearly see that the texture and depth edges are not perfectly aligned.

To circumvent the edge misalignment problem, we perform a simple dilation process. Specifically, we first copy the computed sub-block boundary to the texture block. We next perform edge detection in the texture block. Then, we perform dilation of the depth boundary—thickening of the edge—until a texture edge is found. Fig. 5.6 shows an example of dilation.



(a) depth edge dilation (b) boundary illustration

FIGURE 5.6: (a) edge dilation to identify corresponding texture edge for texture block partitioning. (b) a blurred boundary and the corresponding gradient function across boundary.

Using the discovered texture edge, the reference block in frame $\mathbf{x}^r_{t-1}$ is also partitioned into two sub-blocks. Then, the corresponding full block in frame $\mathbf{x}^r_t$ can be partitioned into two sub-blocks, as well, by copying the texture edge in $\mathbf{x}^r_{t-1}$ using the MVs computed in Section 5.3.1.

5.3.3 Overlapped Sub-block Motion Estimation

For each partitioned sub-block Φ^i_p in $\mathbf{x}^t_t$, we find its best match in reference frames $\mathbf{x}^r_{t-1}$ and $\mathbf{x}^r_{t+1}$, as described in Section 5.3.1. The only difference is that now we use sub-blocks instead of full blocks. MVs for each sub-block are computed.

Optionally, we can now perform OMC for better reconstruction of the target block. Specifically, when copying a best-matched sub-block from the reference frame to the missing block in the target frame, we copy the sub-block plus l pixels across the sub-block boundary. The extra copied pixels will be alpha-blended with overlapping pixels copied from the opposing sub-block. See Fig. 5.7 for an illustration.

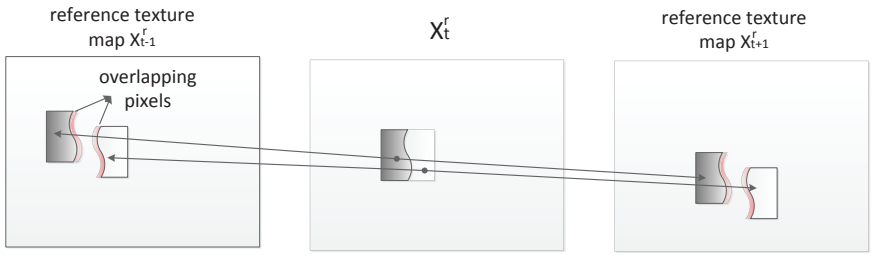


FIGURE 5.7: Illustration of overlapping sub-blocks.

The width of the overlapping region l is determined by the sharpness of the texture edge (sub-block boundary) in the reference block of frame $\mathbf{x}^r_{t-1}$. The key insight here is that unlike a depth map which always has sharp edges, object boundaries in the texture map can be blurred due to out-of-focus, motion blur, etc. On the other hand, sub-block motion compensation tends to result in sharp sub-block boundaries. So to mimic the same blur across a boundary in the reference block in frame $\mathbf{x}^r_{t-1}$, we first compute a texture gradient function for a line of pixels in the reference block perpendicular to the sub-block boundary [131].

We then compute the width of the plateau corresponding to the sub-block boundary, which we define as the number of pixels across the plateau at half the peak τ of the gradient plateau. Finally, we set l to be a linear function of the computed width w (*i.e.* more blur, more overlap) as follows:

$$l = round\,(\varepsilon\, w)\ , \tag{5.3}$$

where ε is a chosen parameter. See Fig. 5.6(b) for an example of a blurred sub-block boundary, its corresponding gradient function across the boundary, and the width of the plateau w.

5.4 DIBR-based Frame Recovery and Pixel Selection Framework

Having described how using TSR we can reconstruct a recovery candidate for each pixel in a missing texture frame $\mathbf{x}^r_t$, we now discuss how using DIBR we can reconstruct another recovery candidate. In particular, in Sections 5.4.1 and 5.4.2 we first discuss

how we reconstruct the missing depth map $\mathbf{z}_t^r$, which is easier given its known piecewise smooth characteristics. We then discuss how the corresponding texture map $\mathbf{x}_t^r$ can be reconstructed using the recovered depth map $\mathbf{z}_t^r$ in Section 5.4.3. Finally, we propose a patch-level candidate selection scheme for the final missing texture map reconstruction by choosing between the two recovery candidates.

5.4.1 Depth Map Reconstruction

We first synthesize the missing right-view depth map $\mathbf{z}_t^r$ via DIBR [118] using the corresponding left-view depth map $\mathbf{z}_t^l$. Specifically, given that the captured camera views are rectified [124], each depth pixel $z_t^l(x, y)$ of row x and column y in the left-view depth map is mapped to a corresponding pixel $z_t^r(x, y')$ in the right-view depth map, where the new column index y' is computed as:

$$y' = y - round\left(\frac{1}{z_t^l(x, y)} * \gamma\right) \tag{5.4}$$

From (5.4), we note that the horizontal disparity (pixel translation) is governed by $1/(z_t^l(x, y))$ and the shift parameter γ, which depends on the physical distance between the two capturing cameras.

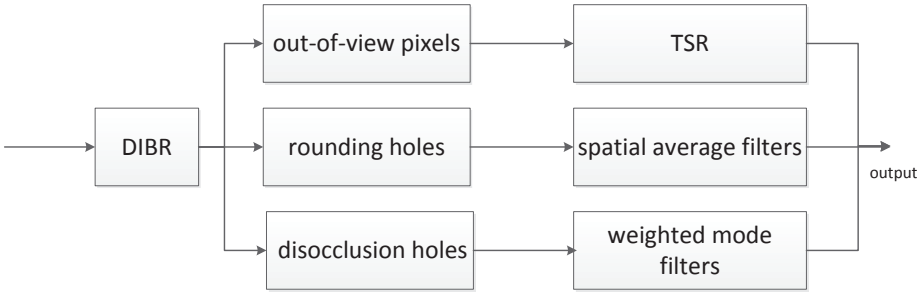


FIGURE 5.8: Flow chart of the proposed depth map recovery method.

In general, depth pixels synthesized via DIBR are more reliable than color pixels, because while color pixels of the same object surface can contain different values at different viewpoints if the surface is non-Lambertian [102], depth pixels are not affected by the object's surface reflectance property. Hence a depth pixel mapped from the left view to the right view is very likely to be correct. To recover all pixels in the right-view depth map, only missing pixels need to be completed using neighboring spatial and

temporal information. We discuss in detail how the missing pixels are filled, in this section. Fig. 5.8 shows the flow diagram of our depth map reconstruction procedure.

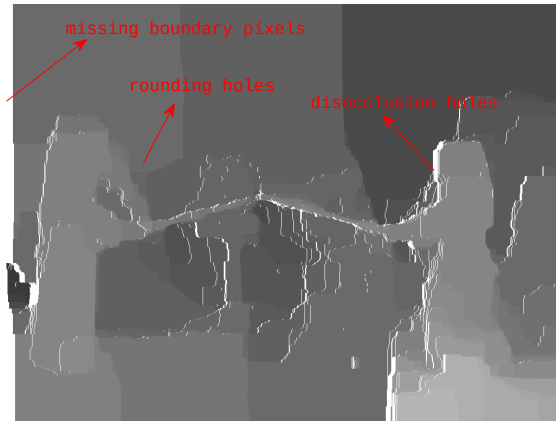


FIGURE 5.9: Three kinds of holes in a synthesized depth map.

DIBR's simple pixel-to-pixel translational mapping results in three types of pixel holes illustrated in Fig. 5.9. First, there are *out-of-view pixels* in the right-view depth map $\mathbf{z}_t^r$ that are out-of-view in the left-view depth map $\mathbf{z}_t^l$. Second, due to the rounding operation in (5.4), there might not be any left-view depth map pixels that map to a given pixel location in a right-view depth map. These are called *rounding holes*. Finally, there are spatial regions in the synthesized right-view image that are occluded by foreground objects and therefore not visible in the reference view. These are called *disocclusion holes*.

Due to the operation of rounding to the nearest pixel column, carried out in (5.4), the thereby created rounding holes are characterized by being narrow in width. Because neighboring depth pixels around a rounding hole usually belong to the same physical object, they have very similar depth values. Hence, simple spatial average filtering can adequately fill in these rounding holes.

By definition, out-of-view pixels in $\mathbf{z}_t^r$ are not in the field of view in depth map $\mathbf{z}_t^l$, and so $\mathbf{z}_t^l$ contains no information to reconstruct out-of-view pixels in $\mathbf{z}_t^r$. Hence we fill out-of-view pixels in $\mathbf{z}_t^r$ by reusing the MVs computed in TSR for the texture candidates used to copy depth pixels from matched blocks in $\mathbf{z}_{t-1}^r$ and $\mathbf{z}_{t+1}^r$ to $\mathbf{z}_t^r$. We focus our discussion on the filling of disocclusion holes next.

5.4.2 Filling of Disocclusion Holes in a Depth Map

First, using MVs computed for a texture map during TSR described in Section 5.3, we initialize the depth values in these disocclusion holes by copying the corresponding reference blocks in neighboring temporal depth frames $\mathbf{z}^r_{t-1}$ and $\mathbf{z}^r_{t+1}$. The initialized depth values may not lead to a piecewise smooth solution. Thus, we next employ a *weighted mode filter* (WMF) [132] to sharpen the overly smoothed pixels.

Mathematically, for a pixel location p with neighbors $q \in \mathcal{N}_p$, we first compute a *relaxed histogram* $H(p,d)$ with index d as follows:

$$H(p,d) = \sum_{q \in \mathcal{N}_p} G_s(p-q) \;\; G_f(z^r_t(p) - z^r_t(q)) \;\; G_r(d - z^r_t(q)) \tag{5.5}$$

where $G_s(p-q)$ is a Gaussian term with the geometric distance between pixel locations p and q as its argument, $G_f(z^r_t(p) - z^r_t(q))$ is a Gaussian term based on the photometric distance between depth values $z^r_t(p)$ and $z^r_t(q)$, and $G_r(d - z^r_t(q))$ is a Gaussian term based on the error between bin index d and $z^r_t(q)$. Note that G_s and G_f are similarly computed in *bilateral filter* [133].

Having computed $H(p,d)$ for different bin indices d, the new depth value $z^r_t(p)$ is the index with the largest histogram value:

$$z^r_t(p) = \arg\max_d H(p,d) \tag{5.6}$$

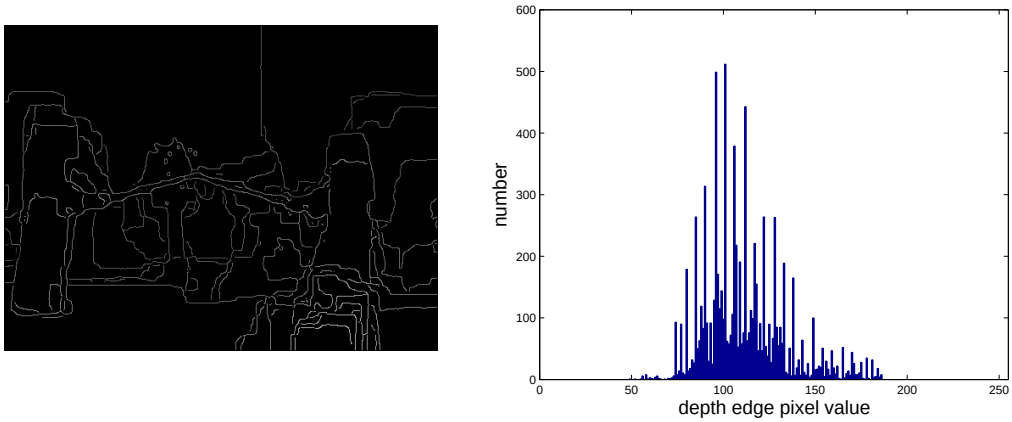
5.4.3 Depth Image Based Rendering for Texture Maps

We apply the same procedure we used for reconstructing depth map $\mathbf{z}^r_t$, to generate recovery candidates for texture map $\mathbf{x}^r_t$ via DIBR. Rounding holes are also filled using spatial average filtering. Out-of-view pixels and disocclusion holes are left unfilled. They make up a small percentage of the total pixels, and these pixels will be reconstructed via TSR exclusively. We now discuss how we select between TSR candidates and DIBR candidates for the rest of the texture pixels.

5.4.4 Selection of Recovery Candidates

Given the constructed recovery candidates for pixels in a missing texture frame $\mathbf{x}_t^r$, we now describe a procedure to select candidates at a patch level. A patch roughly corresponds to a depth layer of a physical object, so that selecting candidates consistently in a patch would lead to a visually pleasing reconstructed image.

5.4.4.1 Image Segmentation



(a) detected edges

(b) depth histogram

FIGURE 5.10: Detected edges and depth histogram of detected edges for frame 6, view 3 of the `Kendo` sequence.

We first segment a missing texture map $\mathbf{x}_t^r$ into patches based on the reconstructed depth map $\mathbf{z}_t^r$. The algorithm is a variant of the Lloyd's algorithm in *vector quantization* (VQ) [134]. To initialize a segmentation, we first construct a histogram of depth values for the detected edge pixels (edges are detected using a Canny edge detector) and identify the K highest peaks $\hat{z}_k$'s. See Fig. 5.10 for an example depth image with detected edges in white and corresponding depth histogram of detected edge pixels. For each pair of adjacent peaks $\hat{z}_k$ and $\hat{z}_{k+1}$ in the histogram, we identify a depth value that is a minimum between the peaks and denote it as a boundary b_k. Using $K-1$ boundary values b_k's, we can segment the image into at least K patches, where a patch is a set of contiguous pixels with depth values within two boundaries b_k and b_{k+1}. Fig. 5.11 shows resulting patches (marked in brown) between two boundary points b_k and b_{k+1} after the segmentation.

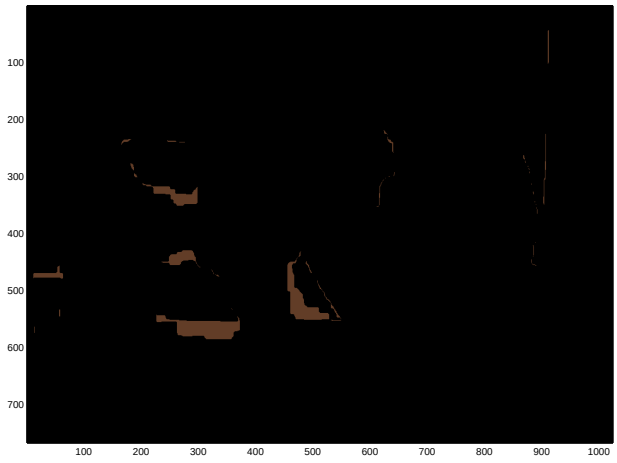


FIGURE 5.11: Patches (in brown) between two boundary points after segmentation.

Having initialized patches, we then perform the following two steps, alternately, until convergence. In the first step, we solve for the *centroid* for each patch, which is the depth value that minimizes the MSE between the centroid and the depth values in the patch. In the second step, given the computed centroids of different patches, each pixel on the border of a patch can be associated with the centroid of a neighboring patch such that its squared error is further minimized. The iteration ends when neither of the two steps can further decrease the MSE.

5.4.4.2 Recovery Candidate Selection

To select recovery candidates between TSR and DIBR for a given patch with centroid c, we examine frames from the most recent correctly received descriptions to see if patches with centroids close to c have smaller reconstruction errors using TSR or DIBR. The idea is that patches with similar depth centroids are more likely to represent the same physical objects. Assuming the same object exhibits similar motion patterns (which affect the performance of TSR) and surface reflectance properties (which affect the performance of DIBR) over time, previous frames provide valuable side information for good selection of recovery candidates for a current frame.

5.5 Data Transport Optimization

Having discussed the description recovery method in the previous sections, when the client receives only one description out of two, we describe now how we optimally select the source and channel coding rates for each description, given the available bandwidth and packet loss statistics associated with a transmission path, such that the client's expected video quality is maximized.

Within one description, the video frames (texture or depth) comprising a GOP are split into N sub-groups each with the same number of frames, as shown in Figure 5.12. Let n_i denote the total number of packets (source plus channel) that are transmitted for sub-group i. k_i denotes the number of source packets only for sub-group i, where texture and depth maps in the description are encoded using different QPs (to be discussed). $n_i - k_i$ packets in sub-group i are for FEC packets, generated as linear combinations of the corresponding k_i source packets. Correct delivery of any k_i of n_i transmitted packets will recover all k_i source packets.

For simplicity, we assume also that the playback deadline of the first frame of each sub-group is the transmission deadline for the whole sub-group. We now formulate an optimization problem for selecting QP Q, for every video frame comprising packet n_i of sub-group i.

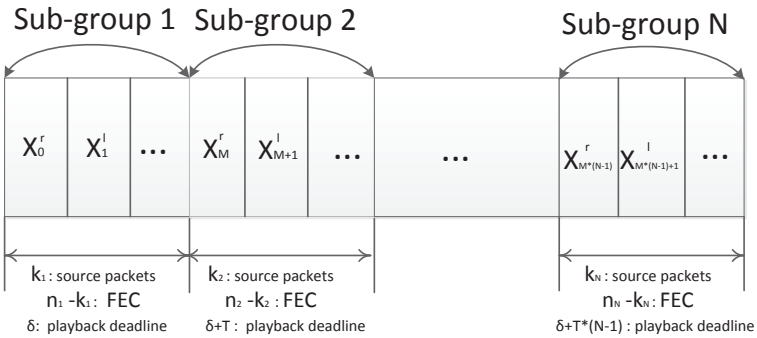


FIGURE 5.12: Illustration of the video frame grouping.

The preliminaries needed for calculating the probabilities of correctly decoding the video frames given a GE packet loss model has been introduced in Section 4.4.2. We then show how to derive our objective function. Two optimization algorithms are described thereafter.

5.5.1 System Constraints

Given that the transmission deadline for the first frame in a sub-group i is the delivery deadline for whole sub-group, we can derive the maximum number of packets l_j that can be transmitted by the first j sub-groups as follows:

$$l_j = (\delta + T \times (j-1))B \tag{5.7}$$

where δ is the initial buffer time, T is the playback duration of the frames in each sub-group, and B is the bandwidth of the transmission path in number of packets per second. This means that the total number of packets $\sum_{j=1}^{i} n_j$ expended for transmission of frames up to and including sub-group i cannot exceed the budget l_i, i.e.,

$$\sum_{j=1}^{i} n_j \leq l_i, \qquad \forall i \in \{1, \ldots, N\} \tag{5.8}$$

Otherwise, we assume that sub-group i is not correctly delivered since the transmitted packets do not meet the required playback deadline.

5.5.2 Probability of Correct Decoding

Due to the predictive nature of video coding, the probability β_i of correctly decoding the frames in sub-group i is a product of: i) the probability α_i of timely and correct recovery of all source packets in sub-group i, and ii) the probability β_{i-1} of correctly decoding the frames in the previous sub-group $i-1$. Thus, we can write β_i as follows:

$$\beta_i = \beta_{i-1} * \alpha_i \tag{5.9}$$

We compute β_{i-1} as follows:

$$\beta_{i-1} \approx \prod_{j=1}^{i-1} \alpha_j \tag{5.10}$$

The assumption of independence of α_j's is an approximation; since the GE packet loss model has memory, and the GE state in which last packet was transmitted in sub-group $i-1$ can affect the probability of correct packet transmission of the following

sub-group i. However, if the number of transmitted packets in each sub-group is large, the approximation is nonetheless a good one.

We use the same ideas to calculate α_i as introduced in Section 4.4.3 and Section 4.4.4. For each sub-group i, the initial state of the G-E model at transmission could be good or bad with different probabilities. We write the probability α_i of correctly recovering all source packets in sub-group i as a weighted sum of α_i^G and α_i^B, which are the probabilities of correctly receiving at least k_i of n_i transmitted packets, given that packet transmission begins at a good or bad state, respectively:

$$\alpha_i = \left(\frac{q}{p+q}\right)\alpha_i^G \; + \; \left(\frac{p}{p+q}\right)\alpha_i^B \tag{5.11}$$

Assuming first that transmission starts in the good state, m of n_i total packets can be transmitted in good state with probability $S(m, n_i)$. Source packets in sub-group i can be successfully recovered if at least k_i of n_i transmitted packets are correctly delivered. Among r received packets, $r \geq k_i$, r_G can be delivered packets in good state while $r - r_G$ can be delivered packets in bad state. We can hence write α_i^G as:

$$\alpha_i^G = \sum_{m=0}^{n_i} S(m, n_i) \sum_{r=k_i}^{n_i} \sum_{r_G=0}^{r} P_G(r_G, m)\; P_B(r - r_G, n_i - m) \tag{5.12}$$

where $P_G(x, y)$ and $P_B(x, y)$ are the probabilities of exactly x delivered packets in y iid trials, in the good and bad states, respectively. These quantities can be computed easily using binomial expansion and the packet loss probability g and b, respectively for the good and bad states. α_i^B can be derived similarly.

5.5.3 Optimization Problem

The objective we selected for optimization is the rendered virtual view image quality, where the virtual viewpoint chosen for evaluation is the middle view between left and right captured views. Inserting superscript R to denote the right path[5], let β_i^R be the probability of correctly decoding frames in sub-group i of the *right* path. Furthermore, denote d_i to be the rendered virtual view's quality, if frames of sub-groups i of both paths are correctly decoded, and d_i^R the reandered view quality if frames of sub-group

[5]Note that unlike previous superscripts l and r that denote left and right captured views, we use here L and R to denote left and right transmission paths.

i of right path only are correctly decoded. d_i and d_i^R are dependent on the QPs used for the two paths: Q_T^R and Q_D^R for texture and depth maps of the right path, and Q_T^L and Q_D^L for texture and depth maps of the left path. We can now write the expected synthesized view quality for sub-group i as:

$$\begin{aligned} D_i &= \beta_i^R \beta_i^L d_i(Q_T^L, Q_D^L, Q_T^R, Q_D^R) + \\ &\quad \beta_i^R (1-\beta_i^L) d_i^R(Q_T^R, Q_D^R) + (1-\beta_i^R) \beta_i^L d_i^L(Q_T^L, Q_D^L) \end{aligned} \tag{5.13}$$

We assume here that having frames lost in both descriptions (simultaneous burst loss events on two disjoint transmission paths) is rare and hence not considered.

We can now formally define the optimization problem as follows. The optimization variables are: i) QPs Q_T^R, Q_D^R, Q_T^L, Q_D^L, and ii) the number of transmitted packets n_i^L and n_i^R for each sub-group i in each path. The optimization is subject to the system constraints (5.8):

$$\max_{Q_T^R, Q_D^R, Q_T^L, Q_D^L, \{n_i^L\}, \{n_i^R\}} \sum_i D_i \quad \text{s.t.} \begin{array}{l} \sum_{j=1}^i n_j^L \le l_i^L, \ \forall i \in \{1, \ldots, N\} \\ \sum_{j=1}^i n_j^R \le l_i^R, \ \forall i \in \{1, \ldots, N\} \end{array} \tag{5.14}$$

5.5.4 Optimization Algorithms

Solving (5.14) is complicated, as it involves variables from both transmission paths. We thus elect to solve for variables in one path at a time with variables in the other path fixed, then iterate until convergence.

Fixing a given set of variables in a single path (say the left path), we also iterate between right path QPs Q_T^R, Q_D^R and rates $\{n_i^R\}$ until convergence. When n_i^R's are fixed, we find the optimal QPs Q_T^R and Q_D^R as follows. We alternately perturb Q_T^R and Q_D^R locally in an attempt to increase the objective (5.14), while respecting the transmission rate constraints. We stop when no further attempt to increase the objective is possible. Given there are only two QPs, this iteration takes little time and converges quickly.

We now discuss two proposals for finding $\{n_i^R\}$, when QPs Q_T^R and Q_D^R are fixed. The first proposal finds the optimal n_i^R's, but has high complexity. The second proposal solves the problem approximately, but exhibits lower computational complexity. Alg. 1 outlines the overall optimization procedure.

5.5.4.1 Dynamic Programming Algorithm

One can search for the optimal n_i^R's to (5.14), for fixed QPs, using the following recursive algorithm. Let $\Delta_i^R(m)$ be the maximum quality for sub-group i to N, given that m total packets were transmitted for previous sub-groups 1 to $i-1$ *and* the previous groups are all decoded correctly. We know sub-group i must transmit at least k_i^R source packets and no more than $l_i^R - m$ total packets to observe the system constraint (5.8). We can thus write $\Delta_i^R(m)$ recursively as follows:

$$\Delta_i^R(m) = \max_{n_i^R \in \{k_i^R, \ldots, l_i^R - m\}} \begin{array}{l} \alpha_i^R(n_i^R) \ \left[D_i^R + \Delta_{i+1}^R(m + n_i^R)\right] \\ + \ \left(1 - \alpha_i^R(n_i^R)\right) \ \sum_{j|j \geq i} \beta_j^L d_j^L \end{array} \tag{5.15}$$

where correct recovery probability $\alpha_i^R(n_i^R)$ for sub-group i is a function of the number of transmitted packets n_i^R only, and D_i^R, the contribution from sub-group i of the right path, is $D_i^R = \beta_i^L d_i + (1 - \beta_i^L) d_i^R$ from (5.13).

Initial call $\Delta_1^R(0)$ would return the optimal solution to (5.14).

We note that the solution to $\Delta_i^R(m)$ can be stored in entry (i, m) of a DP table Γ, so that a repeated call to the sub-routine $\Delta_i^R(m)$ can be simply looked up, instead of being actually computed fully. Thus, the complexity of (5.15) is bounded by the size of the DP table Γ multiplied by the complexity of computing each table entry: $O(N \left(\sum_{i=1}^N l_i^R\right) \left(\max_{i=1}^N l_i^R\right))$.

5.5.4.2 Branch and Bound

Given fixed QPs, using (5.15) to find the optimal n_i^R's can still be expensive. We thus now present modifications to (5.15) using a BB method to further limit the search space.

We first compute the objective (5.13) for a naïve selection of n_i^R's, *e.g.*, equal loss protection where the proportion of FEC packets employed for each sub-group relative to source packets, $(n_i^R - k_i^R)/k_i^R$, is roughly the same for all sub-groups. We denote its objective value as D^e.

When (5.15) is called, for each possible value n_i^R, we first compute an *upper bound* $\Delta_i^{R,u}(m, n_i^R)$, which is the upper limit of quality given n_i^R is chosen for sub-group i.

$\Delta_i^{R,u}(m, n_i^R)$ can be computed recursively similar to (5.15), but without any search for the optimal n_i^R's:

$$\begin{aligned} \Delta_i^{R,u}(m, n_i^R) &= \alpha_i^R(n_i^R) \left[D_i^R + \Delta_{i+1}^{R,u}(m + n_i^R)\right] \\ &+ \left(1 - \alpha_i^R(n_i^R)\right) \sum_{j|j\geq i} \beta_j^L d_j^L \end{aligned} \tag{5.16}$$

$$\begin{aligned} \Delta_i^{R,u}(m) &= \alpha_i^R(l_i^R - m) \left[D_i^R + \Delta_{i+1}^{R,u}(m)\right] \\ &+ \left(1 - \alpha_i^R(n_i^R)\right) \sum_{j|j\geq i} \beta_j^L d_j^L \end{aligned} \tag{5.17}$$

In words, (5.16) states that using the selected n_i^R yields recovery probability $\alpha_i^R(n_i^R)$ and increases the argument passed to future sub-groups by n_i^R. In contrast, (5.17) states that using *all* permissible packets $l_i^R - m$ for sub-group i will yield correct delivery probability $\alpha_i^R(l_i^R - m)$, *but* we do not increase the argument passed to future sub-groups to seek an upper-bound. Thus, the returned objective value for $\Delta_i^{R,u}(m, n_i^R)$ is from a selection of n_i^R's that may not be feasible (may not observe constraints (5.8)), and therefore is a *super-optimal* solution.

We use the upper bound $\Delta_i^{R,u}(m, n_i^R)$ as follows. If $\Delta_i^{R,u}(m, n_i^R) < D^e$, then we know that n_i^R cannot lead to a solution that is better than our naïve solution, and hence there is no need to recursively compute $\Delta_{i+1}^R(m + n_i^R)$ in (5.15), thereby reducing the computation cost.

Similarly, we can also compute the *lower bound* $\Delta_i^l(m, n_i)$, for each selected n_i in (5.15):

$$\begin{aligned} \Delta_i^{R,l}(m, n_i^R) &= \alpha_i^R(n_i^R) \left[D_i^R + \Delta_{i+1}^{R,l}(m + n_i^R)\right] \\ &+ \left(1 - \alpha_i^R(n_i^R)\right) \sum_{j|j\geq i} \beta_j^L d_j^L \end{aligned} \tag{5.18}$$

$$\begin{aligned} \Delta_i^{R,l}(m) &= \alpha_i^R(l_i^R - r) \left[D_i^R + \Delta_{i+1}^{R,l}(m + r)\right] \\ &+ \left(1 - \alpha_i^R(n_i^R)\right) \sum_{j|j\geq i} \beta_j^L d_j^L \,. \end{aligned} \tag{5.19}$$

The definition of $\Delta_i^{R,l}(m, n_i^R)$ in (5.18) is analogous to that of $\Delta_i^{R,u}(m, n_i^R)$ in (5.16). (5.19) returns a performance point when n_i^R is chosen randomly from the feasible range set $\{k_i^R, \ldots, l_i^R - m\}$. Note though that unlike the upper bound in (5.16), the solution produced by the lower bound in (5.18) is guaranteed to be feasible.

We use the computed lower bound $\Delta_i^{R,l}(m, n_i^R)$ as follows. If the difference between the upper bound $\Delta_i^{R,u}(m, n_i^R)$ and the lower bound $\Delta_i^{R,l}(m, n_i^R)$ is smaller than a threshold δ, then the permuted random solution produced by (5.18) is already good enough. Then, again there is no need to recursively compute $\Delta_{i+1}^R(m + n_i^R)$, and we can simply return the computed random solution instead. This also leads to computational savings.

Algorithm 1 Transport optimization

1: Assuming Q_T^R, Q_D^R, Q_T^L, Q_D^L are equal and use dynamic programming or branch and bound to compute the optimal n_i^R and n_i^L, let $\sum_i D_i^{old} = \sum_i D_i$
2: Fix Q_T^R and Q_D^R, alternately perturb Q_T^L and Q_D^L until there is no gain in the objective (5.14)
3: Fix the Q_T^L and Q_D^L computed in step 2, alternately perturb Q_T^R and Q_D^R until there is no gain in the objective (5.14), mark the newly computed objective as $\sum_i D_i^{new}$
4: **if** $\sum_i D_i^{new} \leq \sum_i D_i^{old}$ **then**
5: Exit (we have converged)
6: **else**
7: $\sum_i D_i^{old} = \sum_i D_i^{new}$
8: Return to Line 2
9: **end if**

5.6 Experimentation

5.6.1 Experimental Setup

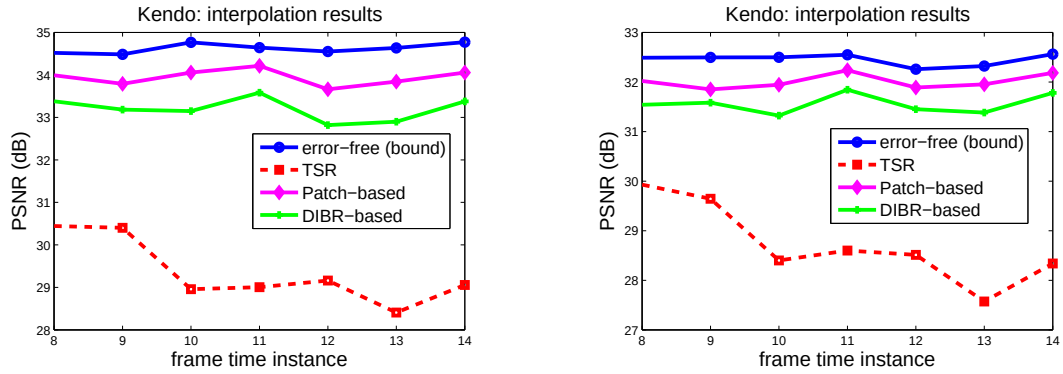
We evaluate the performance of our system, denoted as `Patch-based`, via extensive experiments. We used the 30fps MPEG free viewpoint test sequences `Kendo` and `Pantomime` from Nagoya University, where the texture and depth signals were encoded using H.264 JM18.0. The spatial resolution of `Kendo` and `Pantomime` is 512×384 and 640×480, respectively. The MTU in the transmission network was set to 1500 bytes. Each GOP had 30 frames and was divided into three sub-groups. The initial video buffering time was set to 0.4s.

5.6.2 Lost Frame Recovery

We compare our frame recovery scheme to two competing schemes: `DIBR-based` and `TSR`. `DIBR-based` is the scheme proposed in [51], which recovers the lost texture and depth pixels first using DIBR, and then fills the remaining missing pixels using TSR. `TSR` recovers missing pixels using TSR only. For TSR in `DIBR-based`, `TSR`, and `Patch-based`,

the block size was set to be 4×4, and the search was performed in 1/2-pixel accuracy. `error-free (bound)` is the synthesized intermediate view quality when both the left and right views are correctly delivered.

The recovery performance of these schemes on the content `Kendo` is shown in Fig. 5.13. View 1 and view 3 were the left and right views respectively during the experiment, and view 2 was used as the synthesized intermediate view. The x-axis denotes the frame number (index), and the y-axis measures the quality of the synthesized middle view. In Fig. 5.13(a), uncompressed video was used, and the video sequences used in Fig. 5.13(b) were encoded with QP=40.

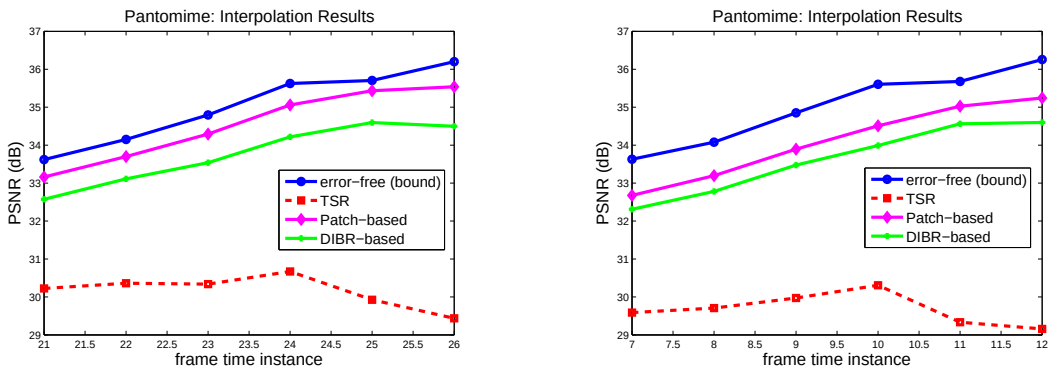


(a) uncompressed video (b) video at QP=40

FIGURE 5.13: Lost frame recovery results using different recovery methods for `Kendo`.

We observe that our proposed scheme `Patch-based` outperformed `DIBR-based` by up to 1.1dB. This is due to the more accurate sub-block motion estimation method and patch-level candidate selection method. Further, `Patch-based` outperformed `TSR` by up to 4.3dB. Comparing Fig 5.13(a) and (b), we see that larger QP leads to worse synthesized view quality as expected, but the performance trend remains consistent.

We also conducted comparison experiments for `Pantomime`. These results are shown in Fig. 5.14(a) and (b). In Fig. 5.14(a), uncompressed video was used, and the video sequences used in Fig. 5.14(b) were encoded with QP set to 23. We observe similar performance as for `Kendo`, where here `Patch-based` outperforms `DIBR-based` by up to 1.04dB.



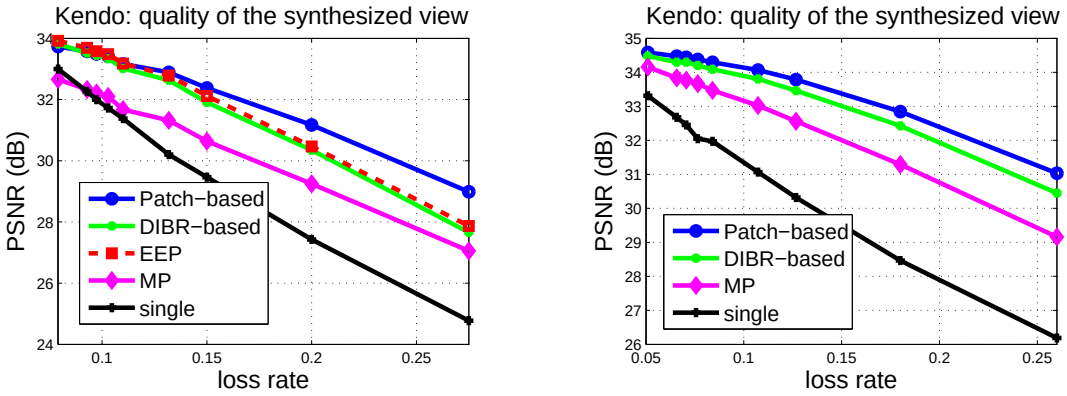
(a) uncompressed video (b) video at QP=23

FIGURE 5.14: Lost frame recovery results using different recovery methods for `Pantomime`.

5.6.3 Video Streaming

We conducted streaming experiments involving six competing schemes: `Patch-based`, `Patch-based SQP`, `single`, `DIBR-based`, `EEP`, and `MP`. `single` stands for the state-of-the-art single path / single description video transmission. Left- and right-view frames were sent in succession. At streaming time, the server will vary the amount of source packets by choosing the best source and channel coding rates via an exhaustive search. `Patch-based SQP` is a modified version of `Patch-based`, where the same QP is used for encoding of texture and depth maps on each path. `DIBR-based`, `EEP` and `MP` used two paths for video delivery, but `DIBR` used the `DIBR-based` recovery scheme to recover frames lost in the missing description, and `MP` used `TSR` as the recovery scheme. `EEP` used the same frame recovery scheme as `Patch-based`, but with equal error protection, which means the FEC packets were equally allocated to each subgroup. In `MP`, FEC packets were allocated to each subgroup optimally via an exhaustive search. `DIBR`, `EEP`, and `MP` used optimized QP for source coding. *Frame freeze* was used for the incorrectly decoded video frames, *i.e.* the user will play back the last correctly decoded frames if both descriptions are not correctly received.

We first set the bandwidth for each path in the multi-path transmission scenario to be 400 kbps, and `single` had the combined bandwidth of the two paths. *i.e.*, 800 kbps. The streaming results are shown in Fig. 5.15. In Fig. 5.15(a), the GE parameters assumed were g=0.05, b=0.95, q=0.1 with p varied throughout the simulation to induce different loss rates. We observe that our proposed scheme outperformed all competing schemes, and the transmission schemes using multi-path outperformed `single`, although the latter



(a) channel 1 (b) channel 2

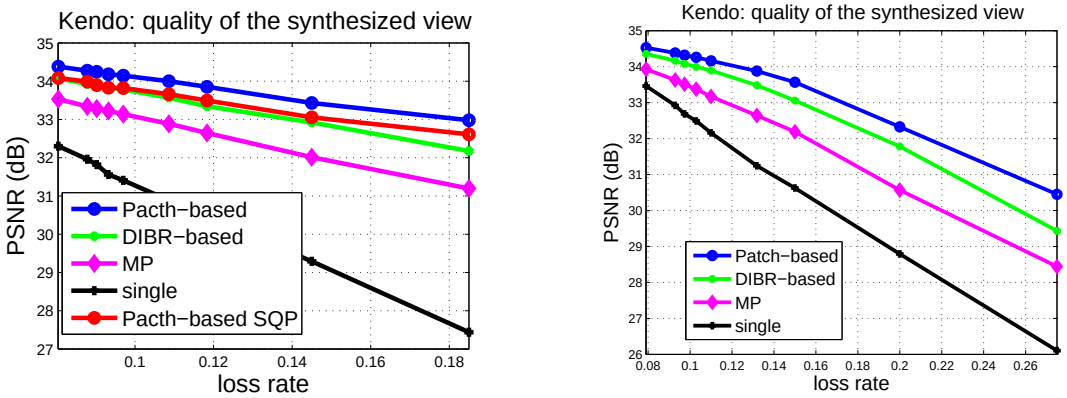
FIGURE 5.15: `Kendo`: Streaming results with different channel loss rates.

is more efficient in terms of source coding. The reason for this outcome is that if the communication channel enters a bad state, FEC cannot sufficiently protect lost data, and a lost frame can lead to a long error propagation. For multi-path transmission, the probability of both paths entering a bad state is quite low. Compared with `DIBR-based`, our proposed scheme `Patch-based` has better performance because of our advanced frame recovery scheme and source / channel rate optimization. `EEP`'s performance is worse compared with `Patch-based` because the FEC packets in `EEP` are not optimally allocated. To save space, we omit `EEP` in the following figures.

Then we changed the parameters of the GE model to be the following: g=0.02, b=0.98, q=0.05 with p varied to induce different loss rates. The results are shown in Fig. 5.15(b). Similar performance trend can be observed. `Patch-based` outperformed `single` by up to 4.2dB and 4.8dB in Fig. 5.15(a) and (b), respectively.

We also tested the cases when the two paths have asymmetric path loss rates, and the results are shown in Fig. 5.16(a). For the multi-path transmission, the GE parameters assumed for one path were g=0.05, b=0.95, q=0.1 p=0.0071, and GE parameters for the other path were g=0.02, b=0.98, q=0.05 with p varied throughout the simulation. Then we computed the expected loss characteristics for the two paths (expected bad state duration and expected loss rate) and selected comparable single-path GE parameters, $g = 0.035$, $b = 0.965$, and $q = 0.1333$ for `single`, so that the single path also has the same expected bad state duration and expected loss rate. In Fig. 5.16 (a), `Patch-based` outperformed `Patch-based SQP` by up to 0.4dB, which shows the advantages of using different QPs for texture and depth video encoding.

Next, we tested the case when the two paths have different transmission bandwidth, and the results are shown in Fig. 5.16(b), where all the paths were simulated using GE parameters g=0.05, b=0.95, q=0.1 with p varied to induce different loss rates. The bandwidth values of the two paths in the multi path scenario were set to 400 kbps and 500 kbps, respectively, and the bandwidth of *single* was 900 kbps. For both Fig. 5.16(a) and (b), similar performance could be observed as in Fig. 5.15. Relative to the single path / single description scheme, the performance gain of our system reaches up to 5.5dB and 4.4dB in Fig. 5.16(a) and (b), respectively.

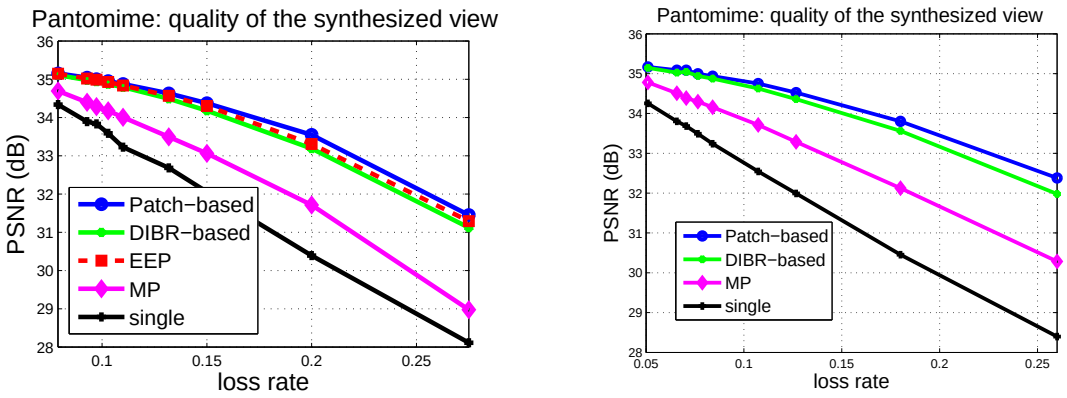


(a) Asymmetric loss rates (b) Asymmetric bandwidth

FIGURE 5.16: `Kendo`: Streaming results with asymmetric loss rates and bandwidth values.

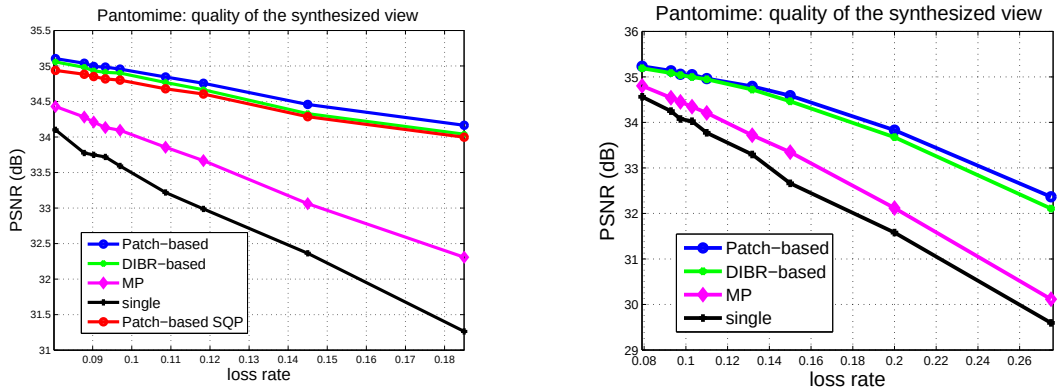
We also conducted the same experiments for `Pantomime`. In Fig. 5.17(a), the GE parameters assumed were g=0.05, b=0.95, q=0.1 with p varied to induce different loss rates. In Fig. 5.17 (b), the GE parameters assumed were g=0.02, b=0.98, q=0.05 with p varied. The bandwidth for each path in the multi-path scenario was 400 kbps and the bandwidth for `single` was 800 kbps. From the results, we can observe similar performance as for `Kendo`. The maximum performance gain relative to `single` is 3.4dB and 4.0dB in Fig. 5.17(a) and (b), respectively.

For `Pantomime`, we also tested the cases when the two paths have different loss conditions and different channel bandwidth values, as shown in Fig. 5.18. In Fig. 5.18(a), the GE parameters assumed for one of the paths were g=0.05, b=0.95, q=0.1 p=0.0071, and the GE parameters for the other path were g=0.02, b=0.98, q=0.05 with p varied to induce different loss rates. Then, the two paths' expected loss characteristics were used to construct a comparable single-path loss GE model for `single`, with parameters $g = 0.035$, $b = 0.965$, and $q = 0.1333$. We again varied p to control the overall loss



(a) Channel 1 (b) Channel 2

FIGURE 5.17: `Pantomime`: Streaming results with different channel loss rates.



(a) Asymmetric loss rates (b) Asymmetric bandwidth

FIGURE 5.18: `Pantomime`: Streaming results for asymmetric loss rates and bandwidth values.

rate in this case. From the simulation results, we could observe that our proposed scheme outperformed `single` by up to 2.9dB. In Fig. 5.18(a), `Patch-based` outperformed `Patch-based SQP` by up to 0.2dB.

When the two transmission paths have different bandwidth with the corresponding transmission channels simulated using the GE parameters g=0.05, b=0.95, q=0.1, and with p varied throughout the simulation, the results are shown in Fig. 5.18(b). The two paths were with 400 kbps and 500 kbps bandwidth respectively for the multi-path transmission schemes, and the single path transmission had 900 kbps bandwidth available. We observe that our proposed scheme can outperform `single` by up to 2.8dB.

5.7 Summary

Streaming of free viewpoint video in the texture-plus-depth format over wireless networks is a challenging problem due to the burstiness of the packet losses in wireless links and the stringent packet delivery deadlines of interactive video. In this paper, we propose to first encode the texture and depth signals of two camera-captured viewpoints into two independently decodeable descriptions for transmission over two disjoint wireless network paths. The source and channel coding rates for each description are optimized using an efficient branch-and-bound algorithm. In the event that a description is lost during transmission, missing frames in the lost description can be partially reconstructed using frames in the received description by exploiting the temporal and inter-view correlation of the transmitted viewpoints. Experimental results show that our proposed scheme can outperform a naïve single description / single path streaming solution by up to 5.5dB in PSNR.

Chapter 6

Discussion, Future Work and Conclusion

This chapter discusses the video streaming and the proposed solutions, talks about the issues that have not been addressed and concludes this dissertation.

6.1 Discussion

Video is playing a more and more important role nowadays. A significant fraction of commercial streaming traffic are HTTP-based, which is not desirable due to the retransmission. UDP is more suitable for time sensitive applications such as video streaming, but there is no guarantee of the data delivery. FEC is introduced to help improve the network video transmission but insufficient due to the wireless channels' burst-loss prone characters. UDP plus FEC seems a good choice for video streaming. What we are focusing in this dissertation is application-level streaming optimization, which unlike TCP, has knowledge about what's inside the packet payload so smarter decisions can be made regarding what to transmit, how to transmit and when. The lost frames could also be recovered to some extend.

Orthogonally, IMVS and FVV are introduced to provide users the freedom of viewing angle selection during playback, which is the core technology for a number of emerging applications such as education (such as on line lecture, training), sports events broadcasting (such as soccer, baseball, skating), medical treatment (surgery), travel guide

(multi-view version of place of interest), free viewpoint TV, immersive conferencing, multiview version of YouTube, etc, and is expected to be the next generation visual communication. How to transmit them over the lossy network, including what to transmit, how to transmit and how to recover the lost frame, is one fundamental problems to achieve these applications and hence a very important issue. Comparing with the single view video streaming, the additional challenges are how to utilize the multi-view information besides the view itself for better performance.

In this dissertation, we started from the IMVS, which provides users the ability to switch to other discrete captured view angles. And then we studied the FVV, which provides users more freedom in term of the view angle selection. Specifically, FVV could let users select any viewing angles (mostly not captured) he/she prefers instead of only the captured view. This leads to different strategies in term of selecting the source video, where the server in IMVS only needs to send the users the texture frames requested, but in FVV, the server needs to send the two nearest neighboring views in texture plus depth format. Moreover, when the user sends a different view request, IMVS only needs to send the newly demanded view's texture. While in FVV, the server has to transmit the two nearest neighboring views' texture and depth information to synthesize the requested middle view. Given IMVS and FVV are quite different in term of the delivered source video and view switch, we designed three solutions in this thesis. But please note that the solutions for IMVS and FVV are complementary for the multi-view video streaming, i.e. the techniques behind three solutions could be combined for better performances if possible. E.g. local repair could also be applied in FVV video streaming for recovery if the mobile devices are equipped with more than one interference. When both paths suffer burst loss at the same time, the user can still recover the lost frames via neighboring users' packet sharing if the neighboring users are watching related video content with the delay to be one GOP's playback time. Also, if one description on one path goes into the bad state in the *multi-path free viewpoint video streaming*, all the frames followed by can not be correctly decoded due to the lack of the predictor. In this case, if DE-DSC frames are inserted, the error propagation problem could be solved to some extend. Hence the performance could be improved.

The proposed solutions also have limitations. For example, in *multi-path free viewpoint video streaming*, the proposed frame recovery scheme employs complex TSR, etc for high recovery performance and hence is with high computational complexity. But, nowadays,

mobile devices' battery developments are lagging behind the mobile devices' function and processing development. The battery consumption becomes a critical problem for mobile users. The encoding schemes, transmission schemes and decoding schemes should take into the mobile devices' extra battery consumption (mainly coming from the frame recovery) into consideration. For this, we did some preliminary work on how to trade off the video quality and the energy consumption and our preliminary results [135] have been published. Moreover, in *cooperative peer recovery for multi-view video multicast*, multi interferences are needed and neighboring users within the WLAN range are supposed to watch the same or similar video, which are actually very strong assumptions. Hence the proposed scheme relying on the local repair can only work in limited scenarios.

Video will play a more important role in the future. The next generation visual communication will be user-centric and high quality (in term of the resolution, frame rate, etc). More and more interactivity will be introduced between server and users, such as the view switch ability. I assume multi-view production will become available in the commercial market around 2017. Moreover, multi-view will be used not only for entertainment, but also for other purposes, e.g. multiple surveillance cameras for enhanced securities comparing with only one camera in one angle.

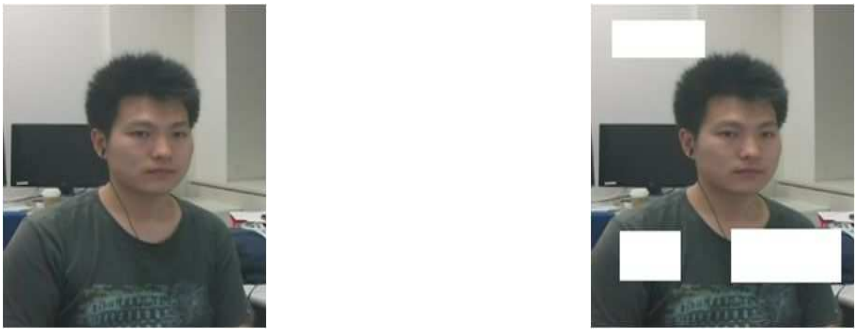
6.2 Future Work

Video is becoming more and more popular in our daily life and this trend will last. Regarding the multi-view video streaming, there are still many issues that have not been addressed.

6.2.1 Saliency-aware Background Recovery

Visually salient blocks of a corresponding visual scene are heavily protected when performing UEP. These salient blocks will be correctly received with high probability, while other blocks' lost probability is high during the transmission. The problem is these lost blocks will become salient and attract attention although they should not. For example, in a video teleconferencing system as shown in Fig. 6.1 (b), we can assume salient object is the person and the background is relatively static and non-salient. However, missing blocks in the background will easily attract attention and hence become salient. Most of

the researches focus on how to protect or recover the lost salient objects. How to recover the lost blocks that are not salient, such as the missing blocks in the background, while trying to keep visual saliency to a minimum is an important problem.



(a) correct view (b)view with losses in the background

FIGURE 6.1: Illustration of the video teleconferencing views.

6.2.2 Multi-view Video Representation

What is the best way of encoding the multi-view is an important issue? Today's internet is overwhelmed by the video traffic. Current MVC is not a good choice for the multi-view video streaming due to its prediction schemes as discussed. We can apply the traditional video coding scheme H.264 or HEVC to each of the view separately. Then upon a view request, only the view demanded needs to be delivered from server to users in IMVS and only two views' texture plus depth will be sent over the networks in FVV streaming, which is more network friendly comparing with the case that applies MVC as the encoding methods. But for both IMVS and FVV streaming, large redundances exist between/inside views. What is the best way of representing the multi-view or how to design a coding efficient and transmission friendly encoding scheme is still an open question.

6.2.3 Cross-layer Network Optimization

Although we jointly considered the video encoding methods, frame error concealments which are the works in the signal processing community at the application layer and streaming protocols, cooperative strategies which are the works in the communication community at the network transport layer. But still there are many other network issues at other layers that have not been addressed, such as how to allocate the radio resources

at *physical/MAC layers* [136, 137] to optimize the multi-view video streaming has not been studied in any formal way as far as we understand. How to optimize the networks for the multi-view video streaming co-considering other layer issues impacts the video transmission performance greatly and is non-trivial.

6.2.4 View Switch in x/y/z Direction

Providing users more interactivity and more freedom is one direction for future video applications, which could make the video system more fancy and attractive. IMVS and FVV in this dissertation only allow viewing angle movement along the x direction (horizontally), how about moving in the the orthogonal z direction or a combination of the x/y direction and z direction? If this freedom is provided, what will the encoding scheme, the transmission scheme and the error concealment scheme be are still open questions.

6.2.5 Multi-view over Next Generation Networks

Last but not the least, nowadays, the cloud computing is more and more popular. Besides, *content-centric networking* [138] *(also named data networking, content-based networking, data-oriented networking or information-centric networking)* was proposed as an alternative approach to the architecture of future Internet. The network may change dramatically in the near future. By when different kinds of networks may exist at the same time, and current transmission schemes may not work efficiently. How to design a video streaming scheme that is robust to different kinds of networks would be important for future researches.

6.3 Conclusion

In this dissertation, we assume the multi-view video have already been captured but have not been encoded, and there is no retransmission for lost packets over the wireless network channels, which are with limited bandwidth constraints and could be simulated using the state-of-the-art channel loss models. We focus on the problem of how to deliver the video in IMVS and FVV scenarios including the video encoding, video transmission

and lost frames' error recovery/concealment. We designed the coding methods, frame recovery schemes which are the work in the single processing community at the application layer; and proposed streaming protocols and cooperative strategies which are the work in the communication community at the network transport layer, the cross-layer, and cross-community solution makes it challenging, effective and impressive.

Specifically, we proposed a cooperative local recovery scheme to alleviate individual WWAN packet losses without asking server's retransmission, so that a loss-stricken peer can then either recover using received CPR packets of the same view, or using packets of two adjacent views and subsequent view interpolation via image-based rendering. We optimized the decision process for individual peers during CPR for recovery of multi-view video content using the distributed MDP. Experiments show that decisions made using our proposed MDP outperforms decisions made by a random scheme by at least 1.8dB in PSNR in received video quality in typical network scenario.

In addition the loss-resilient aspect during network streaming, we also designed a new uDSC frame for periodic insertion into the multi-view frame structure to facilitate view-switching and contain error propagation. After inserting uDSC-frames into the coding structure, we schedule packets for network transmission in a rate-distortion optimal manner for both wireless multicast and wired unicast streaming scenarios. Experimental results show that systems that insert uDSC frames and optimize packet transmission can outperform other competing coding schemes by up to 2.8 and 11.6 dB in wireless multicast and wired unicast streaming scenarios, respectively.

For FVV transmission, we proposed to encode the texture and depth signals of two camera captured viewpoints into two independently decodeable descriptions for transmission over two disjoint wireless network paths. We designed a novel missing frames recovery scheme using frames in the received description by exploiting the temporal and inter-view correlation of the transmitted viewpoints. Near-optimal source and channel coding rates for each description were selected using a branch-and-bound method, for the given transmission bandwidth on each path. Experimental results show that our system can outperform a traditional single-description / single-path transmission scheme by up to 5.2dB in PSNR of the synthesized intermediate view at the client.

Bibliography

[1] M. Mody, P. Swami, and P. Shastry. Ultra-low latency video codec for video conferencing. In *Electronics, Computing and Communication Technologies (IEEE CONECCT), 2014 IEEE International Conference on*, pages 1–5, Jan 2014.

[2] Meng Chen, Guan-Ming Su, and Min Wu. Dynamic resource allocation for robust distributed multi-point video conferencing. *Multimedia, IEEE Transactions on*, 10(5):910–925, Aug 2008.

[3] Bo Li and Hao Yin. Peer-to-peer live video streaming on the internet: issues, existing approaches, and challenges [peer-to-peer multimedia streaming]. *Communications Magazine, IEEE*, 45(6):94–99, June 2007.

[4] Xuening Liu, Hao Yin, Chuang Lin, Yu Liu, Zhijia Chen, and Xin Xiao. Performance analysis and industrial practice of peer-assisted content distribution network for large-scale live video streaming. In *Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference on*, pages 568–574, March 2008.

[5] D. Deloddere, W. Verbiest, and H. Verhille. Interactive video on demand. *Communications Magazine, IEEE*, 32(5):82–88, May 1994.

[6] K.K. Nayfeh and N.J. Sarhan. Design and analysis of scalable and interactive near video-on-demand systems. In *Multimedia and Expo (ICME), 2013 IEEE International Conference on*, pages 1–6, July 2013.

[7] Kunwadee Sripanidkulchai, Bruce Maggs, and Hui Zhang. An analysis of live streaming workloads on the internet. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, IMC '04, pages 41–54, New York, NY, USA, 2004.

[8] Kalle Honkanen et al. Http live streaming. 2011.

[9] Alex Zambelli. Iis smooth streaming technical overview. *Microsoft Corporation*, 3, 2009.

[10] Vasanthi Dwaraka Bhamidipati and Swetha Kilari. Effect of delay/delay variable on qoe in video streaming. *M. Thesis, School of Computing at Blekinge Institute of Technology, Sweden*, 2010.

[11] Yuedong Xu, Eitan Altman, Rachid El-Azouzi, Salah Eddine Elayoubi, and Majed Haddad. Qoe analysis of media streaming in wireless data networks. In *NETWORKING 2012*, pages 343–354. Springer, 2012.

[12] Bur Goode. Voice over internet protocol (voip). *Proceedings of the IEEE*, 90(9): 1495–1517, 2002.

[13] Catherine Boutremans and J-Y Le Boudec. Adaptive joint playout buffer and fec adjustment for internet telephony. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 1, pages 652–662. IEEE, 2003.

[14] Haitao Zheng and Jill Boyce. An improved udp protocol for video transmission over internet-to-wireless networks. *Multimedia, IEEE Transactions on*, 3(3):356–365, 2001.

[15] Shih-Ying Chang, Hsin-Ta Chiao, Xin-Yan Yeh, and Ming-Chien Tseng. Udp-based file delivery mechanism for video streaming to high-speed trains. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on*, pages 3568–3572. IEEE, 2013.

[16] Luca De Cicco, Saverio Mascolo, and Vittorio Palmisano. Skype video congestion control: An experimental investigation. *Computer Networks*, 55(3):558–571, 2011.

[17] J. Widmer, R. Denda, and M. Mauve. A survey on tcp-friendly congestion control. *Network, IEEE*, 15(3):28–37, May 2001.

[18] Michael Luby, Lorenzo Vicisano, Jim Gemmell, Luigi Rizzo, M Handley, and Jon Crowcroft. The use of forward error correction (fec) in reliable multicast. Technical report, RFC 3453, December, 2002.

[19] Hang Liu, Hairuo Ma, Magda El Zarki, and Sanjay Gupta. Error control schemes for networks: An overview. *Mobile Networks and Applications*, 2(2):167–182, 1997.

[20] Rohit Puri, Kannan Ramchandran, Kang-Won Lee, and Vaduvur Bharghavan. Forward error correction (fec) codes based multiple description coding for internet video streaming and multicast. *Signal Processing: Image Communication*, 16(8): 745–762, 2001.

[21] Jianfei Cai and Chang Wen Chen. Fec-based video streaming over packet loss networks with pre-interleaving. In *Information Technology: Coding and Computing, 2001. Proceedings. International Conference on*, pages 10–14. IEEE, 2001.

[22] Richard W Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950.

[23] I.-H. Hou and P. R. Kumar. Scheduling heterogeneous real-time traffic over fading wireless channels. In *IEEE INFOCOM*, San Diego, CA, March 2010.

[24] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. In *IEEE TCSVT*, volume 13, no.7, pages 560–576, July 2003.

[25] Gary J. Sullivan, Jens-Rainer Ohm, Woojin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Trans. Circuits Syst. Video Techn.*, 22(12):1649–1668, 2012.

[26] G. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. In *IEEE SP Magazine*, November 1998.

[27] R. Krishnamurthy, B.-B. Chai, H. Tao, and S. Sethuraman. Compression and transmission of depth maps for image-based rendering. In *IEEE International Conference on Image Processing*, Thessaloniki, Greece, October 2001.

[28] Stephan Wenger, G.D. Knorr, J. Ott, and F. Kossentini. Error resilience support in h.263+. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8 (7):867–877, Nov 1998. ISSN 1051-8215. doi: 10.1109/76.735382.

[29] A. Sehgal, A. Jagmohan, and N. Ahuja. Wyner-Ziv coding of video: an error-resilient compression framework. In *IEEE Transactions on Multimedia*, volume 6, no.2, pages 249–258, April 2004.

[30] P. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. Technical Report MSR-TR-2001-35, Microsoft Research, February 2001.

[31] P. Frossard and O. Verscheure. Joint source/FEC rate selection for quality-optimal MPEG-2 video delivery. In *IEEE Trans. Image Processing*, volume 10, no.12, pages 1815–1825, December 2001.

[32] H. Song and C.-C. J. Kuo. Rate control for low-bit-rate video via variable-encoding frame rates. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 11, no.4, pages 1051–8215, April 2001.

[33] G. Cheung, W. t. Tan, and T. Yoshimura. Rate-distortion optimized application-level retransmission using streaming agent for video streaming over 3G wireless network. In *IEEE International Conference on Image Processing*, Rochester, NY, September 2002.

[34] J. Chakareski and P. Chou. Application layer error correction coding for rate-distortion optimized streaming to wireless clients. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 2513–2516, Orlando, FL, May 2002.

[35] J. Chakareski, P. Chou, and B. Aazhang. Computing rate-distortion optimized policies for streaming media to wireless clients. In *IEEE Data Compression Conference*, pages 53–62, Snowbird, UT, April 2002.

[36] D. Li, G. Cheung, C. N. Chuah, and S. J. Yoo. Joint server/peer receiver-driven rate-distortion optimized video streaming using asynchronous clocks. In *IEEE International Conference on Image Processing*, Singapore, October 2004.

[37] P. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. In *IEEE Transactions on Multimedia*, volume 8, no.2, pages 390–404, April 2006.

[38] Y. Wang and Q.-F. Zhu. Error control and concealment for video communication: A review. *IEEE Proceedings*, 86:974–997, 1998.

[39] C.-M. Huang, K.-C. Yang, and J.-S. Wang. Error resilience supporting bi-directional frame recovery for video streaming. In *IEEE International Conference on Image Processing*, Singapore, October 2004.

[40] X. Xiu, G. Cheung, A. Ortega, and J. Liang. Optimal frame structure for interactive multiview video streaming with view synthesis capability. In *IEEE International Workshop on Hot Topics in 3D (in conjunction with ICME 2011)*, Barcelona, Spain, July 2011.

[41] G. Petrazzuoli, M. Cagnazzo, F. Dufaux, and B. Pesquet-Popescu. Using distributed source coding and depth image based rendering to improve interactive multiview video access. In *IEEE International Conference on Image Processing*, Brussels, Belgium, September 2011.

[42] Y. Morvan, P.H.N. de With, and D. Farin. Platelets-based coding of depth maps for the transmission of multiview images. In *SPIE Stereoscopic Displays and Applications*, San Jose, CA, January 2006.

[43] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp. Client-driven selective streaming of multiview video for interactive 3DTV. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 17, no.11, pages 1558–1565, November 2007.

[44] G. Cheung, A. Ortega, and T. Sakamoto. Coding structure optimization for interactive multiview streaming in virtual world observation. In *IEEE International Workshop on Multimedia Signal Processing*, Cairns, Queensland, Australia, October 2008.

[45] G. Cheung, N.-M. Cheung, and A. Ortega. Optimized frame structure using distributed source coding for interactive multiview streaming. In *IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009.

[46] Z. Liu, G. Cheung, and Y. Ji. Unified distributed source coding frames for interactive multiview video streaming. In *in IEEE International Conference on Communications*, Ottawa, Canada, June 2012.

[47] T. Fujii and M. Tanimoto. Free viewpoint TV system based on ray-space representation. In *Proceedings of SPIE*, volume 4864, page 175, 2002.

[48] M. Maitre, Y. Shinagawa, and M.N. Do. Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering. In *IEEE Transactions on Image Processing*, volume 17, no.6, pages 946–957, June 2008.

[49] X. Xiu, G. Cheung, and J. Liang. Delay-cognizant interactive multiview video with free viewpoint synthesis. In *IEEE Transactions on Multimedia*, volume 14, no.4, pages 1109–1126, August 2012.

[50] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo. Free-viewpoint TV. In *IEEE Signal Processing Magazine*, volume 28, no.1, January 2011.

[51] Z. Liu, G. Cheung, J. Chakareski, and Y. Ji. Multiple description coding of free viewpoint video for multi-path network streaming. In *2012 IEEE Global Communication Conference*, Anaheim, USA, Decemeber 2012.

[52] P. Merkle, A. Smolic, K. Muller, and T. Wiegand. Efficient prediction structures for multiview video coding. In *IEEE TCSVT*, volume 17, no.11, pages 1461–1473, November 2007.

[53] M. Flierl, A. Mavlankar, and B. Girod. Motion and disparity compensated coding for multiview video. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 17, no.11, pages 1474–1484, November 2007.

[54] S. Shimizu, M. Kitahara, H. Kimata, K. Kamikura, and Y. Yashima. View scalable multiview coding using 3-D warping with depth map. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 17, no.11, pages 1485–1495, November 2007.

[55] G. Cheung, A. Ortega, and N.-M. Cheung. Interactive streaming of stored multiview video using redundant frame structures. In *IEEE Transactions on Image Processing*, volume 20, no.3, pages 744–761, March 2011.

[56] J.-G. Lou, H. Cai, and J. Li. A real-time interactive multi-view video system. In *ACM International Conference on Multimedia*, Singapore, November 2005.

[57] P. Merkle, A. Smolic, K. Mueller, and T. Wiegand. Multi-view video plus depth representation and coding. In *IEEE International Conference on Image Processing*, San Antonio, TX, October 2007.

[58] G. Shen, W.-S. Kim, S.K. Narang, A. Ortega, J. Lee, and H. Wey. Edge-adaptive transforms for efficient depth map coding. In *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[59] J. Gautier, O. Le Meur, and C. Guillemot. Depth map coding: exploiting the intrinsic properties of scenes and surface layout. In *Picture Coding Symposium 2012*, Krakow, Poland, May 2012.

[60] W. Hu, G. Cheung, X. Li, and O. Au. Depth map compression using multi-resolution graph-based transform for depth-image-based rendering. In *Proc. of the IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

[61] Yi Liang, Markus Flierl, and Bernd Girod. Low-latency video transmission over lossy packet networks using rate-distortion optimized reference picture selection. In *in Proc. of the IEEE International Conference on Image Processing*, Rochester, New York, USA, September 2002.

[62] G. Cheung, W.-T. Tan, and C. Chan. Reference frame optimization for multiple-path video streaming with complexity scaling. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 17, no.6, pages 649–662, June 2007.

[63] Abhik Majumdar, Daniel G. Sachs, Igor V. Kozintsev, Kannan Ramchandran, and Minerva M. Yeung. Multicast and unicast real-time video streaming over wireless LANs. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6): 524–534, 2002.

[64] Ming-Fong Tsai, Tzu-Chi Huang, Chih-Heng Ke, Ce-Kuen Shieh, and Wen-Shyang Hwang. Adaptive hybrid error correction model for video streaming over wireless networks. *Multimedia Syst.*, 17(4):327–340, 2011.

[65] X. Liu, G. Cheung, and C.-N. Chuah. Structured network coding and cooperative wireless ad-hoc peer-to-peer repair for WWAN video broadcast. In *IEEE Transactions on Multimedia*, volume 11, no.4, 2009.

[66] Nikolaos Thomos, Jacob Chakareski, and Pascal Frossard. Randomized network coding for UEP video delivery in overlay networks. In *in Proc. of the IEEE International Conference on Multimedia and Expo*, New York City, NY, USA, June 2009.

[67] Sajid Nazir, Dejan Vukobratovic, and Vladimir Stankovic. Expanding window random linear codes for data partitioned H.264 video transmission over DVB-H

network. In *in Proc. of the IEEE International Conference on Image Processing*, Brussels,Belgium, September 2011.

[68] B. Macchiavello, C. Dorea, M. Hung, G. Cheung, and W. t. Tan. Reference frame selection for loss-resilient depth map coding in multiview video conferencing. In *IS&T/SPIE Visual Information Processing and Communication Conference*, Burlingame, CA, January 2012.

[69] B. Macchiavello, C. Dorea, M. Hung, G. Cheung, and W. t. Tan. Reference frame selection for loss-resilient texture & depth map coding in multiview video conferencing. In *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

[70] B. Macchiavello, C. Dorea, M. Hung, G. Cheung, and W. t. Tan. Loss-resilient texture & depth map coding in multiview video conferencing. In *(accepted to) IEEE Transactions on Multimedia*, November 2013.

[71] A.A. Abouzeid, S. Roy, and M. Azizoglu. Stochastic modeling of tcp over lossy links. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, volume 3, pages 1724–1733 vol.3, 2000.

[72] M. Yajnik, Sue Moon, J. Kurose, and D. Towsley. Measurement and modelling of the temporal dependence in packet loss. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, volume 1, pages 345–352 vol.1, 1999.

[73] J. Zfzal, T. Stockhammer, T. Gasiba, and W. Xu. Video streaming over MBMS: A system design approach. In *Journal of Multimedia*, volume 1, no.5, pages 25–35, August 2006.

[74] P. Sharma, S.-J. Lee, J. Brassil, and K. Shin. Distributed communication paradigm for wireless community networks. In *IEEE International Conference on Communications*, Seoul, Korea, May 2005.

[75] Randeep Bhatia, Li (Erran) Li, Haiyun Luo, and Ram Ramjee. ICAM: Integrated cellular and ad hoc multicast. In *IEEE Transactions on Mobile Computing*, volume 5, no. 8, pages 1004–1015, August 2006.

[76] G. Cheung, P. Sharma, and S.J. Lee. Smart media striping over multiple burst-loss channels. In *IEEE Transactions on Selected Topics in Signal Processing*, volume 1, no.2, pages 319–333, August 2007.

[77] D. Nguyen and T. Nguyen. Network coding-based wireless media transmission using POMDP. In *Seventeenth International Packet Video Workshop*, Seattle, WA, May 2009.

[78] H. Park and M. van der Schaar. A framework for foresighted resource reciprocation in P2P networks. In *IEEE Transactions on Multimedia*, volume 11, no.1, pages 101–116, January 2009.

[79] D. Slepian and J. K. Wolf. Noiseless coding of correlated information sources. *IEEE Transaction on Information Theory*, 19:471–480, 1973.

[80] A. D. Wyner and Jacob Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Trans. Inform. Theory*, 22:1–10, 1976.

[81] S. Sandeep Pradhan, Julius Kusuma, and Kannan Ramchandran. Distributed compression in dense sensor networks. *IEEE Signal Processing Magazine*, 19:51–60, 2002.

[82] Zixiang Xiong, A. D. Liveris, and S. Cheng. Distributed source coding for sensor networks. *Signal Processing Magazine, IEEE*, 21(5):80–94, Aug 2004.

[83] A. Aaron, P. Ramanathan, and Bernd Girod. Wyner-Ziv coding of light fields for random access. In *IEEE International Workshop on Multimedia Signal Processing*, Siena, Italy, September 2004.

[84] N.-M. Cheung, H. Wang, and A. Ortega. Video compression with flexible playback order based on distributed source coding. In *IS&T/SPIE Visual Communications and Image Processing (VCIP'06)*, San Jose, CA, January 2006.

[85] N.-M. Cheung, A. Ortega, and G. Cheung. Distributed source coding techniques for interactive multiview video streaming. In *27th Picture Coding Symposium*, Chicago, IL, May 2009.

[86] G. Cheung, A. Ortega, N.-M. Cheung, and B. Girod. On media data structures for interactive streaming in immersive applications. In *SPIE Visual Communications and Image Processing*, Huang Shan, China, July 2010.

[87] J.G. Apostolopoulos. Error-resilient video compression via multiple state streams. *Proc. International Workshop on Very Low Bitrate Video Coding (VLBV'99)*, pages 168–171, October 1999.

[88] Leana Golubchik, John C. S. Lui, Tak Fu Tung, Alix L. H. Chow, Adam Woei-Jyh Lee, Giuliana Franceschinis, and Cosimo Anglano. Multi-path continuous media streaming: what are the benefits? *Perform. Eval.*, 49(1/4):429–449, 2002.

[89] J. Chakareski, S. Han, and B. Girod. Layered coding vs. multiple descriptions for video streaming over multiple paths. In *In Proc. of ACM Multimedia*, pages 422–431, 2003.

[90] Jacob Chakareski, Sangeun Han, and Bernd Girod. Layered coding vs. multiple descriptions for video streaming over multiple paths. In *Multimedia Syst.*, volume 10, pages 275–285, 2005.

[91] Varun Singh, Saba Ahsan, and Joerg Ott. Mprtp: Multipath considerations for real-time media. In *Proc. of ACM Multimedia Systems*, 2013.

[92] J. Zhai, K. Yu, J. Li, and S. Li. A low complexity motion compensated frame interpolation method. In *IEEE International Symposium on Circuits and Systems*, Kobe, Japan, May 2005.

[93] Pengye Xia, S.-H.G. Chan, and Xing Jin. Optimal bandwidth assignment for multiple-description-coded video. *Multimedia, IEEE Transactions on*, 13(2):366–375, April 2011.

[94] S. Wenger. Video redundancy coding in h.263+. In *Proceedings of AVSPN 97*, Aberdeen, U. K., 1997.

[95] Ce Zhu and Minglei Liu. Multiple description video coding based on hierarchical b pictures. *IEEE Trans. Circuits Syst. Video Techn.*, 19(4):511–521, 2009.

[96] J. Kim, Y. g. Kim, H. Song, T. y. Kuo, Y. Chung, and J. Kuo. TCP-friendly internet video streaming employing variable frame-rate encoding and interpolation. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 10, pages 1164–1177, 2000.

[97] C. Wang, L. Zhang, Y. He, and Y.-P. Tan. Frame rate up-conversion using trilateral filtering. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 20, no.6, pages 886–893, June 2010.

[98] Byeong-Doo Choi, Jong-Woo Han, Chang-Su Kim, and Sung-Jea Ko. Motion-compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation. *IEEE Trans. Cir. and Sys. for Video Technol.*, 17(4):407–416, April 2007.

[99] Jose Diogo Areia, Catarina Brites, O Pereira, and Jodo Ascenso. Wyner–ziv stereo video coding using a side information fusion approach. In *in: IEEE International Workshop on Multimedia Signal Processing, Chania*, pages 453–456, 2007.

[100] Mourad Ouaret, Frederic Dufaux, and Touradj Ebrahimi. Multiview distributed video coding with encoder driven fusion. In *in Proceedings of the European Conference on Signal Processing*, 2007.

[101] Giovanni Petrazzuoli, Marco Cagnazzo, and Beatrice Pesquet-Popescu. Novel solutions for side information generation and fusion in multiview dvc. *EURASIP Journal on Advances in Signal Processing*, 2013(1):154, 2013.

[102] H.-Y. Shum, S.-C. Chan, and S. B. Kang. *Image-Based Rendering*. Springer, 2007.

[103] *Technical Specification Group Services and System Aspects; Multimedia Broadcast/Multicast Service (MBMS) user services; stage 1 (Release 6) (3GPP TS.26.246 version 6.3.0*, March 2006.

[104] Tracey Ho, Muriel Medard, Ralf Koetter, David R. Karger, Michelle Effros, Jun Shi, and Ben Leong. A random linear network coding approach to multicast. In *IEEE TIT*, volume 52, pages 4413–4430, Oct. 2006.

[105] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang. Multiview imaging and 3DTV. In *IEEE Signal Processing Magazine*, volume 24, no.6, November 2007.

[106] C. Zhang, Z. Yin, and D. Florencio. Improving depth perception with motion parallax and its application in teleconferencing. In *IEEE MMSP*, Rio de Jeneiro, Brazil, October 2009.

[107] T. Fujii, K. Mori, K. Takeda, K. Mase, M. Tanimoto, and Y. Suenaga. Multipoint measuring system for video and sound—100 camera and microphone system. In *IEEE International Conference on Multimedia and Expo*, Toronto, Canada, July 2006.

[108] *Video Coding for Low Bitrate Communication.* ITU-T Recommendation H.263, February 1998.

[109] J. Crowcroft and K. Paliwoda. A multicast transport protocol. In *ACM SIGCOMM*, New York, NY, August 1988.

[110] Stephen B. Wicker and Vijay K. Bhargava, editors. *Reed-Solomon Codes and Their Applications.* John Wiley & Sons, 1999.

[111] L. Li, R. Ramjee, M. Buddhikot, and S. Miller. Network coding-based broadcast in mobile ad hoc networks. In *IEEE INFOCOM 2007*, Anchorage, AL, May 2007.

[112] G. Cheung and W. t. Tan. Redundant representation for network video streaming using reconstructed P-frames and SP-frames. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Dallas, TX, March 2010.

[113] J. Proakis. *Digital Communications.* McGraw Hill, third edition, 1995.

[114] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, , and R. Szeliski. High-quality video view interpolation using a layered representatio. *ACM Transactions on Graphic.*, 23(3):600–608, 2004.

[115] S. Reichelt, R. Haussler, G. Futterer, and N. Leister. Depth cues in human visual perception and their realization in 3D displays. In *SPIE Three-dimensional Imaging, Visualization, and Display*, Orlando, FL, April 2010.

[116] S. Gokturk, H. Yalcin, and C. Bamji. A time-of-flight depth sensor—system description, issues and solutions. In *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, Washington, DC, June 2004.

[117] M. Tanimoto, T. Fujii, and K. Suzuki. Multi-view depth map of Rena and Akko & Kayo. ISO/IEC JTC1/SC29/WG11 MPEG Document M14888, Oct. 2007.

[118] D. Tian, P.-L. Lai, P. Lopez, and C. Gomila. View synthesis techniques for 3D video. In *Applications of Digial Image Processing XXXII, Proceedings of the SPIE*, volume 7443, (2009), pages 74430T–74430T–11, February 2009.

[119] M. Podolsky, S. McCanne, and M. Vetterli. Soft arq for layered streaming media. Technical Report UCB/CSD-98-1024, University of California, Berkeley, November 1998.

[120] J.H. Sørensen, J. Østergaard, P. Popovski, and J. Chakareski. Multiple description coding with feedback based network compression. In *Proc. Globecom*, Miami, FL, USA, December 2010. IEEE.

[121] Y. Ding, Y. Yang, and L. Xiao. Multi-path routing and rate allocation for multi-source video on-demand streaming in wireless mesh networks. In *IEEE INFOCOM*, Shanghai, China, April 2011.

[122] E. N. Gilbert. Capacity of burst-noise channel. *Bell System Technical Journal*, 39:1253–1265, 1963.

[123] J. Chakareski. Transmission policy selection for multi-view content delivery over bandwidth constrained channels. *Image Processing, IEEE Transactions on*, 23(2): 931–942, Feb 2014.

[124] Y.-S. Kang, C. Lee, and Y.-S. Ho. An efficient rectification algorithm for multi-view images in parallel camera array. In *3DTV-Conference*, IStanbul, Turkey, May 2008.

[125] I. Daribo and B. Pesquet-Popescu. Depth-aided image inpainting for novel view synthesis. In *IEEE Multimedia Signal Processing Workshop*, Saint-Malo, France, October 2010.

[126] I. Ahn and C. Kim. Depth-based disocclusion filling for virtual view synthesis. In *IEEE International Conference on Multimedia and Expo*, Melbourne, Australia, July 2012.

[127] S. Reel, G. Cheung, P. Wong, and L. Dooley. Joint texture-depth pixel inpainting of disocclusion holes in virtual view synthesis. In *APSIPA ASC*, Kaohsiung, Taiwan, October 2013.

[128] G. Cheung, V. Velisavljevic, and A. Ortega. On dependent bit allocation for multiview image coding with depth-image-based rendering. In *IEEE Transactions on Image Processing*, volume 20, no.11, pages 3179–3194, November 2011.

[129] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *IEEE CVPR*, San Francisco, CA, June 2010.

[130] I. Daribo, D. Florencio, and G. Cheung. Arbitrarily shaped sub-block motion prediction in texture map compression using depth information. In *Picture Coding Symposium 2012*, Krakow, Poland, May 2012.

[131] Jian Sun, ZongBen Xu, and Heung-Yeung Shum. Image super-resolution using gradient profile prior. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*. IEEE Computer Society, 2008.

[132] Dongbo Min, Jiangbo Lu, and M.N. Do. Depth video enhancement based on weighted mode filtering. *Image Processing, IEEE Transactions on*, 21(3):1176–1190, 2012.

[133] C.Tomasi and R.Marnduchi. Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision*, Washington, DC, USA, 1998.

[134] Allen Gersho and Robert M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[135] Zhi Liu, Jingyun Feng, Yusheng Ji, and Yongbing Zhang. Adaptive energy-aware free viewpoint video transmission over wireless networks. In *Computing, Networking and Communications (ICNC), 2014 International Conference on*, pages 157–162, Feb 2014.

[136] Guan-Ming Su, Zhu Man, Min Wu, and KJ Ray Liu. Multiuser cross-layer resource allocation for video transmission over wireless networks. *Network, IEEE*, 20(2): 21–27, 2006.

[137] Hao Zhou, Yusheng Ji, and Baohua Zhao. Tabu search-based metaheuristic resource allocation algorithm for svc multicast over wireless relay networks. *IEEE TVT*, 2013.

[138] Kalika Suksomboon, Saran Tarnoi, Yusheng Ji, Michihiro Koibuchi, Kensuke Fukuda, Shunji Abe, Nakamura Motonori, Michihiro Aoki, Shigeo Urushidani,

and Shigeki Yamada. Popcache: Cache more or less based on content popularity for information-centric networking. In *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, pages 236–243. IEEE, 2013.

Publication List

Journal Publications

1. Zhi Liu, Jingyun Feng, Yusheng Ji and Yongbing Zhang, 'EAF: Energy-aware Adaptive Free Viewpoint Video Wireless Transmission,' accepted by Journal of Network and Computer Applications, July 2014.

2. Zhi Liu, Gene Cheung, Jacob Chakareski, Yusheng Ji, 'Multiple Description Coding & Recovery of Free Viewpoint Video for Wireless Network Streaming,' accepted to special issue on 'Visual Signal Processing for Wireless Networks,' IEEE Journal of Selected Topics in Signal Processing, February 2014.

3. Zhi Liu, Gene Cheung, and Yusheng Ji, 'Optimizing Distributed Source Coding for Interactive Multiview Video Streaming over Lossy Networks,' IEEE Transactions on Circuits and Systems for Video Technology, vol.23, no.10, pp.1781-1794, October 2013.

Conference Publications

1. Ruijian An, Zhi Liu, Yusheng Ji, 'Video Streaming for Highway VANET Using Scalable Video Coding,' accepted by IEEE VTC Fall, Canada, September 2014.

2. Jingyun Feng, Zhi Liu, Yusheng Ji, 'Wireless Channel Loss Analysis- A Case Study Using WiFi-Direct,' accepted by 10th International Wireless Communications Mobile Computing Conference (IWCMC) Cyprus, August 2014.

3. Zhi Liu, Yusheng Ji, 'Intercell Interference Coordination under Data Rate Requirement Constraint in LTE-Advanced Heterogeneous Networks,' in the proceedings

of the IEEE VTC Spring, Korea, May 2014 (VTC2014-Spring Young Researcher's Encouragement Award).

4. Zhi Liu, Jingyun Feng, Yusheng Ji and Yongbing Zhang, 'Adaptive Energy-aware Free Viewpoint Video Transmission over Wireless Networks,' in the proceedings of the 2014 International Conference on Computing, Networking and Communications, Hawaii, USA, February 2014

5. Zhi Liu, Yu Mao, Ning Lu, Yusheng Ji and Sherman Shen, 'Resource Allocation for WWAN Video Multicast with Cooperative Local Repair,' in the proceedings of the IEEE ICC13, Budapest, June 2013.

6. Zhi Liu, Gene Cheung, Jacob Chakareski and Yusheng Ji, 'Multiple Description Coding of Free Viewpoint Video for Multi-Path Network Streaming,' in the proceedings of the IEEE GLOBECOM 2012, Anaheim, California, USA, December 2012.

7. Zhi Liu, Gene Cheung and Yusheng Ji, 'Unified Distributed Source Coding Frames for Interactive Multiview Video Unicast,' accepted by the IEEE INFOCOM 2012 Student Workshop, Orlando, Florida, March 2012.

8. Zhi Liu, Gene Cheung and Yusheng Ji, 'Unified Distributed Source Coding for Loss Recovery in Interactive Multiview Video Streaming,' in the proceedings of the IEEE International Conference on Communications(IEEE ICC),Ottawa, Canada, June 2012.

9. Zhi Liu, Gene Cheung, Yusheng Ji, 'Distributed Markov Decision Process in Cooperative Peer Recovery for WWAN Multiview Video Multicast,' in the proceedings of the IEEE Visual Communication and Image Processing (VCIP) Conference, Tainan City, Taiwan, November 2011.

10. Zhi Liu, Gene Cheung, Yusheng Ji, 'Distributed Markov Decision Process in Cooperative Peer-to-peer Repair for WWAN Video Broadcast,' in the proceedings of the IEEE Workshop on Streaming and Media Communications (in conjunction with ICME 2011), Barcelona, Spain, July 2011(Best Student Paper award).

11. Junfeng Jin, Yusheng Ji,Baohua Zhao, Zhou Hao, Zhi Liu, 'Error-resilient Video Multicast with Layered Hybrid FEC/ARQ over Broadband Wireless Networks,' in

the proceedings of the IEEE Global Communication Conference(IEEE GlobeCom), Houston, USA, December 2011.

12. Zhi Liu, Gene Cheung, Yusheng Ji, 'Distributed Source Coding for WWAN Multiview Video Multicast with Cooperative Peer-to-peer Repair,' in the proceedings of the IEEE International Conference on Communications(IEEE ICC), Kyoto, Japan, June 2011.

13. Zhi Liu, Gene Cheung, Vladan Velisavljevic, Erhan Ekmekcioglu, Yusheng Ji, 'Joint Source / Channel Coding for WWAN Multiview Video Multicast with Cooperative Peer-to-peer Repair,' in Proceedings of special session on "Advanced Interactive Multimedia Streaming'in Packet Video Workshop, Hong Kong, December 2010.

14. Jing Yuan, Guang-Zhong Sun, Ye Tian, Guoliang Chen, Zhi Liu, 'Selective-NRA Algorithms for Top-k Queries,' in the proceedings of The Joint International Conferences on Asia-Pacific Web Conference (APWeb) and Web-Age Information Management (WAIM)(APWeb/WAIM), Suzhou, China, April 2009.