University of Alberta

Multiview Video Compression

by

Baochun Bai

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

©Baochun Bai
Fall 2009
Edmonton, Alberta

Permission is hereby granted to the University of Alberta Libraries to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only. Where the thesis is converted to, or otherwise made available in digital form, the University of Alberta will advise potential users of the thesis of these terms.

The author reserves all other publication and other rights in association with the copyright in the thesis and, except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatsoever without the author's prior written permission.

Examining Committee

Prof. Janelle Harms, Computing Science, University of Alberta

Prof. Pierre Boulanger, Computing Science, University of Alberta

Prof. Anup Basu, Computing Science, University of Alberta

Prof. Herbert Yang, Computing Science, University of Alberta

Prof. Ivan Fair, Electrical and Computer Engineering, University of Alberta

Prof. Robert Laganière, Information Technology and Engineering, University of Ottawa

Abstract

With the progress of computer graphics and computer vision technologies, 3D/multiview video applications such as 3D-TV and tele-immersive conference become more and more popular and are very likely to emerge as a prime application in the near future. A successful 3D/multiview video system needs synergistic integration of various technologies such as 3D/multiview video acquisition, compression, transmission and rendering. In this thesis, we focus on addressing the challenges for multiview video compression. In particular, we have made 5 major contributions: (1) We propose a novel neighbor-based multiview video compression system which helps remove the inter-view redundancies among multiple video streams and improve the performance. An optimal stream encoding order algorithm is designed to enable the encoder to automatically decide the stream encoding order and find the best reference streams. (2) A novel multiview video transcoder is designed and implemented. The proposed multiview video transcoder can be used to encode multiple compressed video streams and reduce the cost of multiview video acquisition system. (3) A learning-based multiview video compression scheme is invented. The novel multiview video compression algorithms are built on the recent advances on semi-supervised learning algorithms and achieve compression by finding a sparse representation of images. (4) Two novel distributed source coding algorithms, EETG and SNS-SWC, are put forward. Both EETG and SNS-SWC are capable to achieve the whole Slepian-Wolf rate region and are syndrome-based schemes. EETG simplifies the code construction algorithm for distributed source coding schemes using extended Tanner graph and is able to handle mismatched bits at the encoder. SNS-SWC has two independent decoders and thus can simplify the decoding process. (5) We propose a novel distributed multiview video coding scheme which allows flexible rate allocation between two distributed multiview video encoders. SNS-SWC is used as the underlying Slepian-Wolf coding scheme. It is the first work to realize simultaneous Slepian-Wolf coding of stereo videos with the help of a distributed source code that achieves the whole Slepian-Wolf rate region. The proposed scheme has a better rate-distortion performance than the separate H.264 coding scheme in the high-rate case.

Acknowledgements

I would like to first express my utmost gratitude to my two supervisors, Professor Janelle Harms and Professor Pierre Boulanger, for their guidance, support, and encouragement. They have created an excellent academic environment to make me freely pursuit my research interests and stimulate me for academic excellence. This thesis would have not been possible without their guidance.

I would also be very grateful to my candidacy and thesis examination committee members, Professor Mike MacGregor, Professor Herbert Yang, Professor Anup Basu, Professor Ivan Fair, and Professor Robert Laganière, for spending time in reading my candidacy document and thesis and asking me challenging questions and giving me invaluable comments. I also wish to thank Professor Csaba Szepesvári for chairing my thesis committee and for organizing various reading groups, from which I learned a lot on various theoretical subjects. I am thankful to Professor Zixiang Xiong for giving me the opportunity to visit his lab at Texas A & M University and for deepening my understanding on distributed source coding and distributed video coding through fruitful discussion.

I would like to specially thank Dr. Yuxi Li, Dr. Li Cheng, Dr. Cheng Lei, and Dr. Yang Yang for their friendship and collaboration during my course of graduate study. The intense interaction with them significantly enriches my life experience and deepens my understanding on various academic subjects.

I am lucky to be affiliated with two wonderful labs: computer networks lab and advanced man-machine interface lab. I would like to thank all the professors and students in the lab, Professor Ioanis Nikolaidis, Professor Pawel Gburzynski, Professor Ehab Elmallah, Professor Walter Bischof, Dr. Martha Steenstrup, Baljeet Malhotra, Nicholas Boers, Jinhui Shen, Qinghua Ye, Shoudong Zou, Israat Tanzeena Haque, Qiang Ye, Chen Liu, Robyn Taylor, Matthew Hamilton, Victor M. Ochoa Mayorga, Xing Dong Yang, Rui Shen, Rui Gao, Steven Eliuk, Fraser Anderson, Michelle Annett, Biao She, Xiaozhou Zhou, and many others, for creating the amiable and stimulating lab environment.

I would also like to thank the diligent staff members in the department for their top

notch services. I am especially grateful to Sunrose Ko, Linda Jiao, and Fran Moore for handling my travel claims and to Edith Drummond for helping me with all kind of errands related with the graduate study.

I am deeply indebted to my wonderful friends, Guohua Liu, Chonghai Wang, Linli Xu, Na Huang, Shengjiu Wang, Zhipeng Cai, Yi Shi, Yongxi Cheng, Guowei Wu, Guofan Hao, Zhuang Guo, Gongping Tang, Jianjun Zhou, Jiayuan Lu, Qin Wang, Xian Chen, Wenye Li, Zhentao Jia, Jiyang Chen, Xiaozhen Niu, and many others, for making my life in Edmonton full of joy and fun.

Finally, I would like to express sincere thanks to my family. I thank my parents for their love, care, and upbringing and wish that they can share my joy and happiness in the heaven. I am very grateful to my sisters and brothers for their intensive care, utmost support and ardent encouragement. No words are enough to express my love and appreciation for them.

Table of Contents

List of Tables

List of Figures

List of Acronyms

ADSC	Asymmetric distributed source coding
ASWC	Asymmetric Slepian-Wolf coding
B	Backward prediction mode
BBSC	Block-based stereoscopic coding
BD	Backward and disparity interpolation
BEC	Binary erasure channel
BER	Bit error rate
BPP	Bit per pixel
BPS	Bit per second
BSC	Binary symmetric channel
CPDMVT	Cascaded pixel domain multiview video transcoder
CPDT	Cascaded pixel-domain transcoder
D	Disparity prediction mode
DCP	Disparity-compensated prediction
DCT	Discrete cosine transform
DDT	DCT-domain transcoder
DF	Frame memory for disparity compensation
DMVC	Distributed multiview video coding
DSC	Distributed source coding
EETG	Enhanced extended Tanner graph
F	Forward prediction mode
FB	Forward and backward interpolation
FD	Forward and disparity interpolation
GGOP	Group of group of picture
GOP	Group of picture
IBR	Image-based rendering
IDCT	Inverse discrete cosine transform
III	H.264 sequence structure with only I frames
IPP	H.264 sequence structure with I and P frames
IQ	Inverse quantization
ITU-T	International telecommunication union - telecommunication standardization sector
JMVC	Joint multiview video coding
JMVM	Joint multiview video model
JSC	Joint source coding
JVT	Joint video team

KF	Key Frame
LapRLS	Laplacian regularized least square
LDPC	Low density parity check code
LLR	Log-likelihood ratio
LMVC	Learning-based multiview video coding
LSAS	Least square angle sum
LSS	Least square sum
MAD	Mean absolute difference
MAP	Maximum a posteriori probability
MC	Motion compensation
MCP	Motion-compensated prediction
MDC	Motion and disparity compensation
MDV	Motion and disparity vector
MF	Frame memory for motion compensation
MPEG	Moving picture experts group
MV	Motion vector
MVC	Multiview video coding
MVD	Multiview-video-plus-depth format
MVP	MPEG2 multiview profile
MVSN	Multiview video system over networks
MVT	Multiview video transcoder
PNS-DSC	Parity-based nonuniform distributed source coding
POVRAY	Persistence of vision raytracer
PSNR	Peak signal to noise ratio
Q	Quantization
QoS	Quality of service
QP	Quantization parameter
RGB	Red, green, blue
RKHS	Reproducing kernel hilbert space
RP	Representative pixel
SC	Coded representative pixel set
SDDT	Simplified DCT-domain transcoder
SDMVC	Symmetric distributed multiview video coding
SDSC	Symmetric distributed source coding
SN	Candidate representative pixel set
SNR	Signal to noise ratio
SNS-SWC	Syndrome-based nonuniform symmetric Slepian-Wolf coding
SPDMVT	Simplified pixel domain multiview video transcoder
SSC	Separate source coding
SSIF	Symmetric syndrome former - inverse syndrome former framework
SU	Uncoded representative pixel set
TCQ	Trellis coded quantization
TM	Two-machine algorithm
VCEG	Video coding experts group
WZF	Wyner-Ziv frame

Chapter 1

Introduction

With the advance of computer graphics and vision technologies, free viewpoint video or 3D video [1, 2, 3, 4, 5, 6, 7, 8, 9] will become a reality in the near future. Traditional videos such as those shown on TV or viewed on the Internet are a two-dimensional medium in nature. Namely, viewers can only passively observe the event captured by the cameraman and have no ability to actively change the viewpoint once the video is recorded. On the contrary, 3D video will allow the viewer to select an arbitrary viewpoint to a dynamic scene and thus enjoy a feeling of immersion into events such as an Olympic competition or a popular theater show. In other words, 3D video has the desirable feature of "interactivity", which is absent in the traditional 2D video. 3D video applications such as 3D TV and virtual theater [10, 11] will likely become very popular and emerge as a prime application.

A 3D video system is composed of various components: scene acquisition and representation, compression, transmission, rendering and display. To make 3D video applications become a reality in the future and foster a mass consumer market, significant technical challenges in all components of the processing chain need to be tackled. 3D video applications can be classified into two categories based on the 3D video distribution channel: (i) 3D video applications using a network, either a dedicated broadcast network or Internet, as a distribution channel. (ii) 3D video applications using a non-network channel, such as magnetic disks or CD-ROMs. In this thesis, we are mainly interested in 3D video applications that use networks as a distribution channel. For easy exposition, we call these system 3D video system over networks. In the following sections, we will give an overview of various components that composes a 3D video system over networks: video acquisition, video preprocessing, video compression, video transmission and rendering.

1.1 Image-based Rendering

Image-based rendering (IBR) [12, 13] is a common technique used to generate 3D videos. Unlike a conventional 3D rendering pipeline that generates photo-realistic images using geometric models, IBR creates images at an arbitrary viewpoint from a set of images captured from a real-world scene. The advantage of IBR is to use images rather than geometric models to represent the scene. Images are easier to obtain, simpler to handle and more realistic to render.

The key challenge of IBR is to how to perform a good scene description. Traditional computer graphics uses geometric models to represent the shapes of the objects, describes the properties of object surfaces by texture maps and reflection models, and creates images through the interaction between light sources and objects. Although the model-based scene representation works well to describe the synthetic world, it is not very good at accurately representing real-world dynamic scenes due to the lack of robust technology to reconstruct the geometric models of real-world scenes. An alternative scene representation technique is to use the pattern of light rays filling the space. These light rays can be described through a *plenoptic function* coined by Adelson and Bergen [14].

The original plenoptic function is defined as a seven-dimensional (7D) function that models a 3D dynamic environment by recording the intensity of light rays at every space location (V_x, V_y, V_z), towards every possible direction (θ, ϕ), for every wavelength λ and at any time t, i.e.,

$$P_7 = P(V_x, V_y, V_z, \theta, \phi, \lambda, t).$$

IBR can be characterized as a set of techniques to reconstruct a continuous representation of the plenoptic function from observed discrete samples. IBR is a two-stage process: sampling and rendering. In the sampling stage, samples of all possible light rays in the real-world dynamic scene are sensed and stored. Note that sufficient samples need to be taken so that the continuous plenoptic function can be reconstructed. In the rendering stage, the continuous plenoptic function is reconstructed from the captured samples.

Although the 7D plenoptic function is powerful and general to describe a real-world dynamic scene, it demands a tremendous amount of samples to reconstruct the continuous function, which is impossible to achieve in practice . Therefore, previous research have focused on how to make reasonable assumptions to reduce the sample size while achieving the acceptable rendering quality. The common assumptions are simplifying the wavelength

space into three channels such as red, green and blue channels to remove λ, imposing a static environment to remove t, assuming constant radiance of any light ray in the space to remove one dimension in viewpoints, restraining the viewing space to reduce other dimensions. For example, McMillan and Bishop [15] proposed the notion of plenoptic modeling with a five-dimensional (5D) plenoptic function, $P_5 = P(V_x, V_y, V_z, \theta, \phi)$, by removing wavelength λ and time t. Light field rendering [16] and Lumigraph [17] use a four-dimensional plenoptic function, $P_4 = P(u, v, s, t)$, by removing wavelength λ, time t, and assuming constant light radiance and imposing that viewers always stay outside the convex hull of an object. Concentric Mosaics [18] and panoramic video [19] adopt a three-dimensional (3D) plenoptic function by removing wavelength λ or time t, assuming the constant light radiance and constraining the viewer along a path or in a fixed location. The detailed taxonomy of IBR can be found in the recent surveys [13, 12, 20].

The ultimate goal of IBR is to create a 3D video that users can view at an arbitrary viewpoint. However, in practice, only a limited number of viewpoints can be generated from video streams captured by multiple cameras. Some researchers use "light field video" [21, 22] to describe the case. In this thesis, *multiview video* is used as the term to describe the video created by IBR based on multiple video streams captured by multiple cameras. Specifically, multiview video is defined as the reconstruction of five-dimensional plenoptic function by removing the wavelength λ and assuming constant light radiance. The system that distributes multiview video over networks is thus called multiview video system over networks (MVSN).

1.2 System Architecture of MVSN

MVSN aims at efficiently realizing the image-based rendering by appropriately dividing the system functions and assigning them to different subsystems. MVSN generally includes five subsystems, video acquisition, video preprocessing, video compression, transmission and rendering. Fig. 1.1 shows an architecture of a multiview video system over the Internet. Although it is a good design philosophy to keep subsystems independent from each other from a software engineering perspective, no subsystem can be totally independent in reality. The success of the system depends on the synergistic interaction of all subsystems. The goal of the system is to make perceived quality of experience adapt to the users' hardware capacity. All subsystems are indispensable and strive to make the rendering result as perfect as possible. We will briefly discuss the functions of each subsystem in the following

sections.

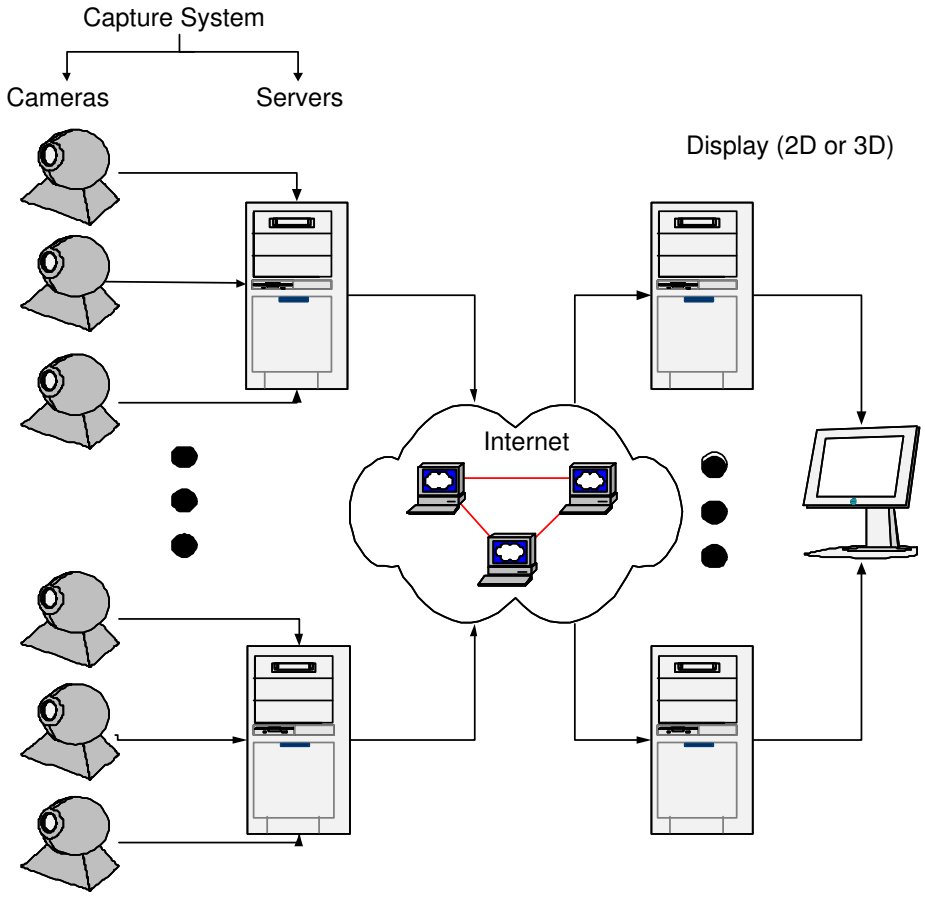


Figure 1.1: Multiview Video System over Internet

1.2.1 Video Acquisition

The function of the video acquisition system is to obtain multiple synchronized video streams through a camera array. Video acquisition is essentially a plenoptic function sampling process. Each camera represents a sample point in the space. Given the assumption of constant radiance along a light ray, samples can be taken from an arbitrary surface surrounding the convex hull of a scene of interest. Namely, cameras can be placed on any arbitrary surface that encloses the scene. Light field [16] and Lumigraph [17] choose the surface to be a box. A spherical surface is used in the spherical light field [23, 24]. The number of samples is directly related to the rendering quality. An interesting question is how many samples are needed for anti-aliasing reconstruction. The normal practical solution is to use oversampling to counter the undesirable aliasing effect in the output display since the sampling rate is affected by numerous factors such as scene geometry, textures on the scene

surface, reflection property of scene objects, motion of the objects, etc., which makes it hard to determine the exact sample size. Chai et. al. [25] give a theoretical analysis on plenoptic sampling. They found that the spectral support of a light field signal is bounded only by the minimum and maximum depth of scene objects. Zhang and Chen [26] proposed a generalized sampling strategy to extend the plenoptic sampling analysis. Some approximate knowledge on the scene geometry could help reduce the sample size. The video acquisition subsystem includes functions such as camera calibration, hardware or software-based video synchronization.

1.2.2 Video Preprocessing

The multiple synchronized video streams might need to be preprocessed before compression and rendering to reduce the amount of data, to accelerate the rendering speed, and to improve the rendering quality. Implicit or explicit geometry information about the scene such as per-pixel depth map or 3D scene mesh model can be obtained in the video preprocessing stage and used in the rendering subsystem.

1.2.3 Video Compression

A practical multiview video system might need hundreds, if not thousands, of cameras to capture the real-world dynamic scene to reconstruct the image at an arbitrary viewpoint. Hundreds of video cameras will generate a tremendous amount of video data for storage and transmission. Video compression is indispensable to reduce the size of video data since there exist temporal and inter-view redundancies among multiple synchronized video streams [27, 28].

1.2.4 Video Transmission

The objective of MVSN is to deliver the multiview video to millions of users so that they can enjoy the feeling of immersion. Different users might experience different network environments and use different hardwares. The challenge of video transmission is how to design good mechanisms so that the perceived quality of experience is proportional to the capability of users' hardware and network environment. The simplest case is to deliver multiview videos over a dedicated network where every user has the same capable hardware like today's TV networks. The ideal case is to deliver multiview videos over a QoS (Quality of Service) capable network [29, 30, 31] where the perceived quality of experience can be guaranteed by the inherent mechanisms built into the networks. The interesting case is how

to design sub-optimal and effective schemes so that the system can be successfully deployed in the current best-effort Internet.

1.2.5 Rendering

The rendering subsystem reconstructs the image at an arbitrary viewpoint from the available texture and geometry information based on the user inputs and outputs the rendered images in different types of displays. The rendering subsystem might be deemed as the core subsystem of MVSN since it determines the actual realization of other subsystems. Users directly interact with the rendering subsystem. Rendering methods can be grouped into three main categories: rendering without geometry, rendering with implicit geometry, and rendering with explicit geometry [13]. Rendering without geometry such as light field rendering [16] reconstructs the image at a new viewpoint directly from captured videos without the help of geometry. Rendering without geometry does not need a video preprocessing subsystem to obtain the geometry information. Rendering with implicit geometry calculates the 3D geometry information based on projection and positional correspondence and then uses the geometry to help improve the rendering quality. View interpolation [32] and view morphing [33] are rendering methods using implicit geometry. Rendering with explicit geometry normally uses direct 3D geometry information such as a mesh model or volume model to help reduce the video data size and improve the rendering quality. Rendering without geometry normally needs more cameras to capture real-world dynamic scenes than rendering with implicit or explicit geometry, and thus generates a larger amount of video data to store and transmit. In this thesis, we assume that rendering without geometry or rendering with implicit geometry is used in the rendering subsystem.

1.2.6 Non-realtime vs. Realtime

MVSN can be further grouped into two classes, realtime MVSN and non-realtime MVSN. Non-realtime MVSN first performs computationally intensive tasks such as depth estimation and compression offline, while making transmission and decompression and rendering as the online tasks. Realtime MVSN requires that acquisition, preprocessing, compression, transmission, and rendering should all be done online. The essential difference between realtime and non-realtime MVSN is how to make a tradeoff among computational capability, processing time and rendering quality. To achieve the same rendering quality, realtime MVSN normally demands more computational power, more stringent transmission latency, and less processing time than the non-realtime one. Given enough computational power and

abundant network capacity, all non-realtime MVSN can arguably be converted to realtime MVSN. From the other perspective, all the technologies that enable realtime MVSN can be used in non-realtime MVSN while the opposite might not work. Therefore, realtime is one of design objectives for MVSN.

1.3 Multiview Video Compression

In this thesis, we focus our work on solving issues related to multiview video compression subsystems. The scene representation format is assumed to be image and depth map. Namely, real world scenes are captured by multiple synchronized cameras and the resulting image and depth sequences are the input to the multiview video compression subsystem. Fig. 1.2 illustrates the generic architecture of the multiview video compression subsystem. As with most compression schemes, multiview video compression is composed of an encoder and a decoder. The input to the multiview video encoder is the image and depth sequences $\{S_i\}, i = 1, \cdots, n$ and the output is compressed multiview video streams. The multiview video decoder takes the compressed multiview video streams as its input and outputs the approximate multiple video and depth streams $\{\hat{S}_i\}, i = 1, \cdots, n$.

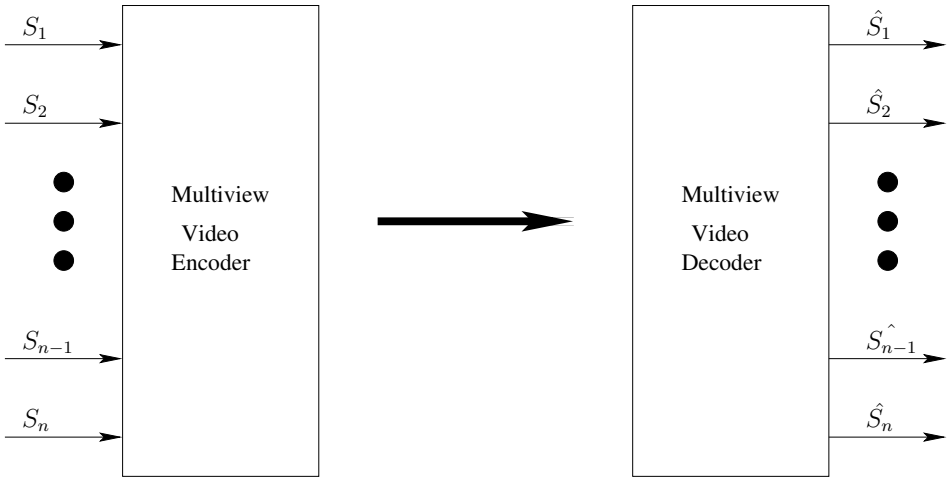


Figure 1.2: The Generic Architecture of Multiview Video Compression Subsystem

The challenge of the multiview video compression subsystem is mainly how to achieve a better tradeoff between the compression ratio and computational complexity by exploiting the intra-stream and inter-stream redundancies among multiple synchronized video and its associated depth streams.

1.4 Thesis Contributions

In this thesis, we concentrate on the design of novel joint multiview video compression and distributed multiview video compression algorithms to improve the performance of multiview video compression subsystem. The thesis makes the following contributions.

- *We propose a novel neighbor-based joint multiview video compression scheme*. In particular, a new stream encoding order algorithm is designed to help automatically decide the video stream encoding order and enable each stream to find its best reference streams and thus improve the performance.

- *A novel multiview video transcoder is designed and implemented.* The proposed multiview video transcoder can be used to encode multiple compressed video streams and reduce the cost of video acquisition subsystem.

- *A learning-based multiview video compression scheme is proposed*. The novel multiview video compression algorithms take advantage of recent advances on semi-supervised learning algorithms and achieve the compression by finding a sparse representation of images.

- *We propose two novel symmetric distributed source coding algorithms: EETG and SNS-SWC*. Both EETG and SNS-SWC are syndrome-based schemes and able to achieve the whole Slepian-Wolf rate region. EETG is able to handle mismatched bits at encoders and realizes the benefits of simplified code construction for distributed source coding schemes using extended Tanner graph. SNS-SWC has two independent decoders and simplifies the decoding process of the symmetric distributed source coding schemes.

- *A novel symmetric distributed multiview video coding scheme is proposed.* The scheme is based on the symmetric distributed source coding scheme SNS-SWC. It enables flexible rate allocation between distributed multiview video encoders. It can achieve significant performance improvement from separate H.264 coding schemes at high rate.

1.5 Thesis Organization

The rest of thesis is organized as follows. We first present some background knowledge on information theory and give an overview of previous multiview video coding schemes

in Chapter 2. In Chapter 3, we elaborate on the details of the neighbor-based multiview video compression algorithm. The multiview video transcoder is discussed in Chapter 4. Learning-based multiview video coding scheme is presented in Chapter 5. In Chapter 6, we present two syndrome-based symmetric distributed source coding schemes. The design and implementation of the distributed multiview video coding scheme is elaborated in Chapter 7. We conclude the thesis and propose some future directions of research in Chapter 8.

Chapter 2

Background

Shannon's information theory [34] lays the foundation for data compression and communication. This thesis focuses on multiview video compression which is underpinned by information theory. In this chapter, we provide some background knowledge on information theory and give an overview of previous multiview video coding schemes. A more complete discussion on these subjects can be found in standard textbooks and popular tutorials [34, 35, 36, 37, 38, 28, 39, 27].

2.1 Source Coding

In this section, we review the basic theory of source coding. Source coding theory specifies the minimum amount of bits needed to represent the source information without redundancy. Given multiple random sources $X_1, X_2, \cdots, X_n$, there are three general approaches to compress them: separate source coding, joint source coding and distributed source coding. Each method will be described in the following sections. We will illustrate the idea using two sources, X and Y.

2.1.1 Separate Source Coding

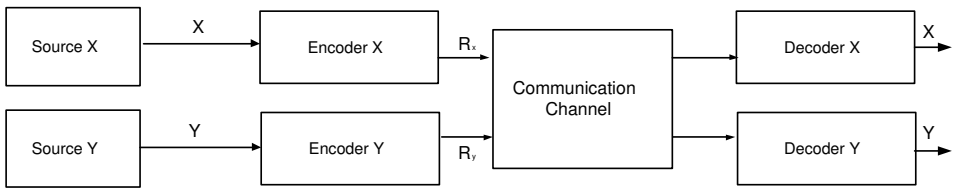


Figure 2.1: Separate Source Coding Architecture

Separate source coding (SSC) encodes and decodes each source X and Y separately. The encoders and decoders of two sources are totally independent as shown in Fig. 2.1. The

achievable compression rate, R_X and R_Y, is greater than or equal to the source entropy which measures the uncertainty of a random variable. Namely,

$$R_X \geq H(X) \tag{2.1}$$

$$R_Y \geq H(Y) \tag{2.2}$$

where $H(X)$ and $H(Y)$ are the entropy of the source X and Y.

2.1.2 Joint Source Coding

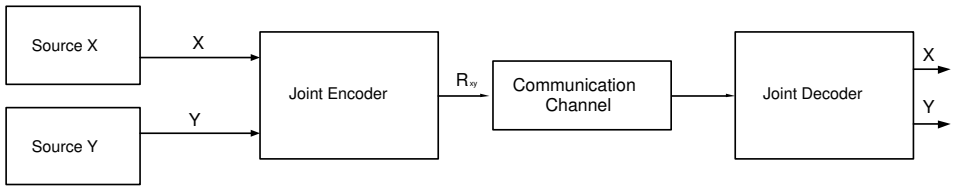


Figure 2.2: Joint Source Coding Architecture

Joint source coding (JSC) encodes and decodes two sources X and Y jointly. Fig. 2.2 illustrates its architecture. The essence of joint source coding is to exploit the correlation between two sources at the encoder for efficient compression. Joint source coding can only be done centrally. Namely two sources X and Y must both be available at the encoder to make compression possible. If two sources X and Y are collected at physically separate nodes, for examples, two sensors in sensor networks, extra communication cost is needed to jointly encode them. JSC normally demands a computationally complicated encoder. The achievable rate of joint source coding, R_{XY}, is equal to or greater than the joint entropy of two sources to achieve lossless compression. In other words,

$$R_{XY} \geq H(X,Y) \tag{2.3}$$

where $H(X,Y)$ is the joint entropy of the source X and Y.

2.1.3 Distributed Source Coding

Distributed source coding (DSC) separately encodes and jointly decodes two sources X and Y. Fig. 2.3 shows its architecture. The correlation between two source, X and Y, can be modeled as a virtual channel. Unlike JSC, the essence of DSC is to exploit the source correlation at the decoder for efficient compression. The information-theoretic bound proved by Slepian and Wolf [40] in the 1970s shows that efficient lossless distributed compression can be achieved. Wyner and Ziv [41] extended the result to the case of lossy coding

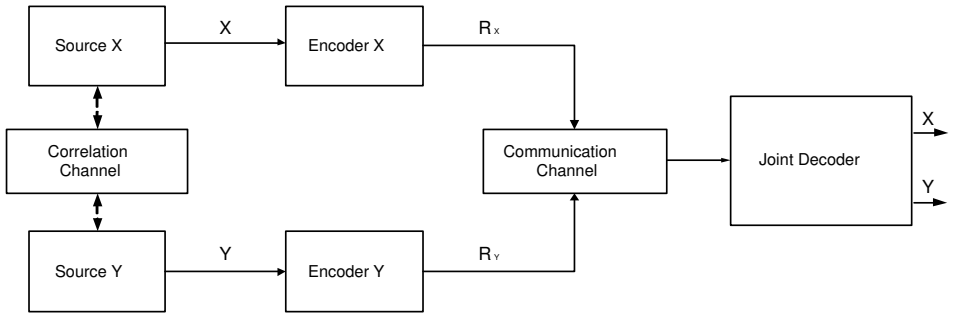


Figure 2.3: Distributed Source Coding Architecture

with side information. Therefore, DSC is further grouped into two categories: lossless distributed source coding and lossy distributed source coding. When two sources are captured by physically isolated nodes, DSC does not incur extra communication cost to achieve a similar compression ratio to JSC. Normally DSC has a simple encoder and a complicated decoder.

Lossless Distributed Source Coding

Lossless distributed source coding aims at lossless compression of two sources, X and Y. Slepian and Wolf [40] established the rate region to achieve the lossless compression of two sources. Fig. 2.4 shows the Slepian-Wolf rate region, which is the set of all achievable rate pairs (R_X, R_Y). Namely, the achievable rate R_X and R_Y needs to satisfy the following equations:

$$R_X \geq H(X|Y) \tag{2.4}$$

$$R_Y \geq H(Y|X) \tag{2.5}$$

$$R_X + R_Y \geq H(X,Y) \tag{2.6}$$

where $H(X|Y)$ and $H(Y|X)$ are the conditional entropy, $H(X,Y)$ is the joint entropy.

A special case for distributed source coding is to compress one source X when the other source Y is available as side information at the decoder. This scheme is called asymmetric distributed source coding (ADSC) as illustrated in Fig. 2.5. Asymmetric lossless distributed source coding can be achieved with rate $R_X \geq H(X|Y)$ while the side information Y is losslessly compressed by using separate source coding with rate $R_Y = H(Y)$. This case corresponds to the corner point in the rate region of Slepian-Wolf theorem shown in Fig. 2.4. Distributed source coding is dual to channel coding [42]. The distributed source coding schemes that can achieve an arbitrary point in Slepian-Wolf rate region is normally referred to as symmetric distributed source coding (SDSC).

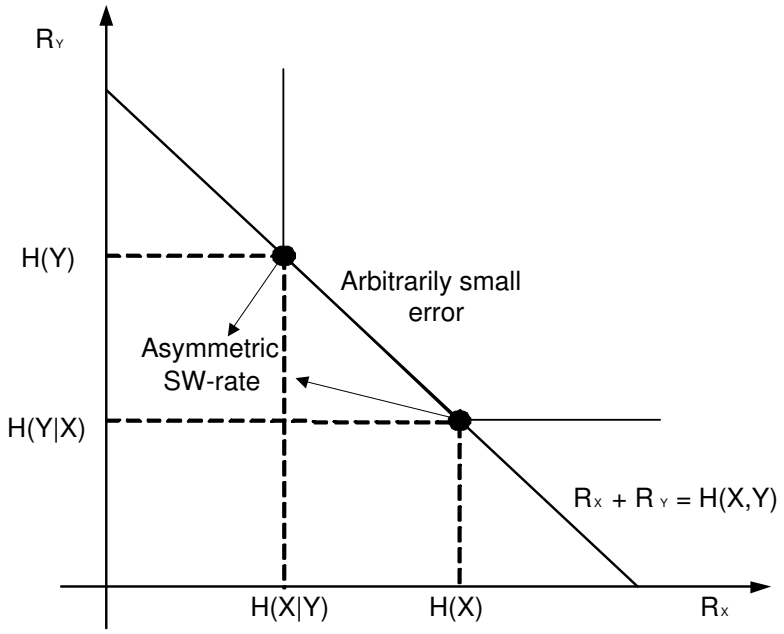


Figure 2.4: Rate Region for Slepian-Wolf Coding (Two Sources)

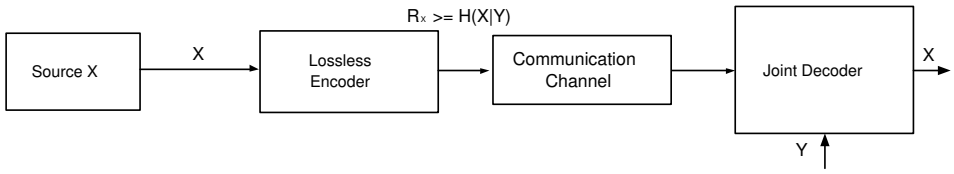


Figure 2.5: Asymmetric Lossless Slepian-Wolf Coding Scheme (Two Sources)

Lossy Distributed Source Coding

Lossy distributed source coding performs the compression with respect to a distortion measure. Wyner and Ziv [41] extend the Slepian-Wolf Theorem for lossless compression to establish information-theoretic bounds for lossy compression with side information at the decoder. For two correlated i.i.d random sequences, X and Y, the source X is encoded without access to the side information Y; while the decoder, with access to Y, decompresses X subject to a distortion $D = E[d(X, \hat{X})]$, where $\hat{X}$ is the reconstruction of X at decoder. The Wyner-Ziv rate-distortion function $R_{X|Y}^{WZ}$ specifies the achievable lower bound for the bit-rate with a distortion D. However, unlike Slepian-Wolf coding which achieves lossless compression with the same bit-rate as conventional joint source coding, there is a rate loss between Wyner-Ziv coding compared with the lossy joint source coding. Namely, $R_{X|Y}^{WZ}(D) - R_{X|Y}(D) \geq 0$, where $R_{X|Y}$ is the rate when the encoder has access to the side information. Zero rate loss can only be achieved in the case of Gaussian memoryless sources and mean-square error distortion.

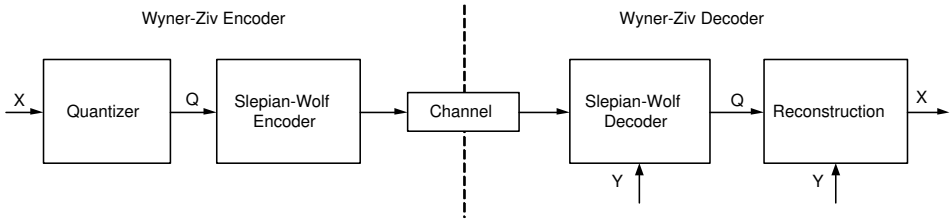


Figure 2.6: Practical Wyner-Ziv Coding Architecture

A practical Wyner-Ziv coder is generally composed of a quantizer followed by a Slepian-Wolf coder, as illustrated in Fig. 2.6. Previous research on Wyner-Ziv coder mainly focus on Gaussian sources.

Multiterminal Source Coding

Multiterminal source coding [43] addresses the problem of separate encoding and joint decoding of multiple correlated sources under a distortion constraint. It can be viewed as the lossy version of Slepian-Wolf coding. Multiterminal source coding is more general than Wyner-Ziv coding. Berger and Tung [43, 44] gave an inner rate region which is now called Berger-Tung inner rate region. For joint Gaussian sources and mean square error distortion measure, it is referred as the quadratic Gaussian multiterminal source coding problem. The rate region of the quadratic Gaussian two-terminal source coding problem is recently completely characterized [45, 46]. The rate region for quadratic Gaussian source

coding with more than two terminals is still unknown.

Given two sources X and Y which are joint Gaussian random variables with variances σ_X^2 and σ_Y^2 and correlation coefficient $\rho = \frac{E(XY)}{\sigma_X \sigma_Y}$ and corresponding distortion D_X and D_Y, the Berger-Tung inner rate region [43, 44] is

$$\hat{R}^{BT}(D_X, D_Y) = \hat{R}_X^{BT}(D_X, D_Y) \cap \hat{R}_Y^{BT}(D_X, D_Y) \cap \hat{R}_{XY}^{BT}(D_X, D_Y) \tag{2.7}$$

where

$$\hat{R}_X^{BT}(D_X, D_Y) = \left\{ (R_X, R_Y) : R_X \geq \frac{1}{2} log_2^+ \left((1 - \rho^2 + \rho^2 2^{-2R_Y}) \frac{\sigma_X^2}{D_X} \right) \right\} \tag{2.8}$$

$$\hat{R}_Y^{BT}(D_X, D_Y) = \left\{ (R_X, R_Y) : R_Y \geq \frac{1}{2} log_2^+ \left((1 - \rho^2 + \rho^2 2^{-2R_X}) \frac{\sigma_Y^2}{D_Y} \right) \right\} \tag{2.9}$$

$$\hat{R}_{XY}^{BT}(D_X, D_Y) = \left\{ (R_X, R_Y) : R_X + R_Y \geq \frac{1}{2} log_2^+ \left((1 - \rho^2) \frac{\beta_{max} \sigma_X^2 \sigma_Y^2}{2 D_X D_Y} \right) \right\} \tag{2.10}$$

with $\beta_{max} = 1 + \sqrt{1 + \frac{4\rho^2 D_X D_Y}{(1-\rho^2)\sigma_X^2 \sigma_Y^2}}$ and $log_2^+ x = max\{log_2 x, 0\}$.

Yang et al. [47] propose a framework for practical multiterminal source coding based on Slepian-Wolf coded quantization, which is the combination of vector quantization and Slepian-Wolf coding. Algorithms using TCQ [48] and Turbo/LDPC codes for Slepian-Wolf coding can almost achieve the sum rate bound of quadratic Gaussian multiterminal source coding.

2.2 Channel Coding

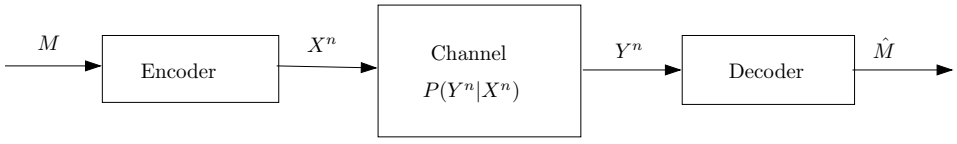


Figure 2.7: Channel Model

Channel coding addresses the problem of how a data source can be efficiently and reliably communicated over a noisy channel. Namely, channel capacity decides the maximum number of distinguishable signals that can be used for reliable communication. Channel coding theory designs particular code and specific encoding and decoding algorithm to achieve the channel capacity. Fig. 2.7 gives a block diagram of communication system. In the figure, M is a set of source messages. X^n is a set of channel code and Y^n is a set of received signals. $\hat{M}$ is a set of recovered source messages. The communication channel

is modeled as a probability transition function $P(Y|X)$. The encoder maps a source word m to a channel code x^n which is transmitted through the channel and received by the decoder as y^n. The decoder recovers the original message $\hat{m}$ from y^n. To achieve the reliable communication, the challenge is to design good channel code and encoding and decoding algorithm so that $\hat{m}$ is equal to m with high probability.

It is well known that channel capacity is represented as the maximum mutual information between channel input alphabet and output alphabet. Formally,

$$C = \max_{p(x)} I(x; y) \tag{2.11}$$

where the maximum is taken over all possible input distribution $p(x)$.

A channel is normally denoted as $(\mathcal{X}, P(y|x), \mathcal{Y})$, where $\mathcal{X}$ and $\mathcal{Y}$ are a finite set of input alphabets and output alphabets, and $P(y|x)$ is probability transition functions. Let M denote a set of possible source messages $\{1, 2, \cdots, \mathcal{M}\}$. An $(n, \mathcal{M})$ code $\mathcal{C}$ for channel $(\mathcal{X}, P(y|x), \mathcal{Y})$ consists of a set of source messages, an encoding function and a decoding function. The encoding function, $\mathcal{E}(\cdot)$, is the mapping from a message to a codeword.

$$\mathcal{E} : 1, 2, \cdots, \mathcal{M} \rightarrow \mathcal{X}^n$$

The decoding function, $\mathcal{D}(\cdot)$, recovers the mostly likely source message sent by the encoder from y^n, the estimated value of x^n.

$$\mathcal{D} : y^n \rightarrow \{1, 2, \cdots, \mathcal{M}\}$$

The maximal probability of error for an $(n, \mathcal{M})$ code is defined as

$$P_e^{(n)} = \max_{i \in \{1,2,\cdots,\mathcal{M}\}} P(\mathcal{D}(y^n) \neq i | x^n = \mathcal{E}(i)) \tag{2.12}$$

The goal is to find a channel code $\mathcal{C}$ so that $P_e^{(n)}$ approaches 0 as the codeword length n approaches ∞. The rate R of an $(n, \mathcal{M})$ code $\mathcal{C}$ is defined as $R = \frac{log_2 \mathcal{M}}{n}$ bits per transmission. Information theory shows that with proper design of $\mathcal{E}(\cdot)$ and $\mathcal{D}(\cdot)$, $P_e^{(n)} \rightarrow 0$ as $n \rightarrow \infty$ if the rate R is less than or equal to the channel capacity C, $R \leq C$.

In the past 60 years, significant research effort has been invested in finding good channel codes. Most practical channel codes are binary linear channel codes. Binary linear channel codes are a subclass of channel codes. Gallager [35] proves that linear channel codes can achieve channel capacity for some practical channels such as discrete memoryless channels. Linear channel codes have two classes: linear block codes and convolutional codes. Let

$k = log_2\mathcal{M}$, an $(n, \mathcal{M})$ code is normally referred as an (n, k) channel code which maps k input source bits into n bits for transmission over a noise channel. An (n, k) linear channel code can be defined by its generator matrix, G. The null space of G is called the parity check matrix, H. H is orthogonal to G; namely $H \cdot G^T = 0$, where G^T is the transpose of G. The parity check matrix is normally used at a decoder to check whether the decoded codeword y^n is a valid codeword of $\mathcal{C}$ or not.

2.3 Low Density Parity Check Codes

Low density parity check (LDPC) code is a linear block channel code. LDPC codes are a class of capacity-approaching channel codes. They are originally introduced by Gallager [49] and then ignored for nearly 40 years and rediscovered by MacKay and Neal [50] and has since attracted considerable interest. In this section, we will give a brief overview on LDPC codes and its decoding algorithm.

Any linear block codes can be defined by its parity check matrix H. Each parity check matrix has a bipartite graph representation. The bipartite graph is commonly referred to as Tanner graph [51]. Given an $m \times n$ parity check matrix H, m rows are mapped to m check nodes and n columns are mapped to n variable nodes. Variable nodes represent bits and check nodes represent the constraints among bits. Table 2.1 and Fig. 2.8 shows a parity check matrix and its corresponding Tanner graph. LDPC codes are linear block codes obtained from a sparse parity check matrix. Formally, a sequence of $m \times n$ matrices is called sparse if mn approaches ∞ and the number of nonzero elements in the matrices is always less than $c \max(m, n)$.

Table 2.1: The parity check matrix corresponding to the bipartite graph shown in Fig. 2.8. If the element in row i and column j is 1, an edge is connected from the corresponding variable node to the check node. A 0 means no edge between corresponding nodes.

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Each LDPC code belongs to a code ensemble. A code ensemble is represented by a degree distribution pair $(\lambda(x), \rho(x))$, where $\lambda(x)$ is the variable node degree distribution and $\rho(x)$ is check node degree distribution. $\lambda(x)$ and $\rho(x)$ are polynomial functions and are defined by

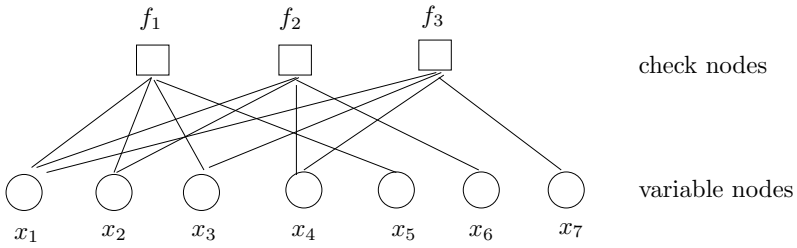


Figure 2.8: The Tanner graph corresponding to the parity check matrix in Table 2.1. Check nodes represent constraints of the code and correspond to the rows of the parity check matrix. Variable nodes represent bits and correspond to the columns of the parity check matrix.

$$\lambda(x) = \sum_{i \geq 2}^{d_v} \lambda_i x^{i-1}, \ \rho(x) = \sum_{i \geq 2}^{d_c} \rho_i x^{i-1}$$

where i is the node degree, and d_v (d_c) is the maximal variable (check) node degree, and λ_i (ρ_i) represents the fraction of edges emanating from variable (check) nodes of degree i. In addition, the sum value of λ_i and ρ_i must be equal to 1.

$$\sum_{i}^{d_v} \lambda_i = 1, \ \sum_{i}^{d_c} \rho_i = 1.$$

Given the node degree distribution pair $(\lambda(x), \rho(x))$ of a code ensemble, we can easily calculate its rate based on the fact that the total number of edges emanating from variable nodes are equal to the total number of edges emanating from the check nodes. Namely,

$$r(\lambda, \rho) = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx}.$$

Richardson and Urbanke [52, 53] show that almost all codes in a code ensemble have similar error correction capability. Therefore, the individual behavior of a specific code can be determined by studying the average behavior of the ensemble. They propose a density evolution algorithm to analyze the performance of a code ensemble and put forward a specific procedure to construct capacity-approaching codes [53]. Sum-product algorithm [54] is used to decode LDPC codes. Sum-product algorithm is also called message-passing algorithm or belief propagation algorithms [55] which achieve successful decoding by iteratively passing messages including probabilities on bits between variable nodes and check nodes. The detailed analysis on LDPC codes and the performance of decoding algorithm can be found in [36].

2.3.1 Sum-product Decoding Algorithm

The sum-product decoding algorithm is an iterative soft decision message-passing algorithm. The algorithm takes the *a priori* probabilities of each bit as input and iteratively refines the estimation of the marginal probability of each bit. The output probability by the decoder is called the *a posteriori* probability and is used to decide the bit value. The probabilities used in the sum-product algorithm are normally represented as *log-likelihood ratios* (LLR).

Given a binary variable x and its respective probability $p(x = 0)$ and $p(x = 1)$, its log-likelihood ratio is $L(x) = log\left(\frac{p(x=0)}{p(x=1)}\right)$. $L(x)$ is positive if $p(x = 0) > p(x = 1)$ and the greater the ratio between $p(x = 0)$ and $p(x = 1)$, the larger the positive value of $L(x)$ and the more confident we are that $x = 0$. Conversely, if $p(x = 0) < p(x = 1)$, $L(x)$ is negative and the smaller the ratio between $p(x = 0)$ and $p(x = 1)$, the larger the negative value for $L(x)$ and the more sure we are that $x = 1$. Thus the sign of $L(x)$ provides the value of x and the magnitude $|L(x)|$ gives the reliability of the decision.

The aim of the sum-product algorithm is to compute the *maximum a posteriori probability* (MAP) for each codeword bit, $P_i = P(c_i = 1|N)$, which is the probability that i-th codeword bit is 1 conditional on the event N that all the parity-check constraints are satisfied. The sum-product algorithm iteratively computes an approximation of the MAP value for each code bit. Each iteration is comprised of two message-passing processes over the Tanner graph representing the corresponding LDPC parity-check matrix. One is message passing from variable nodes to check nodes. The other is message passing from check nodes to variable nodes. The message is the extrinsic information estimated at corresponding variable and check nodes.

The extrinsic message from check node j to variable node i, $u_{j,i}$, is the LLR of the probability that bit i causes parity-check j to be satisfied. It can be computed using the following equation:

$$u_{j,i} = 2tanh^{-1}\left(\prod_{i' \in \mathcal{N}(j), i' \neq i} tanh\left(\frac{v_{i',j}}{2}\right)\right) \tag{2.13}$$

where $\mathcal{N}(j)$ is the set of variable nodes that connect the check node j and $v_{i',j}$ is the extrinsic message from variable node i' to check node j.

The extrinsic message from variable node i to check node j, $v_{i,j}$, is the sum of its initial LLR, $r_i = log\frac{p(c_i=0)}{p(c_i=1)}$, and the LLRs from every connected check nodes excluding the check

node j. Namely,

$$v_{i,j} = r_i + \sum_{j' \in \mathcal{N}(i), j' \neq j} u_{j',i} \tag{2.14}$$

where $\mathcal{N}(i)$ is the set of check nodes that connects variable node i.

The sum-product algorithm repeatedly updates $u_{j,i}$ and $v_{i,j}$ at each iteration. The value of coded bit i is estimated based on the sign of total LLRs of the variable node i, which is the sum of initial a priori probability and all the LLRs from connected check nodes.

$$L_i = r_i + \sum_{j \in \mathcal{N}(i)} u_{j,i} \tag{2.15}$$

The sum-product algorithm stops if the product of the parity check matrix and the estimated codeword is 0 or the maximum number of iteration is reached. Algorithm 1 gives the pseudo code of the sum-product algorithm.

2.4 Network Coding

Network coding [37, 38] is a recent breakthrough in information theory. Network coding characterizes the maximum rate at which the information at a source node can be multicast to multiple nodes. It stipulates that the maximal achievable multicast throughput is less than or equal to the maximum flow capacity dictated by the Max-flow Min-cut theorem [56]. The idea of network coding was initially introduced by Yeung and Zhang [57] and fully developed by Ahlswede et al. [56]. The network code construction algorithm to achieve the Max-flow Min-cut bound is devised in [58, 59]. Network coding theory shows that coding at intermediate network nodes is more efficient than the traditional routing mechanism which only performs the function of replication and forwarding. Fig. 2.9 illustrates a simple example to show that network coding can achieve better throughput than routing. Source S multicasts two bits b_1 and b_2 to terminal nodes T_1 and T_2. All edges have the same delay. Each edge in the network can only carry one data unit per unit time. In Fig. 2.9(a), node v_3 only has the routing capacity and can only replicate and forward. Thus edge (v_3, v_4) can only carry b_1 or b_2 in a time unit. The capacity is 1.5 bits/unit time. In Fig. 2.9(b), node v_3 can do network coding such as bit xor $\oplus$ operation in this example. $b_1 \oplus b_2$ can be transmitted in Edge (v_3, v_4). The terminal node T_1 can recover b_2 by $b_2 = b_1 \oplus (b_1 \oplus b_2)$ since it receives b_1 from the edge (v_1, T_1). Similarly $T2$ can recover b_1. The capacity is 2 bits/unit time.

Formally, a communication network can be represented by a finite directed acyclic graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes in the network and $\mathcal{E}$ is the set of edges

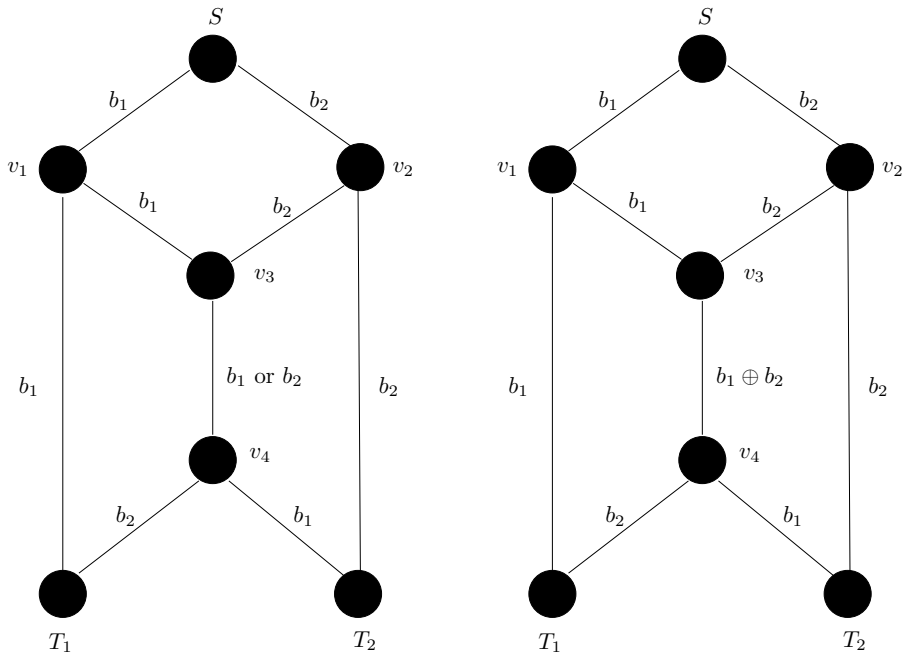
Algorithm 1 Sum-product Decoding Algorithm

H : parity-check matrix;
n : number of variable nodes;
m : number of check nodes;
I_{max} : maximum number of allowed iterations;
r_i : the initial LLR of i-th coded bit;
L_i : the total LLRs of i-th coded bit;
$u_{j,i}$: message from check node j to variable node i;
$v_{i,j}$: message from variable node i to check node j;
z_i : the i-th estimated coded bit;

for i = 1 to n **do**
  for j = 1 to m **do**
    $v_{i,j} = r_i$;
  end for
end for
$k = 0$
$Finished = FALSE$
repeat
  update the message $u_{j,i}$ based on Eqn. 2.13;
  for i = 1 to n **do**
    compute L_i based on Eqn. 2.15;

$$z_i = \begin{cases} 1 & L_i \leq 0 \\ 0 & L_i > 0. \end{cases} \tag{2.16}$$

  end for
  if $k == I_{max}$ or $Hz^T = 0$ **then**
    $Finished = TRUE$;
  else
    update the message $v_{i,j}$ based on Eqn. 2.14;
    $k = k + 1$;
  end if
until $Finished$



(a) Routing

(b) Network Coding

Figure 2.9: Routing vs. Network Coding

in the network. Edges in $\mathcal{E}$ are assumed to be ordered based on the partial order induced by the acyclicity of the network. A directed edge $e_{ij} = (v_i, v_j)$ represents a channel from node v_i to node v_j. Node v_i is called the tail of edge e_{ij} and node v_j is called the head of edge e_{ij}. Let $In(v_i) = \{(v_j, v_i) : (v_j, v_i) \in \mathcal{E}\}$ and $Out(v_i) = \{(v_i, v_j) : (v_i, v_j) \in \mathcal{E}\}$. Namely, $In(v_i)$ is the set of incoming edges to node v_i and $Out(v_i)$ is the set of outgoing edges from node v_i. Multiple edges between two nodes are allowed, and each edge can carry one symbol in field $\mathcal{F} = GF(q)$, where GF denotes Galois field.

Let $s \in \mathcal{V}$ be a source node, and $\mathcal{T} \subset \mathcal{V}$ be a subset of nodes of $\mathcal{V}$ whose members are called sink nodes. A multicast on $\mathcal{G}$ disseminates information from the source s to sink nodes in $\mathcal{T}$. The source messages are defined as a row vector, $\mathbf{x} = (x_i : i = 1, \cdots, \omega)$, where $\mathbf{x} \in \mathcal{F}^\omega$ and $\omega = |Out(s)|$. When messages are multicast, the ω symbols in the vector $\mathbf{x}$ are mapped onto edges in $Out(s)$ and transmitted over the corresponding channels in one use of the network. Define an $\omega \times |\mathcal{E}|$ matrix $A = [A_{i,j}]$ where $A_{i,j} = 1$ if the edge e_j is the ith edge in $Out(s)$ and $A_{i,j} = 0$ if otherwise.

A network code of network $\mathcal{G}$ is specified by a set of local encoding kernels $\{k_{e_u,e_v} : e_u \in In(v), e_v \in Out(v), \forall v \in \mathcal{V}\}$ and the message set $\mathcal{C}$. Suppose that the message transmitted in edge e_u is m_u, the message transmitted on the edge e_v is computed by the

following formula.

$$m_v = \sum_{u \in In(v_i)} k_{e_u,e_v} m_u.$$

Each edge e has a global coding kernel, f_e, which is a ω-dimensional column vector in field $\mathcal{F}$. The message transmitted on the edge e_v can also be calculated from its global coding kernel. Namely,

$$m_v = \mathbf{x} f_{e_v}.$$

The global encoding kernels are determined by local encoding kernels. Define the $|\mathcal{E}| \times |\mathcal{E}|$ transformation matrix $K = [K_{u,v}]$ in network $\mathcal{G}$ as

$$K_{u,v} = \begin{cases} k_{e_u,e_v} & e_u \in In(v), e_v \in Out(v), \forall v \in \mathcal{V}; \\ 0 & \text{otherwise.} \end{cases}$$

From [59], we know that for an acyclic network, $K^N = 0$, for some positive integer N. We can then get the transfer matrix of the network $F = (I - K)^{-1}$ [59]. For a sink node $t \in \mathcal{T}$, let $n_t = |In(t)|$, and define an $\mathcal{E} \times n_t$ matrix $B = [B_{i,j}]$ where $B_{i,j} = 1$ if the edge e_i is the jth edge in $In(t)$ and $B_{i,j} = 0$ if otherwise. Therefore, the received message vector at the sink node t is $y_t = \mathbf{x}AFB$.

The task of network code construction is to assign appropriate local encoding kernels so that at every sink node the original source message $\mathbf{x}$ can be losslessly recovered from the received message $\mathbf{y}$. Namely the matrix $\mathcal{M} = AFB$ should be invertible at every sink node. Given the known network topology, Jaggi et al. [60] put forward a deterministic algorithm for multicast network code construction. Ho et al. [61] prove that successful decoding can be achieved with high probability when local encoding kernels are randomly assigned to a value in field $GF(q)$ if q is large enough. Chou et al. [62] propose a practical scheme to use network codes in real network environments.

Inspired by network coding, Yeung and Cai [63, 64] introduce the concept of network error correction which is the network generalization of classical point-to-point error correction. Specifically, an $|\mathcal{E}|$-tuple error vector $\mathbf{z}$ is defined to represent the error occurred on an edge. Therefore, the received vector at a sink node t can now be denoted as

$$\begin{aligned} y_t &= (\mathbf{x}A + \mathbf{z})FB & (2.17) \\ &= \mathbf{x}F_{s,t} + \mathbf{z}F_t. & (2.18) \end{aligned}$$

where $F_{s,t} = AFB$ is transfer matrix of message transmission and $F_t = FB$ is the transfer matrix of error transmission. The classical error correction is a special case of network error correction when $F_{s,t}$ and F_t are both identity matrices. A more comprehensive discussion of network code can be found in [37, 38].

2.5 Multiview Video Coding

Multiview video coding is the application of source coding theory on multiview video data. The objective of multiview video coding is to achieve the joint entropy of the multiple correlated video streams. Multiview video coding aims at reducing the amount of video data for storage and transmission while maintaining enough fidelity to not negatively affect the rendered image quality. Multiview video coding addresses the problem of compressing the videos captured on a dynamic scene. Multiview image coding is the term used for the problem of compressing images captured on a static scene. Based on the underlying source coding scheme, multiview video coding schemes can be classified into two categories: Joint Multiview Video Coding (JMVC) and Distributed Multiview Video Coding (DMVC).

2.5.1 Joint Multiview Video Coding

Joint multiview video coding exploits the joint statistics of multiple video streams at the encoder and uses the joint source coding as the underlying entropy coding scheme to reduce the redundancies among multiple video streams. JMVC schemes normally have a computationally complicated encoder and relatively simple decoder.

Multiview image compression has been widely investigated since light field rendering [16, 17] was invented. Early research predominantly focus on joint multiview image coding schemes. Multiview images used in image-based rendering are usually taken around a static object and thus have significant spatial coherence. Multiview image compression mainly aims at exploiting the spatial redundancy among image data to improve the compression efficiency without sacrificing the rendering quality. Levoy and Hanrahan [16] put forward a vector quantization approach to compress the light field arrays. Zhang and Li [65] use a multiple reference frame structure to compress the light field images. With multiple reference frame prediction, just-in-time rendering can be achieved. Luo et al. [66] adopt a wavelet transform and rebinning approach to compress concentric mosaics. Chang et al. [67, 68] incorporate disparity compensation and 2-D shape information into the discrete wavelet coder to improve the compression performance. Li et al. [69] do a performance evaluation on vector quantization block coder, reference block coder and wavelet coder.

Results show that a vector quantization block coder is computationally simple but has low compression ratio. A wavelet coder can achieve the best compression performance. However, it is the most complex. The reference block coder has good compression ratio with medium computational complexity. Magnor et al. [70] propose a new multiview image coding approach that employs the knowledge of 3-D scene geometry. Maitre et al. [71] propose a wavelet-based joint estimation and encoding approach for texture image and depth maps.

Joint multiview video compression needs to simultaneously reduce temporal and inter-view redundancy among multiple synchronized video streams taken at different viewpoints. The challenge is how to remove the inter-view redundancies. There are two general approaches: block-based approach and nonblock-based approach. Block-based approaches use blocks as a basic unit to estimate disparity in neighbor frames and remove redundancies using methods adopted by hybrid MPEG/H.26x video coding standard. MPEG2 Multiview Profile (MVP) [72] proposes a block-based stereoscopic coding (BBSC) approach to compress stereo video. Motion-compensated prediction (MCP) is used to reduce the temporal redundancies and disparity-compensated prediction (DCP) is used to reduce the inter-view redundancies. MVP first compresses, say, the left view, with a MCP-based monoview coding algorithm and then encodes the right view using both DCP from the left view and MCP. Chan et al. [73] and Lim et al. [74] extend MVP to compress multiview video sequences. Multiple video streams are classified into two types of streams: main stream and secondary stream. The main stream is like the left stream in MVP and is encoded only by a MCP-based MPEG2-like algorithm. Secondary streams are like the right stream in MVP and are compressed with algorithms based on both MCP and DCP. In MVP-like approaches [72, 73, 74], a frame in secondary streams can only choose reference frames from one reference view. A virtual view synthesis prediction approach is used to remove redundancies among video streams and is integrated into H.264/AVC standard in [75, 76, 77, 78]. Joint Video Team (JVT) of MPEG and ITU-T [79] is currently developing a Joint Multiview Video Model (JMVM) which is based on the H.264 hybrid video coding standard. Previous block-based joint multiview video coding schemes are all designed to compress the multiple uncompressed input video streams. Initially, research on block-based JMVC focuses on how to efficiently compress the multiview video texture data. Recently some researchers start developing methods to efficiently compress both multiview video texture and depth data. Zitnick et al. [3] use MPEG-like scheme to exploit temporal and spatial redundancy and to compress both texture and depth information. Na et al. [80] and Merkle et al. [81] use H.264-based schemes to compress multiview videos represented by a Multiview-video-

plus-depth (MVD) format. In Chapter 3, we propose a neighbor-based approach which allows each stream to choose multiple reference views. In Chapter 4, we put forward a multiview video transcoder which can process multiple compressed input video streams and remove inter-view redundancies among multiple compressed video streams. In Chapter 5, we propose a learning-based multiview video coding scheme to compress multiview videos represented by a MVD format [82].

Nonblock-based approaches estimate more accurate disparity or reconstruct 3D models to remove inter-stream redundancies. Tseng and Anastassiou [83] propose a MPEG2-compatible multiview video compression framework which combines the concept of 2D image processing and 3D computer graphics to estimate the 3D model and encode the multiview videos and construct the new views. Tzovaras et al. [84] and Malassiotis and Strintzis [85] advocate an object-based stereo video coding approach by estimating 3D motion and disparity. Wang et al. [86] and Chien et al. [87] use a mesh-based disparity estimation method to reduce spatial redundancies among video streams. Park et al. [88] propose an improved mesh-based disparity representation method optimized for view synthesis and stereo image compression. Fehn et al. [89] reduces the data to a single view with per-pixel depth map and then compresses and transmits it as an MPEG-2 enhancement. Gross et al. [4] and Lamboray et al. [90] transfer multiview video into 3D video fragments to exploit the spatio-temporal coherence. Ziegler et al. [91, 92] use a 3D scene model to help transform the video images into textures and then employs two-level hierarchical coder or 4D SPHIFT wavelet coder to exploit inter-frame spatio-temporal coherence. Kum et al. [93] propose a stream partitioning algorithm to group 3D video streams and a realtime compression algorithm for the grouped streams. Lamboray et al. [94] introduce a compression framework to encode time-varying 3D point samples and the approach allows arbitrary spatio-temporal navigation throughout the entire 3D streams at constant time. Nath and Dubois [95] propose a wavelet-based stereo video codec with SNR and spatial scalability. Smolic et al. [27] and Flierl and Girod [28] recently give an extensive survey on joint multiview video coding algorithms.

2.5.2 Distributed Multiview Video Coding

Distributed video coding is the application of multiterminal source coding theory on video data. Distributed video coding exploits the joint statistics among video streams at the decoder. Based on the number of streams processed by the distributed video codec, distributed video coding can be grouped into two classes: distributed monoview video coding

and distributed multiview video coding. Practical distributed video coding algorithms have been pioneered by Witsenhausen and Wyner [96] and recently revived by A. Aaron et al.'s work [97] and Puri et. al's work [98]. Girod et al. [99] summarized recent advances on distributed monoview video coding. Distributed multiview video codec [100, 101, 102, 103, 104, 105, 106] have been recently proposed by several researchers.

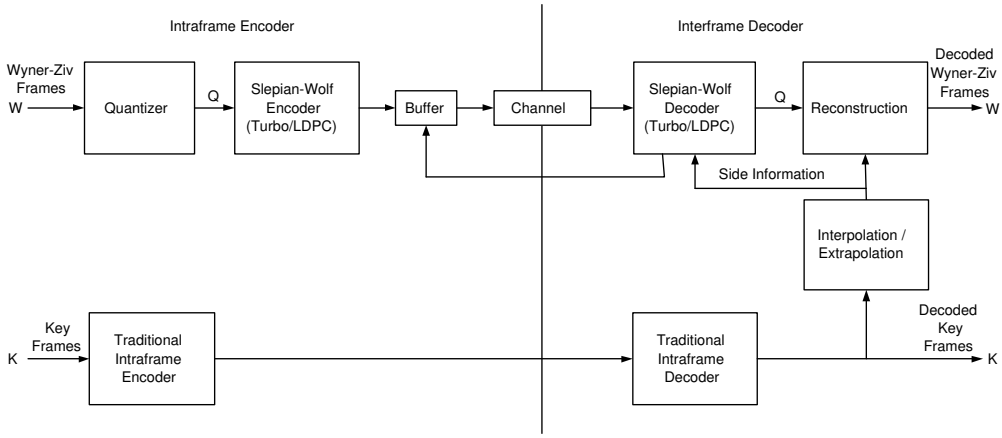
Challenges of Distributed Video Coding

In general, there are two main challenges for a practical distributed video codec to tackle : pixel mismatch and inaccurate correlation model. Pixel mismatch means that there is no prior knowledge at the encoder on the correspondence between two correlated pixels and the decoder likely make inaccurate estimations on matched pixels. The challenge is how to find efficient and effective ways to accurately estimate the pixel correspondence between correlated frames. An inaccurate correlation model means that there is no prior knowledge on how two corresponding pixels are correlated even if two pixels are perfectly matched. The challenge is how to estimate an accurate correlation model between the matched pixels and design a capacity-approaching channel code to approach the correlation model. Previous approaches estimate the correlation model through joint offline and online training. Pixel correspondence is estimated through uncoded neighbor frames. The technique to estimate pixel correspondence normally needs at least two uncoded neighbor frames.

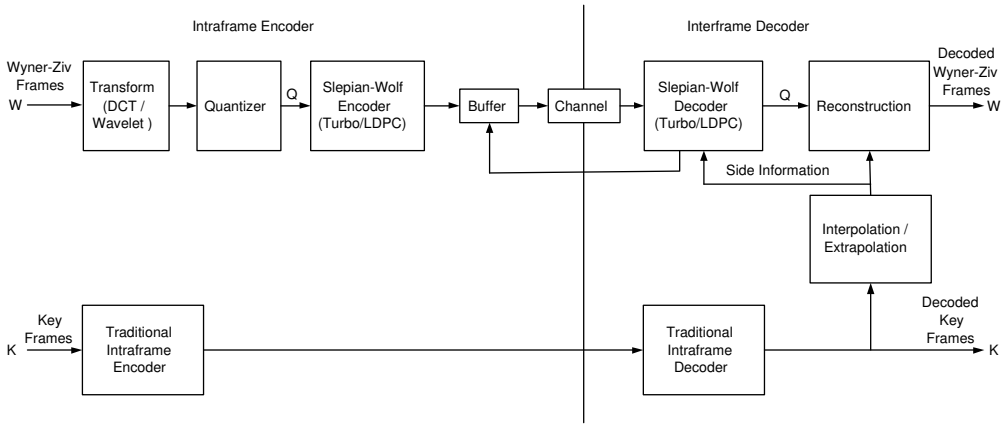
Distributed Monoview Video Codec

Wyner-Ziv principles are first applied in distributed monoview video coding. Previous distributed monoview video coders are mostly designed by using an asymmetric distributed source code. Namely, some video frames are used to estimate side information and others are quantized and coded using Slepian-Wolf coding techniques. More explicitly, video frames are grouped into two classes, Wyner-Ziv frames (WZF) and key frames (KF). Key frames are encoded by using conventional joint video coding techniques and act as side information at the decoder. Wyner-Ziv frames are coded by using a Wyner-Ziv coder. The coding scheme is also referred to as "Wyner-Ziv monoview video codec".

There are two types of Wyner-Ziv video codecs using asymmetric distributed source codes: pixel-domain Wyner-Ziv video codec and transform-domain Wyner-Ziv video codec. Fig. 2.10(a) shows the architecture of the pixel-domain Wyner-Ziv video codec and Fig. 2.10(b) shows the architecture of the transform-domain Wyner-Ziv video codec. The main conceptual difference between the pixel-domain Wyner-Ziv codec and the transform domain



(a) Pixel-Domain Wyner-Ziv Video Codec



(b) Transform-Domain Wyner-Ziv Video Codec

Figure 2.10: Distributed Monoview Video Coding Architectures

Wyner-Ziv codec is that Wyner-Ziv frames in the pixel-domain codec are directly scalar quantized and then coded using a Slepian-Wolf codec while Wyner-Ziv frames in the transform-domain codec need to be first transformed by Discrete Cosine Transform (DCT) or wavelet transform before scalar quantization and Slepian-Wolf coding. Therefore, the transform-domain Wyner-Ziv codec generally is a little more complicated than the pixel-domain one. However, the transform-domain Wyner-Ziv video coder normally has better rate-distortion performance than the pixel-domain one due to the decorrelation function of the transforms. Compared with the transform-domain Wyner-Ziv codec, the pixel-domain Wyner-Ziv codec is easier to implement since an encoder need not worry about the bit-allocation problem which is created by DCT or wavelet transform.

Pixel mismatch problem in a distributed monoview video coder is essentially how to accurately estimate the side information of Wyner-Ziv frames. The Wyner-Ziv monoview video codec uses motion search between neighbor key frames to find matched pixels and generate the side frames through interpolation. The pixels in the estimated side frames are assumed to have an exact correspondence based on the pixel coordinates. The correlation model is assumed to be Gaussian or Laplacian. Parameters of Gaussian or Laplacian models are estimated based on decoded frames.

Distributed Multiview Video Codec

The aim of a distributed multiview video codec is to achieve the joint entropy of multiview video by removing the temporal and inter-view redundancies among multiple video streams and, at the same time, to have a better tradeoff between compression efficiency and computational complexity. Distributed multiview video codec assumes no communication between cameras. Wyner-Ziv principles are first used to compress multiview images. Zhu et al. [107] put forward a distributed multiview image compression algorithm based on the Wyner-Ziv coding. Distributed multiview video codecs have the similar structure as distributed monoview video codecs. Due to the better rate distortion performance, previous distributed multiview video codecs are all transform domain Wyner-Ziv codecs. Fig. 2.11 illustrates a general structure of transform domain Wyner-Ziv multiview video codec.

Like distributed monoview video codecs, the design philosophy of distributed multiview video codecs initially puts emphasis on reducing the computational complexity of an encoder as much as possible while at the same time improving the rate-distortion performance as close to JMVC as possible [100, 101, 102, 103]. Slepian-Wolf codes are used to remove both temporal and inter-view redundancies among multiple video streams. Because a de-

coder needs to do motion search and disparity estimation to find corresponding pixels and decode the channel code based on the estimated correlation model, the complexity of the decoder is significantly increased. Recently some researchers focus on designing distributed multiview video codecs with the objective of at least outperforming the rate-distortion performance achieved by MPEG and H.26x standardized monoview video codec [105, 106]. In this case, an encoder uses the standard hybrid coding algorithms to remove temporal redundancies and then adopts distributed source codes to remove inter-view redundancies. The complexity of the encoder and the decoder is higher than its traditional monoview video counterpart. Distributed multiview video codecs use similar approaches to estimate the correlation model as distributed monoview video codecs. Unlike distributed monoview video codecs, distributed multiview video codecs tackle the pixel mismatch problem by using disparity estimation to find correspondent pixels between the frames in adjacent views and create side frames based on matched pixels. Most previous distributed multiview video codecs are based on asymmetric Slepian-Wolf coding scheme. In this thesis, we propose a symmetric distributed multiview video codec that uses symmetric Slepian-Wolf codes as the underlying entropy coding scheme in Chapter 7.

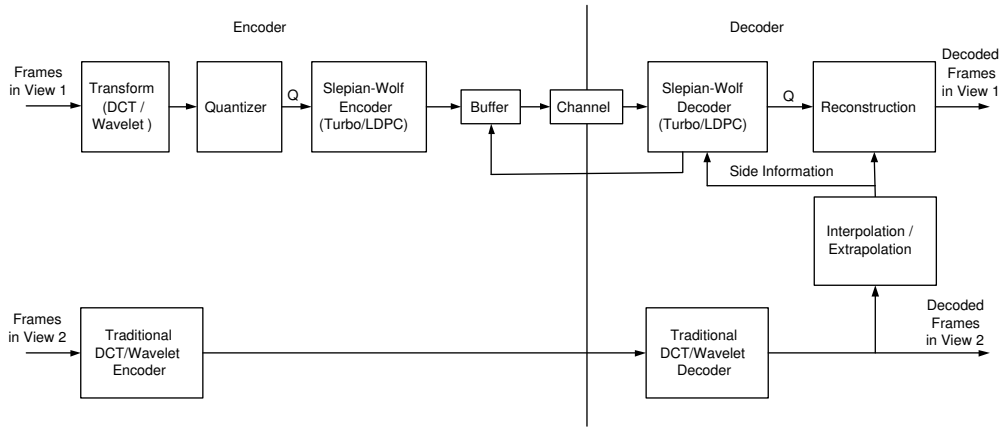


Figure 2.11: Distributed Multiview Video Coding Architecture

2.5.3 Performance Metric

The objective of multiview video coding is to achieve a good trade-off between compression performance and computational complexity. Similar to the traditional mono-view video coding, the compression performance is normally evaluated by using a rate distortion curve. Rate measures how many bits are needed to encode the multiview video and is typically represented by bit per second (bps) or bit per pixel (bpp). Distortion measures the difference

between the decoded image and the original image. Peak Signal to Noise Ratio (PSNR) is normally used to evaluated the distortion. PSNR is defined by the following formula.

$$E_{MSE} = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}|I(i,j)-\hat{I}(i,j)|^2 \quad (2.19)$$

$$PSNR = 20\cdot log_{10}\left(\frac{255}{E_{MSE}}\right) \quad (2.20)$$

where E_{MSE} is the mean square error between an original $m \times n$ image I and its decoded image $\hat{I}$.

2.6 Summary

In this chapter, we introduce some background knowledge on source coding and channel coding theory. We also review previous work in joint multiview video coding and distributed multiview video coding. Joint multiview video coding is based on joint source coding theory. Joint multiview video coding achieves compression by exploiting the temporal and inter-view correlation at encoders. Joint multiview video coding schemes are classified into two categories: block-based approaches and nonblock-based approaches. Distributed multiview video coding is based on distributed source coding theory. Distributed multiview video coding realizes compression by removing temporal redundancies at encoders and remove inter-view redundancies at decoders. Distributed multiview video coding needs to address two special challenges: pixel mismatch and inaccurate correlation model.

Chapter 3

Neighbor-based Multiview Video Compression

3.1 Introduction

As we discussed in Section 1.2.5, multiview video can be created by three approaches: rendering without geometry, rendering with implicit geometry and rendering with explicit geometry. Rendering without geometry and with implicit geometry [1, 3, 4, 5, 17, 2, 21, 108] use densely arranged cameras to acquire high-resolution light fields and generate images at the new viewpoints from captured scene images. Rendering with explicit geometry [109, 110] acquires multiview video from sparsely-arranged cameras and extracts 3D geometry models from the images , and then renders the new viewpoints with the help of a 3D scene model. Rendering with explicit geometry has the advantage of reducing the acquisition cost by using fewer cameras. However, it increases the algorithmic and computational complexity. Namely, it is still a very difficult problem to extract accurate scene models from general real-world scenes in real time. Though rendering without geometry or with implicit geometry demands more storage and transmission bandwidth due to the large amount of video data, we still believe that rendering without geometry or with implicit geometry will be the solution to multiview video in the near future. In this chapter, we propose an efficient joint multiview video compression scheme to reduce the amount of video data used for image-based rendering. Images are captured over a real-world dynamic scene by multiple synchronized cameras and sent to a server for compression. Fig. 3.1 illustrates the architecture.

Multiview video compression aims at achieving high compression efficiency by simultaneously reducing both temporal and inter-view correlation among multiple synchronized video streams. The challenge is how to design efficient algorithms to remove the inter-

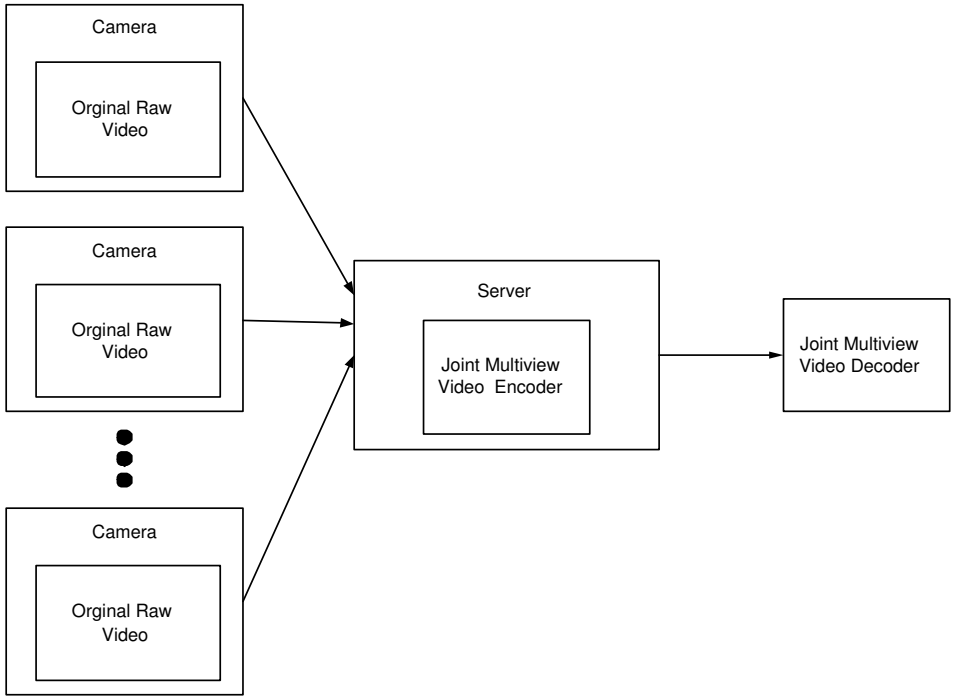


Figure 3.1: Multiview Video Acquisition and Compression

view redundancies. As we discussed in Section 2.5.1, there are two general approaches: block-based approach and nonblock-based approach. Block-based approaches are popular approaches and have been adopted in traditional video coding standards such as MPEG2 and H.264. Blocks are used as basic units to estimate motion and disparity in neighbor frames and remove temporal and inter-view redundancies. MPEG2 Multiview Profile (MVP) [72] uses a block-based stereoscopic coding (BBSC) to compress the stereo video. Motion-compensated prediction (MCP) is adopted to remove the temporal correlation and disparity-compensated prediction (DCP) is adopted to remove the inter-view correlation. The left view is first compressed by a MCP-based monoview coding algorithm and then the right view is coded by using both DCP from the left view and MCP. [73, 74] extends MVP to compress multiview video sequences. Multiple video streams are classified into two types of streams: main stream and secondary stream. The main stream is the central stream among all the streams and is encoded using only a MCP-based MPEG2-like algorithm. Secondary streams are compressed with MCP and DCP based on the main stream as illustrated in Fig. 3.3(a). We call this approach the *center approach*. In the center approach, each stream can have at most one stream as the reference stream used for removing inter-view correlation. In this chapter, we propose a neighbor-based joint multiview compression scheme. We call our approach the *neighbor approach*. It is published in [111].

The neighbor approach is a MPEG2-like block-based method and allows each stream to have multiple streams as reference streams for inter-view redundancy reduction. Since our publication, many block-based schemes integrated with H.264/AVC standard have been proposed [75, 76, 77, 78]. The Joint Video Team (JVT) of VCEG and MPEG is now developing a new standard for multiview video coding called the Joint Multiview Video Model (JMVM). JMVM is also a block-based scheme and allows each streams to have multiple reference streams to remove inter-view redundancy. JMVM uses a configuration file to manually specify reference relationships between different views and the corresponding stream encoding order. Unlike JMVM, an automatic algorithm is used to decide the reference relationships between multiple views and the stream encoding order in the thesis. The algorithm can be integrated into the future version of JMVM to improve its performance.

In the neighbor approach, multiple video streams are also classified into two groups: main stream and secondary stream. One stream is chosen as the main stream and is coded using only a MCP-based MPEG2-like algorithm. Unlike the center approach, the secondary stream is compressed using DCP from multiple nearest neighbor streams and MCP. Our approach can have better inter-view redundancy reduction than the center approach and thus improve the video quality under the same bit rate. In addition, a novel algorithm is proposed to automatically decide the optimal stream encoding order and help each stream find the best reference streams to remove inter-view correlation.

The rest of the chapter is organized as follows. Section 3.2 discusses in detail the proposed multiview video compression scheme. In Section 3.3, we present the experimental results. Conclusion is drawn in Section 3.4.

3.2 Details of The Neighbor Approach

The proposed neighbor-based multiview video compression is a MPEG2-like block codec. Namely, a macroblock is used as a basic unit for motion and disparity estimation and prediction. Similar to the center approach, each stream is one of two types: main stream and secondary stream. There is only one main stream. The pictures are also classified into three types: I, P, and B-frame. I, P, and B pictures in the main stream are encoded with a MPEG-2 like algorithm. However, pictures in the secondary streams are encoded with a different method. I pictures in the secondary streams are encoded with DCP based on the I pictures in neighbor streams. P and B pictures are encoded with MPEG2-like MCP and DCP of corresponding P and B pictures in neighbor streams. The scheme is illustrated

in Fig. 3.3(b). Each stream uses the same group of picture (GOP) structure as MPEG2. All the synchronized GOPs of a multiview video form a group of GOP (GGOP) similar to the center approach. In this chapter, we assume that each camera position and its viewing direction are known.

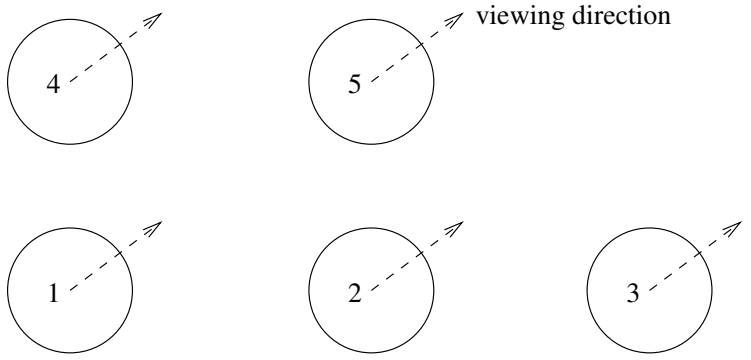
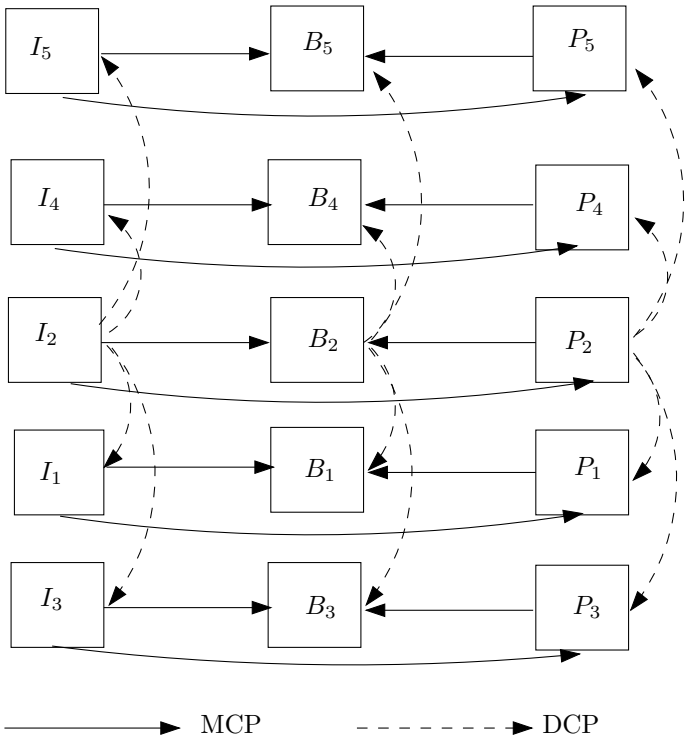


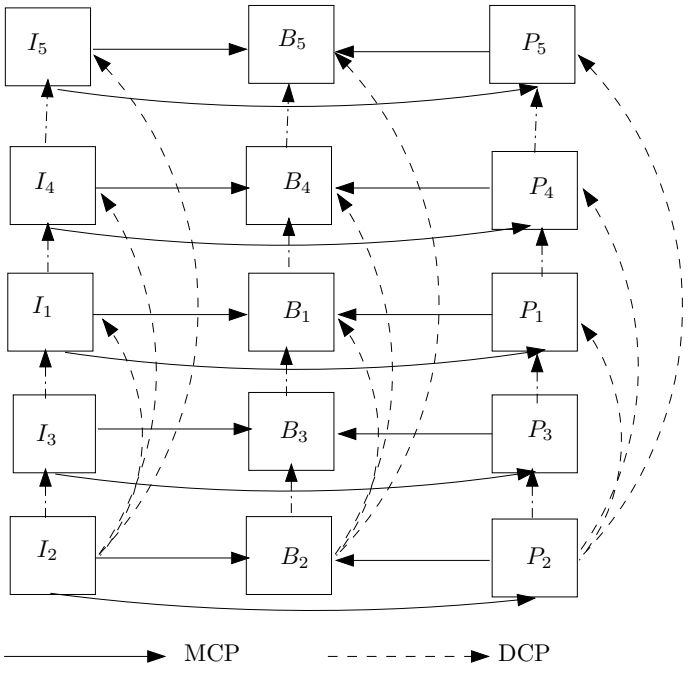
Figure 3.2: Camera Setup. 5 cameras are placed in two parallel lines and spaced uniformly in both horizontal and vertical direction. Cameras have the same viewing direction. Namely they are parallel cameras.

The basic idea of the neighbor approach is to use multiple nearest neighbor frames as reference frames for disparity-compensated prediction. However, the synchronized frames in multiview video are coded in a specific order and the early-coded frames may not find optimal nearest neighbor frames as reference frames. For example, in Fig. 3.2, suppose that a frame can have at most two neighbor frames as reference frames for DCP. The stream encoding order is $S_2S_3S_1S_4S_5$, where S_i denotes stream i. The frames in S_4 can only have frames in stream S_1 and S_2 as reference frames though obviously its two nearest neighbor streams are S_1 and S_5. The key challenge of the neighbor approach is to decide the optimal stream encoding order so that the number of streams which have the optimal neighbor streams as reference streams can be maximized and thus the maximal decorrelation of inter-view coherence can be achieved.

Our basic idea to solve the problem is to first construct a weighted undirected graph based on the geometric relation between cameras and then the optimal stream encoding order problem is equivalent to find an optimal graph traversal order subject to an optimal criteria. In the weighted graph, a node denotes a camera and an edge weight denotes the Euclidean distance between two cameras or angle between the view directions of two cameras. The smaller the weight between two nodes, the larger the overlapping region between the images captured by two cameras. Formally, given a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges in the graph $\mathcal{G}$, let $w(v_i, v_j)$ denote the edge



(a) Center Approach



(b) Neighbor Approach. Each frame can have maximal two neighbor frames as reference frames.

Figure 3.3: Illustration of the Center Approach and the Neighbor Approach

weight between two nodes v_i and v_j. Obviously, $w(v_i, v_j) \geq 0$. Suppose that there are n nodes in the graph. For a specific traversal order, $v_{i_1}, v_{i_2}, \ldots, v_{i_n}$, let a node v_{i_k} have at most m nodes as neighbors in the traversed node set $\{v_{i_1}, v_{i_2}, \ldots, v_{i_{k-1}}\}$. Define the neighbor distance of the node as $d_{i_k} = \frac{1}{m_{i_k}} \sum_{j=1}^{m_{i_k}} w(v_{i_k}, v_{i_j})$, where m_{i_k} is the actual number of neighbors in the traversed node set and $1 \leq m_{i_k} \leq m$, $v_{i_j}, i_1 \leq i_j \leq i_{k-1}$ is a neighbor of the node v_{i_k}. An optimal graph traversal order is defined as an order that minimizes sum of the neighbor distance of all nodes in the order. Namely, an optimal graph traversal order minimizes $\delta = \sum_{k=i_1}^{i_n} d_{i_k}$. We use Algorithm 2 to decide the optimal streaming order.

The algorithm is a greedy algorithm. To find an optimal traversal order of a graph, the algorithm needs to decide which node is first traversed. The first traversed node represents the main stream. Other nodes represent secondary streams. Since there is no prior knowledge on which node should be first traversed, the algorithm iteratively lets each node be the first traversed node and uses a greedy algorithm to make sure that the traverse order starting with any node is an optimal traversal order in all the possible orders starting with the node. After choosing the first node, the algorithm iteratively chooses the node with the smallest neighbor distance to added the traversed node net. After an optimal traversal order starting with every node is found, the one with the minimal total neighbor distance is chosen as the final optimal traversal order. The algorithm can easily be proved to output an optimal streaming encoding order by induction. Theorem 3.1 states the optimality of the algorithm.

Theorem 3.1. *Algorithm 2 finds an optimal graph traversal order that minimize the sum of neighbor distance among all possible traversal orders.*

Proof. We first prove by induction that for a specific starting node, the algorithm gives an optimal traversal order among all possible traversal orders starting with the node. Namely, the code between Line 16 and Line 28 finds a traversal order starting with node v_i that has minimal δ_i.

The induction hypothesis is that δ_i is minimal in any iteration of the loop. The hypothesis is obviously true at the beginning of the loop since initially $\delta_i = 0$. Assume that the hypothesis is true at the j-th iteration. At the beginning of the $j+1$-th iteration, a node v_p with neighbor distance d_p is added into the coded node set. Since v_p is the node with the minimum neighbor distance among all candidate nodes in the set cur, $0 \leq d_p \leq d_k, v_k \in cur$. For nodes are not in the set cur and $coded$, there is no adjacent edge between them with nodes in traversed node set $coded$. Namely, $d_u = \infty, v_u \notin cur \cup coded$. Thus $d_p < d_u$.

Algorithm 2 Pseudo Code to Decide The Stream Encoding Order

n : the total number of nodes;
m : the maximal number of allowed neighbors for each node;
m_j : the actual number of neighbors for node v_j;
v_i : node i;
$sall$: the set of all the n nodes;
nbr_i : the set of neighbors of node i;
d_j : the neighbor distance for node v_j;
δ_i : the total neighbor distance of a traversal order starting with node v_i;
δ : the minimal total neighbor distance of a traversal order;
$coded$: the set of traversed nodes;
cur : the set of candidate nodes to be traversed;
i, j, k p : integer variables;

form a graph based on the geometric relation between cameras
$\delta = \infty$;
for i = 1 to n **do**
  $\delta_i = 0$;
  add node v_i to the traversed node set $coded$
  $coded = coded \cup v_i$;
  $cur = cur \cup nbr_i - coded$;
  while $coded \neq sall$ **do**
    for each node v_j in cur **do**
      calculate $d_j = \frac{1}{m_j} \sum_{k=1}^{m_j} w(v_j, v_k), v_k \in coded$;
    end for
    choose the node v_p with the minimum d_p as the next traversed node;
    $coded = coded \cup v_p$;
    $cur = cur \cup nbr_p - coded$;
    $\delta_i = \delta_i + d_p$;
  end while
  if $\delta_i < \delta$ **then**
    $\delta = \delta_i$;
    remember the traversal order start with node v_i;
  end if
end for
output the δ and the corresponding optimal stream encoding order;

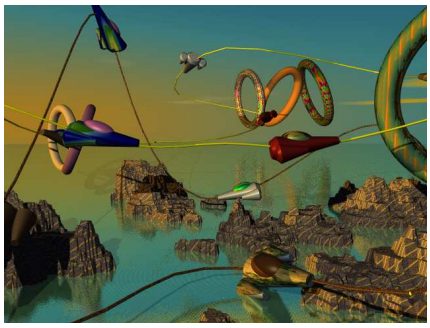
Therefore, $\delta_i = \delta_i + d_p$ is minimal at the end of the $j+1$-th iteration. The hypothesis is always true. The algorithm can find a traversal order with minimal total neighbor distance starting with a node v_i. Since the algorithm outputs a traversal order that has the minimal total neighbor distance among all the minimal traversal orders starting with any node in the graph, the final output traversal order minimizes the sum of neighbor distance among all possible traversal orders. □

After deciding the stream encoding order, frames in the main stream and secondary streams can be predicted based on MCP and DCP. Thereafter, like MPEG2, discrete cosine transform (DCT) is used to transform the prediction error and the coefficients are then quantized and entropy coded using Huffman or arithmetic coding. The algorithm needs $2n + k$ buffers to do motion and disparity prediction, where $m \leq k \leq n$ and n is the number of video streams and m is the maximum number of neighbor streams.

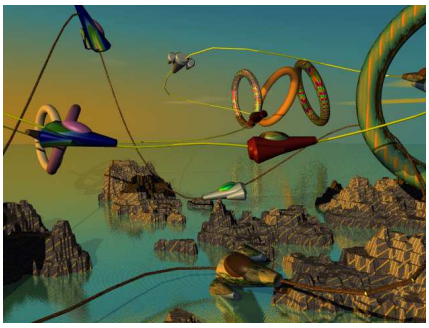
3.3 Experimental Results

Both synthetic video and real video are used to evaluate the rate-distortion performance of proposed multiview compression scheme. Synthetic video is rendered using POVRAY [112]. Cameras are placed in parallel lines and spaced uniformly in both the horizontal and vertical direction. This is a parallel camera setup and the viewing direction is the same for every camera. Fig. 3.4 shows part of the synthesized multiview video used in this research. Synthetic video has 14 streams with 7 camera per line. Each stream is a $24-bit$ RGB video with 640×480 pixels and 25 frames per second. "Breakdance" and "Ballet" sequences [3], which are wildly used as benchmark test video for multiview video compression, are adopted as examples of real-world video. Fig. 3.5 and Fig. 3.6 shows part of Breakdance and Ballet sequence. Both sequences have 8 views. Each stream is a $24 - bit$ RGB video with 1024×768 pixels.

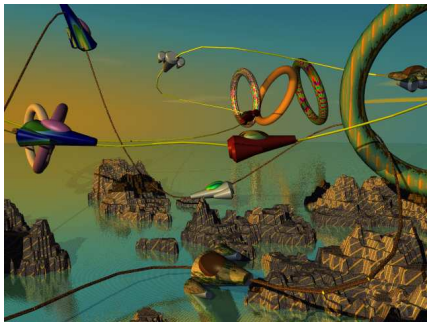
We first study the compression performance of the neighbor approach, center approach, and MPEG2. Fig. 3.7(a), Fig. 3.8(a) and Fig. 3.9(a) present the rate-distortion comparison for Synthesis, Breakdance and Ballet video sequences. They show that the neighbor approach typically has a better rate-distortion performance than both the center approach and MPEG2. This is expected since the neighbor approach uses multiple nearest neighbor frames as reference frames and can better exploit the inter-view coherence than the center approach which uses only one reference frame for inter-view prediction and MPEG2 that uses only temporal prediction and no inter-view prediction. However, the neighbor



(a)



(b)



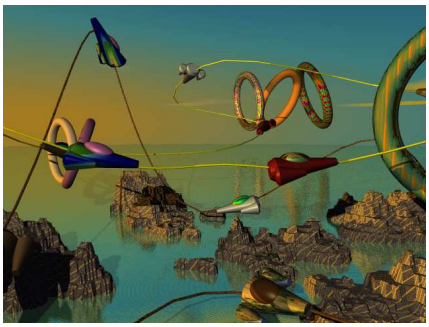
(c)

(d)

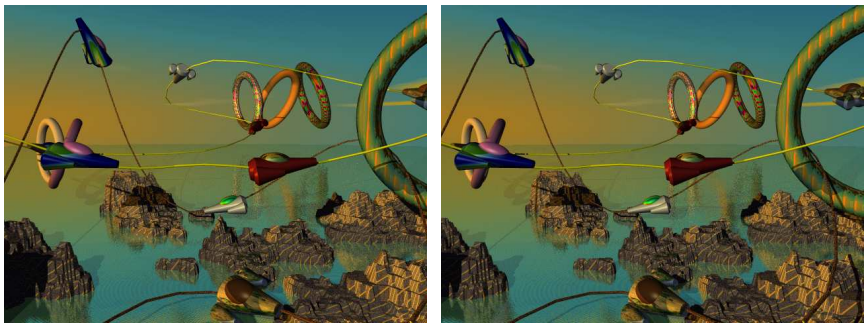
(e)

(f)

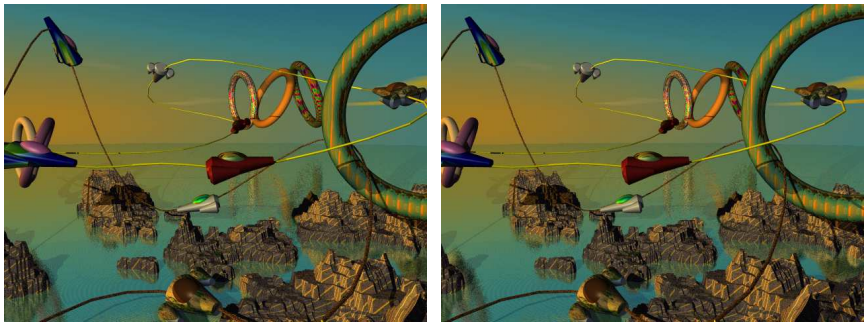
(g)



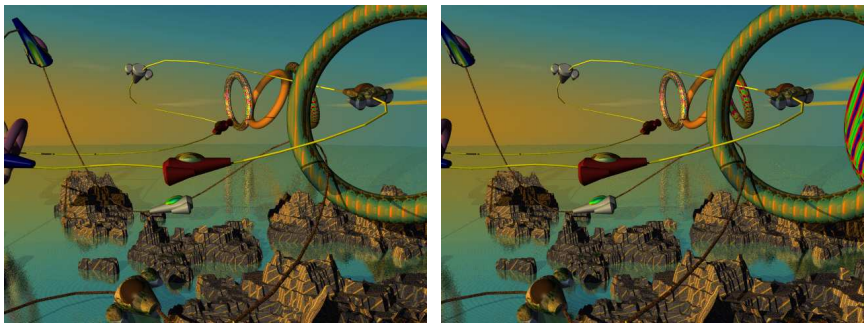
(h)



(i) (j)

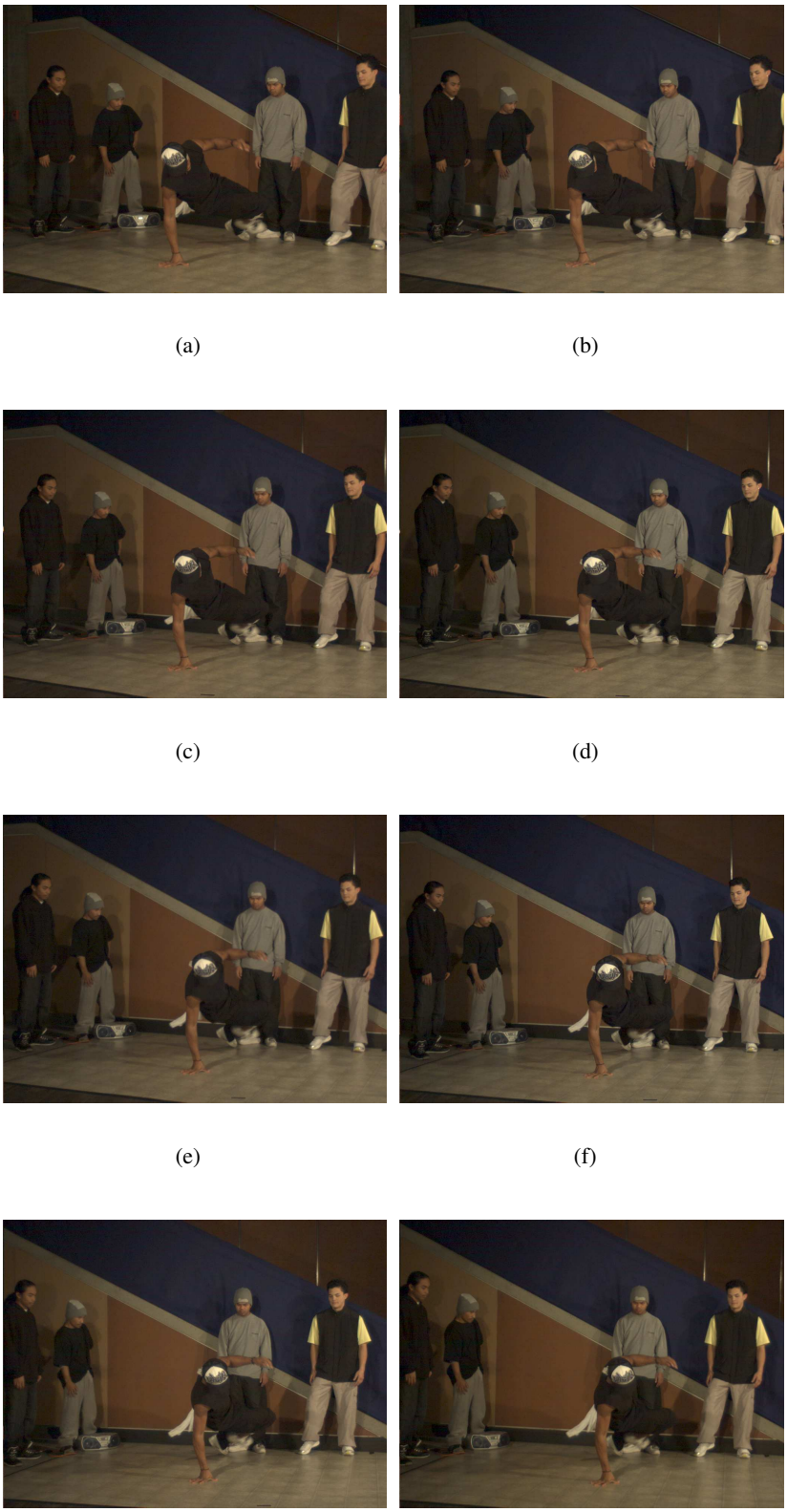


(k) (l)



(m) (n)

Figure 3.4: Synthetic Multiview Video



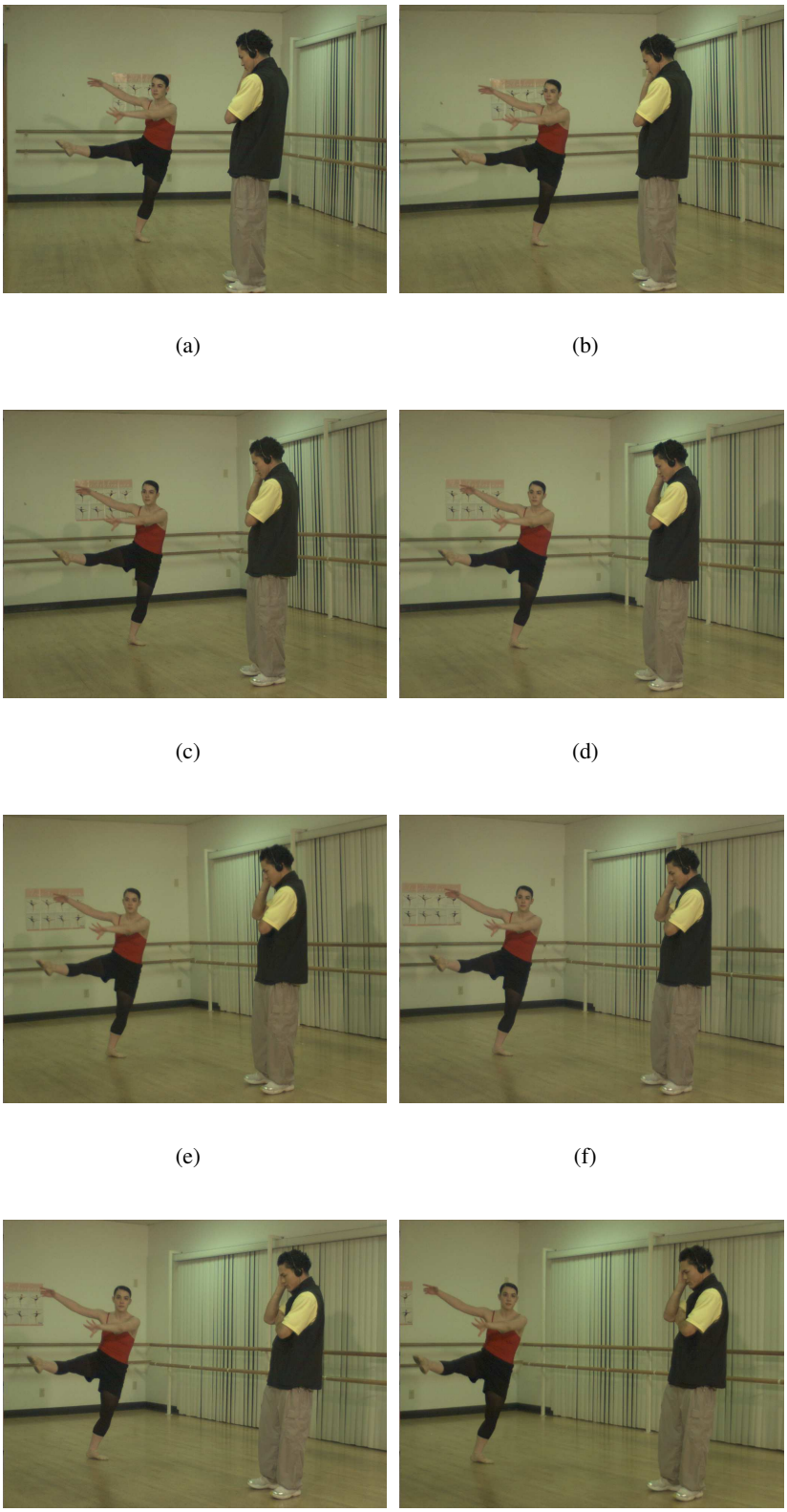
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 3.5: Breakdance Video



(a) (b)

(c) (d)

(e) (f)

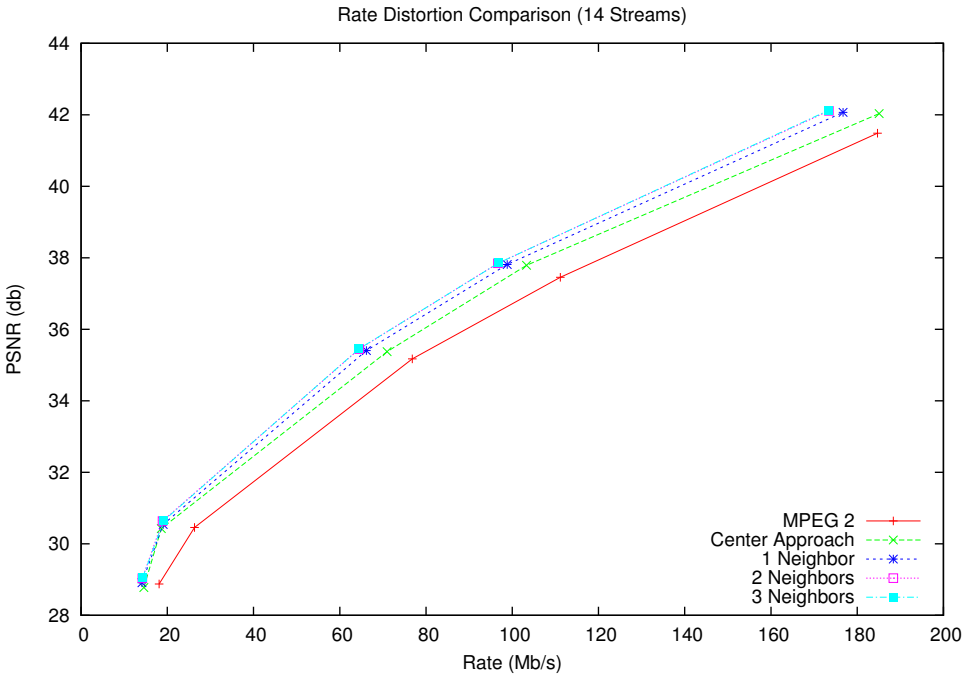
(g) (h)

Figure 3.6: Ballet Video

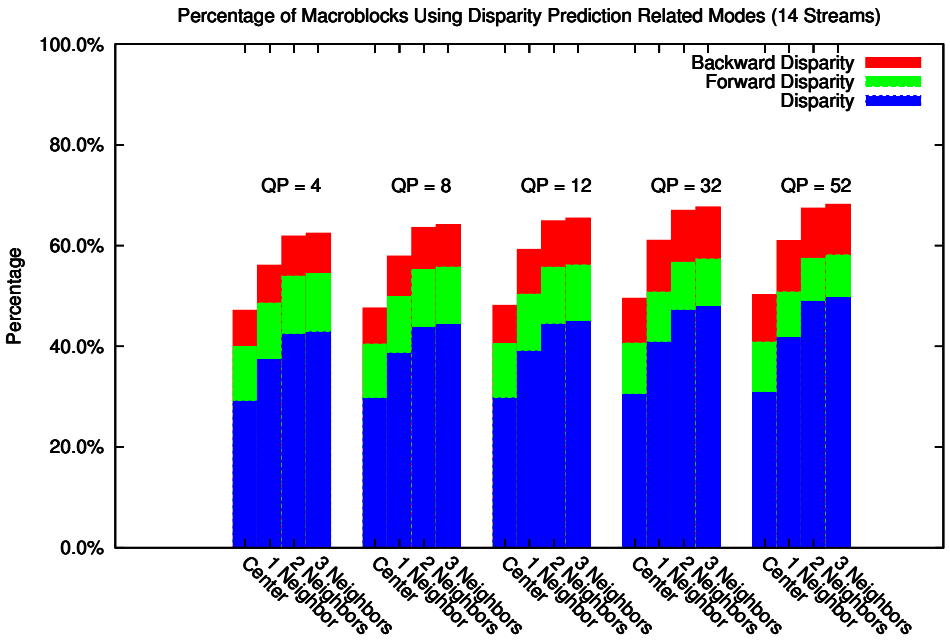
approach cannot guarantee to have a better rate-distortion performance than the center approach and MPEG2. Fig. 3.9(a) indicates that the neighbor approach has an inferior rate-distortion performance than the center approach and MPEG2 at high rate. The reason is that macroblocks predicted by DCP need spend more overhead bits to encode neighbor index and disparity vector. When DCP is only slightly better than MCP, the reduction on bits to encode coefficients might be smaller than the increase in overhead bits.

We also study the number of neighbors on the performance of different multiview coding schemes. Fig. 3.7(a), Fig. 3.8(a) and Fig. 3.9(a) show that as the number of the neighbor frames increases, the rate-distortion performance also gets better since more neighbor frames lead to more inter-view redundancy reduction. However, PSNR improvement is limited as the number of neighbor frames increases. This is understandable since only a limited number of macroblocks will be predicted by new added neighbor frames while most other macroblocks already have a small prediction error and it is hard to further diminish the prediction error with the added neighbor frames. The computational complexity increases linearly as the number of neighbors increases since similar macroblocks need to be searched in every reference neighbor frame.

The compression performance of multiview view coding scheme is greatly dependent on how many macroblocks in secondary streams are predicted by DCP. Each macroblock in the proposed compression scheme can be predicted using one of 6 modes: Forward (F), Backward (B), Disparity (D), Forward and Backward interpolation (FB), Forward and Disparity Interpolation (FD), Backward and Disparity Interpolation (BD). We quantify the effect of DCP on multiview video compression by studying the percentage of macroblocks predicted by disparity-related prediction modes, D, FD and BD. Fig. 3.7(b), Fig. 3.8(b) and Fig. 3.9(b) give the average percentage of macroblocks using disparity prediction related modes in both the center and the neighbor approach for Synthetic, Breakdance and Ballet video sequence for different quantization parameters (QP). They indicate that disparity-related prediction mode accounts for a significant portion of prediction types and plays a significant role to reduce prediction error and improve the image quality. The results also show that the neighbor approach has a higher percentage of macroblocks disparity prediction related modes than the center approach. Thus the better rate-distortion performance in the neighbor approach should obviously be attributed to the inter-view redundancy reduction. From the graphs, it is also evident that the benefits from increasing the number of neighbor frames is limited and the gain becomes smaller and smaller as the the number of neighbor frames continually increases since the percentage of macroblocks using disparity

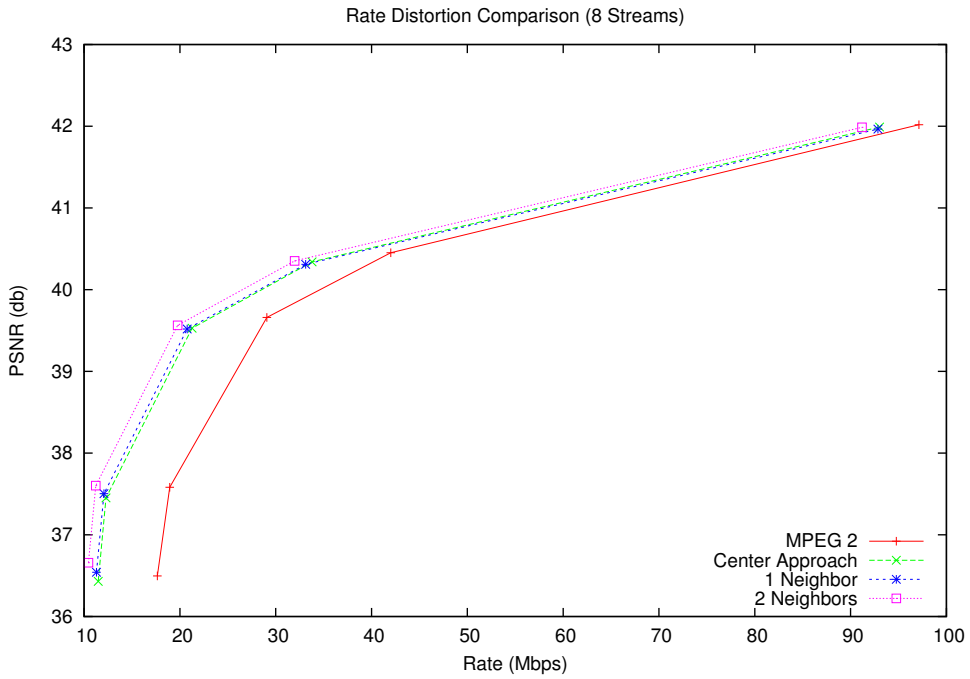


(a) Rate Distortion Comparison

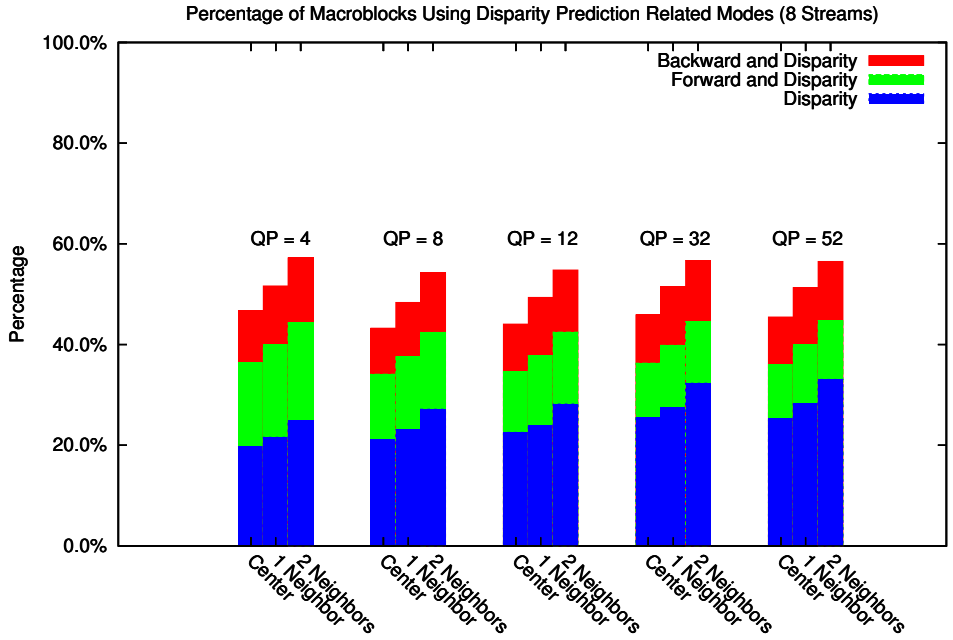


(b) Percentage of Macroblocks Using Disparity Prediction Related Modes

Figure 3.7: Rate Distortion Comparison and Percentage of Macroblocks Using Disparity Prediction Related Modes for Synthetic Video

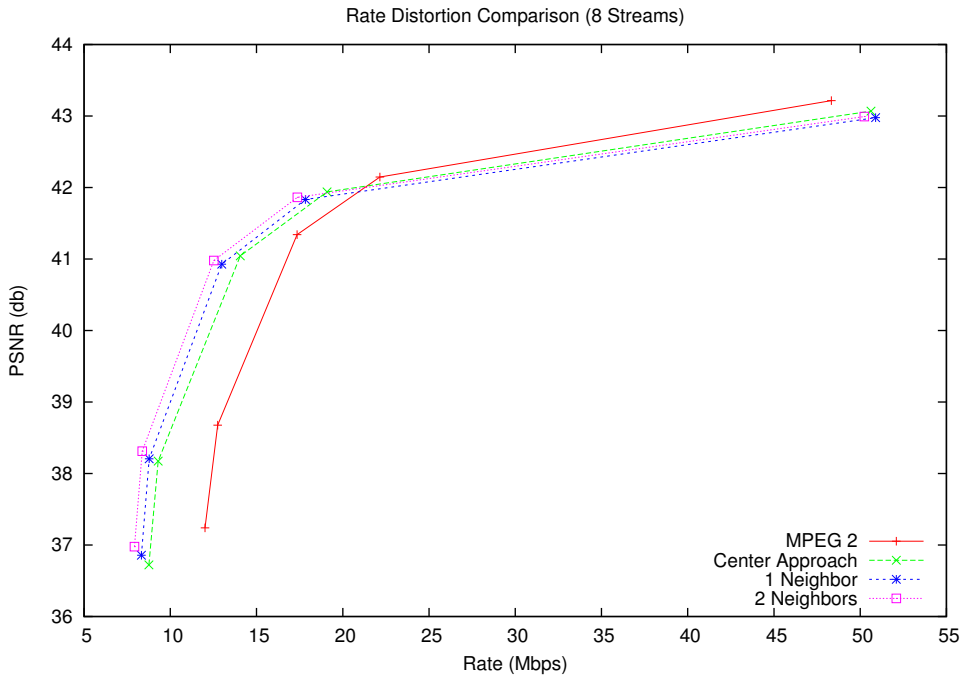


(a) Rate Distortion Comparison

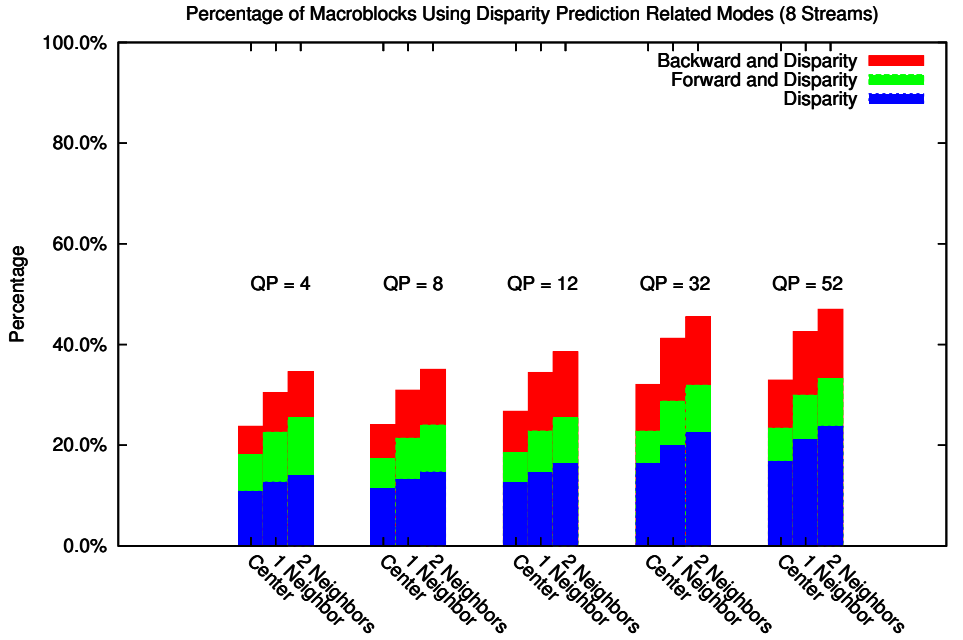


(b) Percentage of Macroblocks Using Disparity Prediction Related Modes

Figure 3.8: Rate Distortion Comparison and Percentage of Macroblocks Using Disparity Prediction Related Modes for Breakdance Video

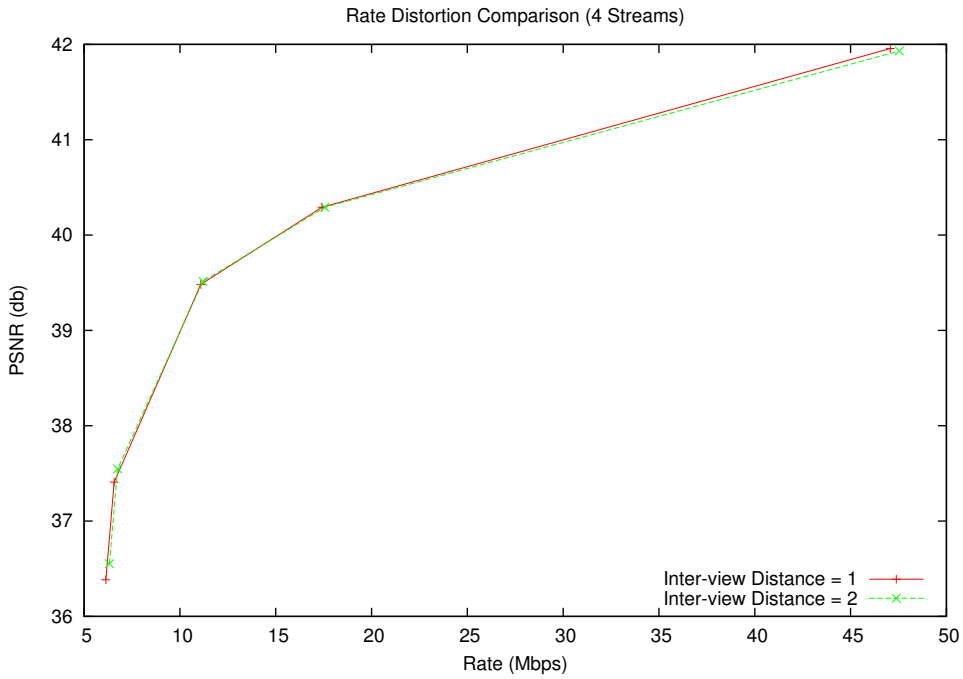


(a) Rate Distortion Comparison

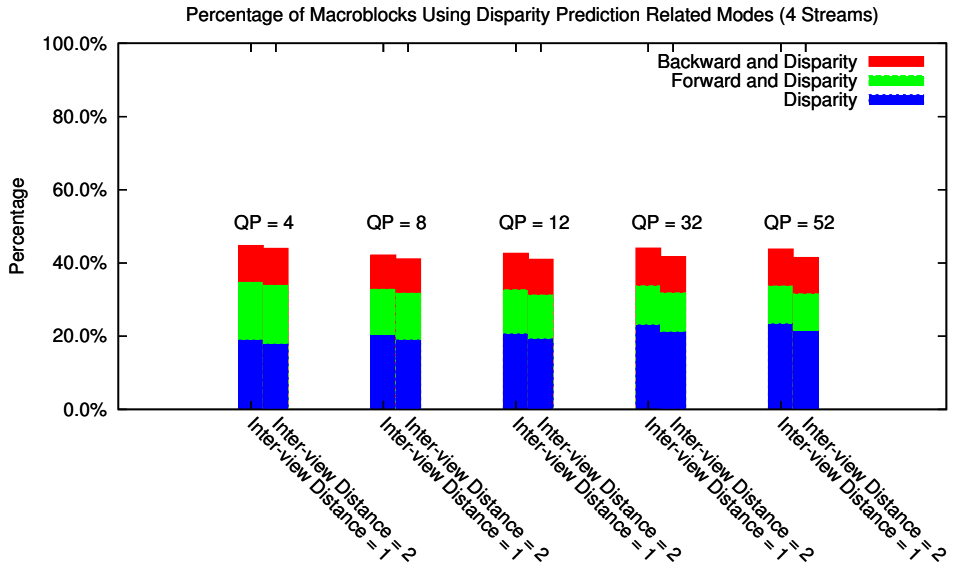


(b) Percentage of Macroblocks Using Disparity Prediction Related Modes

Figure 3.9: Rate Distortion Comparison and Percentage of Macroblocks Using Disparity Prediction Related Modes for Ballet Video

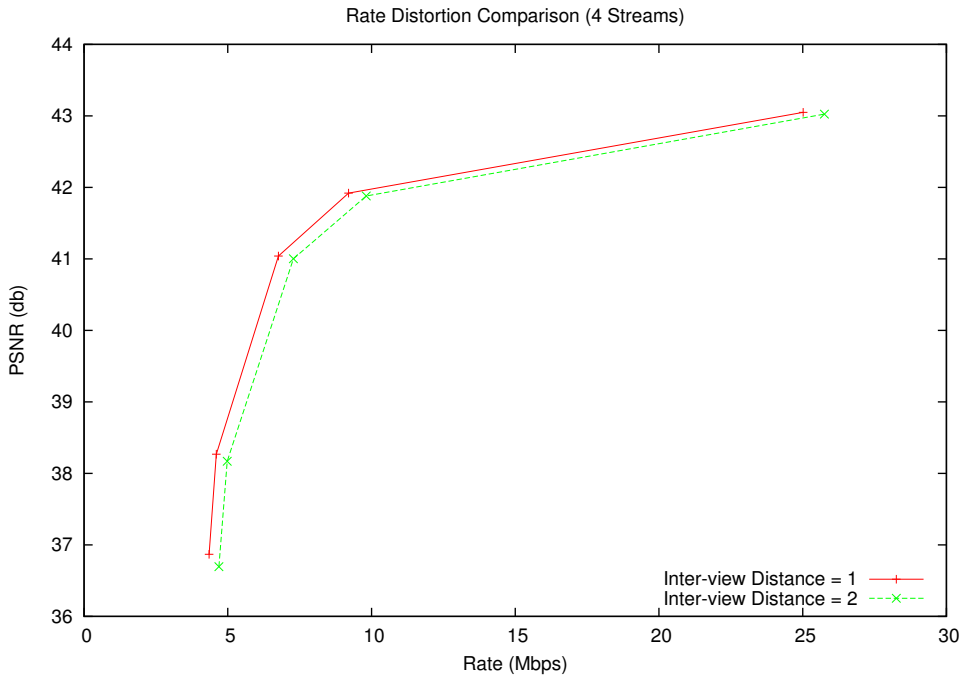


(a) Rate Distortion Comparison

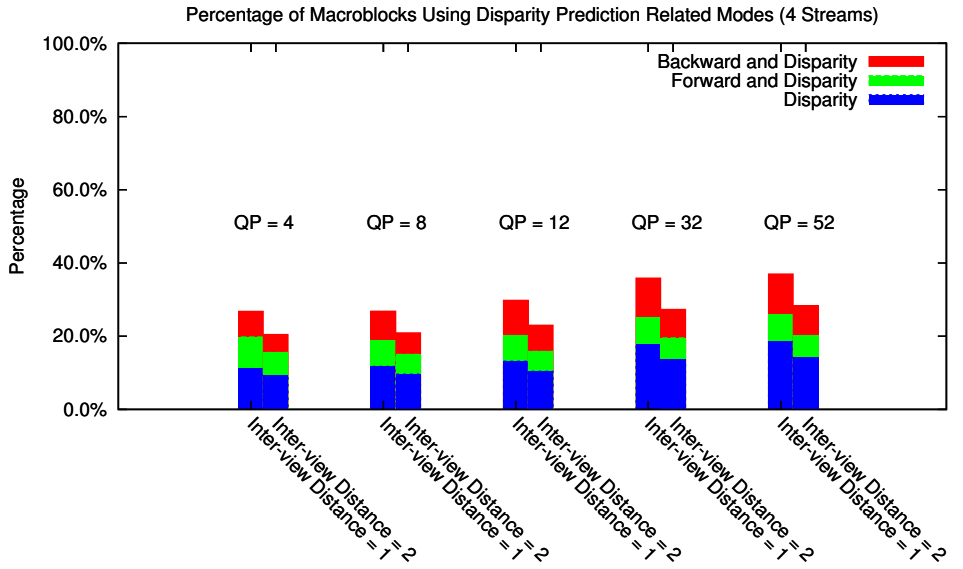


(b) Percentage of Macroblocks Using Disparity Prediction Related Modes

Figure 3.10: Rate Distortion Comparison and Percentage of Macroblocks Using Disparity Prediction Related Modes for Different Inter-view Distance, Breakdance Video, 4 Cameras, 1 Neighbor



(a) Rate Distortion Comparison



(b) Percentage of Macroblocks Using Disparity Prediction Related Modes

Figure 3.11: Rate Distortion Comparison and Percentage of Macroblocks Using Disparity Prediction Related Modes for Different Inter-view Distance, Ballet Video, 4 Cameras, 1 Neighbor

prediction related mode only increases slightly as the number of neighbor frames increases.

The rate-distortion performance of the proposed multiview video compression schemes is mainly decided by the inter-view redundancy between images captured by different cameras. Inter-view redundancy is greatly affected by the inter-view distance between different cameras. We study the effect of inter-view distance on rate-distortion performance of the neighbor approach on "Breakdance" and "Ballet" video. 4 cameras are chosen from 8 cameras based on inter-view distance. Fig. 3.10(a) and Fig. 3.11(a) give the rate-distortion performance of the neighbor approach on Breakdance and Ballet video when the number of neighbors is 1. They show that the performance degrades as the inter-view distance increases. This is expected since the large inter-view distance normally leads to small overlapping region between images captured by different cameras. The rate-distortion difference in Breakdance video is less conspicuous than that in Ballet video because the difference on overlapping region in Breakdance video is smaller than that in Ballet video as the inter-view distance increases. Fig. 3.10(b) and Fig. 3.11(b) show the percentage of macroblocks using disparity prediction related modes. The results confirm that large inter-view distance leads to small overlapping region and thus a small portion of macroblocks are predicted by disparity prediction related modes.

3.4 Conclusion

In this chapter, we present a novel neighbor-based multiview video compression scheme. It is a MPEG2-like block-based video compression scheme. In particular, we put forward an efficient algorithm to find an optimal stream encoding order which helps maximize inter-view redundancy reduction. We compare the proposed scheme with the center approach and MPEG2. Experimental results show that the proposed compression scheme can achieve better compression performance over the center approach and MPEG2. The performance of the proposed multiview video coding scheme is greatly dependent on the size of overlapping region between videos captured by different cameras. The large inter-view distance normally leads to small overlapping region and thus degrades the rate-distortion performance. The more the number of neighbor reference views, the better the rate-distortion performance. The improvement on rate-distortion performance is limited as the number of neighbor reference views is greater than a threshold. The computational complexity in the encoder will increase linearly as the number of reference views increases. Both the encoder and the decoder need more memories to store the reference frames as the number

of neighbor reference views increases.

Chapter 4

A Multiview Video Transcoder

4.1 Introduction

One of the challenges for rendering with no geometry or implicit geometry is to capture and transmit dense light field data in a cost-effective way. Most previous prototype systems [2, 108] adopt an acquisition subsystem as illustrated in Fig. 3.1, where each PC controls one or two cameras for synchronized video acquisition and then compresses the captured videos for transmission. Obviously, the acquisition approach is not an economical solution for realtime multiview video applications which usually demand scores or hundreds of cameras [113]. A practical approach for video acquisition should use a specialized light field video camera such as the one proposed in [22], which can control over 100 cameras using one PC. Each video camera directly compresses the video using MPEG2 and then sends the compressed video to the PC for further processing. Although the MPEG2-compressed video can be directly transmitted to receivers for new view synthesis [2], this approach will waste bandwidth since it does not compress the inter-view redundancies among video streams. An economical solution should further remove the inter-view redundancies among multiple compressed video streams before transmission over a network.

In this chapter, we propose a novel multiview video transcoder which efficiently encodes multiple compressed synchronized video streams. This work is published in [114]. Fig. 4.1 shows its architecture. A multiview transcoder can be deemed as a two-stage coding process: separate source coding and joint source coding. The first stage is done at the camera and temporal correlation is removed. The second stage is used to remove inter-view correlation. It is similar to the traditional video transcoder [115, 116] that converts a video from one format to another format. While the traditional video transcoder normally processes one compressed video stream or multiple uncorrelated compressed video streams with little inter-stream redundancies, a multiview video transcoder processes two or more

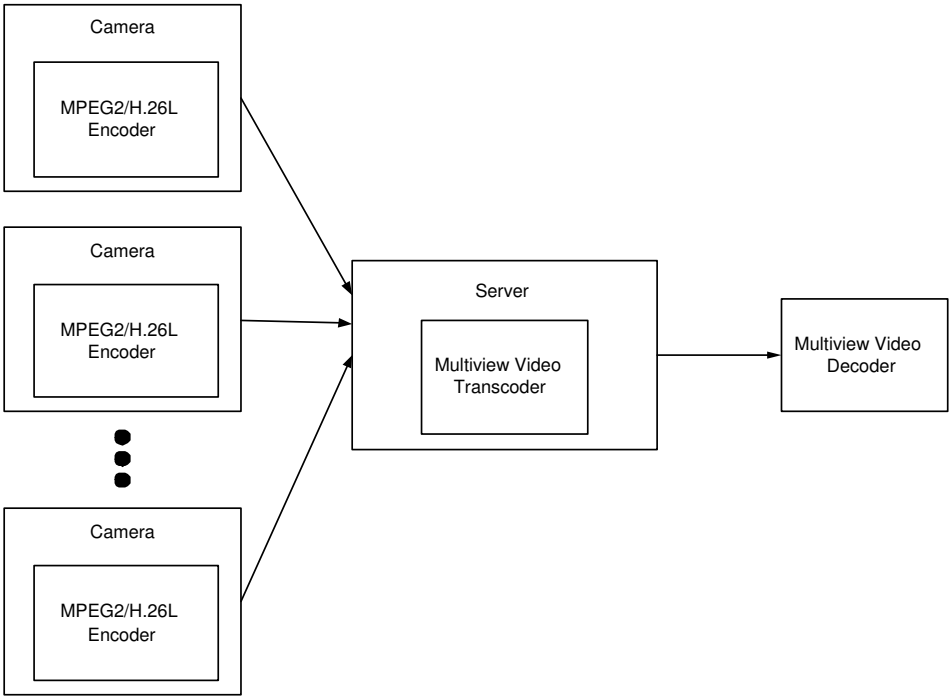


Figure 4.1: Two-stage Multiview Video Acquisition and Compression

compressed synchronized video streams with high inter-view redundancies among video streams. Besides the change of coding parameters such as bit-rate, frame-rate, and spatial resolution of traditional transcoders, a unique task of a multiview video transcoder is to efficiently decorrelate the inter-view redundancies among input video streams. We put forward a fast GOP-based algorithm to effectively limit the search window and efficiently estimate disparity among video streams and thus speed up the inter-view redundancy reduction.

The rest of chapter is organized as follows. We discuss related work in video transcoding in Section 4.2. Section 4.3 elaborates the details of the proposed multiview video transcoder. Experimental results are presented in Section 4.4. The chapter concludes in Section 4.5.

4.2 Video Transcoding

In the past decade, various video transcoding techniques have been invented to convert the coded video from one format to another to accommodate different application scenarios [116, 115]. Each transcoder tries to achieve the highest possible video quality while maintaining lowest computational complexity. These transcoding techniques can be roughly classified into 4 categories: bit-rate transcoding, spatial/temporal transcoding, in-

formation insertion transcoding, and standard transcoding [115].

Bit-rate transcoders aim at reducing the bit rate. Three general transcoding architectures are proposed in the past : open-loop transcoders [117, 118, 119], Cascaded Pixel-Domain Transcoders (CPDT) [118, 120] and DCT-Domain Transcoders (DDT) [121, 122]. Open-loop transcoders are not suitable for practical use since they induces drift problem. In general, DDT demands less computation and memory than CPDT while CPDT can achieve a better video quality than DDT, in particular, when the Group of Pictures (GOP) size is large.

Spatial and temporal transcoders aim at reducing spatial resolution and temporal resolution. The unique challenge in spatial/temporal transcoder is how to efficiently and accurately map the input motion vectors to output motion vectors [123, 124, 125, 126, 127]. A CPDT architecture is preferred for spatial and temporal transcoder since it avoids drift problem and improves the video quality though a DCT-domain architecture [128, 129] can also be used with proper drift compensation mechanism.

Information insertion transcoders are able to add new information into input video streams. Typically, two types of information are inserted : security features such as logo and watermark and error-resilient features. Logo and watermark insertion mainly intend for copyright protection [130, 131]. Error-resilient transcoders aim at achieving robust video transmission over lossy channels such as wireless networks or congested Internet links by inserting error-resilient features [132, 133, 134].

Standard transcoders are used to covert the video coded in one coding standard (MPEG2) to another standard (H.264) [135, 136]. Normally, beside changes in bit-rate and spatial/temporal resolution, standard transcoders need to change other structures such as the frame type.

Although most previous transcoders intend to transcode only one video stream, they are also used to transcode multiple video streams and are called joint transcoding [137, 138, 139]. However, joint transcoders normally assume that there exists no inter-stream redundancies among multiple input video streams. A multiview video transcoder can be deemed as a joint transcoder with the assumption that high inter-stream correlation exists among multiple input video streams.

4.3 Multiview Video Transcoder

4.3.1 Overview

A multiview video transcoder (MVT) converts multiple compressed synchronized video streams into an encoded video stream by changing various coding parameters such as bit-rate, frame-rate, spatial and temporal resolution. A MVT can be deemed as a joint transcoder [137] that operates on multiple synchronized video streams with significant inter-view coherence. Though a MVT can be used to change bit-rate, spatial/temporal resolution, etc, in this thesis, we limit our discussion on bit-rate transcoding. In particular, we assume that the input compressed video streams are produced by a MPEG2 encoder and have the same GOP (group of picture) structure. A MVT needs to compress both temporal redundancy and inter-view redundancy. Namely, motion-compensated prediction (MCP) and disparity-compensated prediction are both indispensable. To remove the inter-view redundancy among different video streams, we use the center approach [73, 74] to decorrelate the inter-view correlation among input video streams shown in Fig. 3.3(a).

In the center approach, all video streams are classified into two categories: main stream and secondary stream. The main stream is the center stream and encoded using only a MCP-based algorithm. Secondary streams are encoded with both MCP and DCP based on the main stream. The main stream can be chosen using the least square sum (LSS) technique proposed in [93], which will be elaborated in the next section. The proposed MVT can also use the neighbor approach discussed in Chapter 3 to remove inter-view redundancies. In that case, the first coded stream is the main stream and encoded with only a MCP-based algorithm. Other streams are secondary streams and encoded by using both MCP and DCP algorithms. There is no essential difference for the MVT used in the center approach and the neighbor approach. We use the MVT in the center approach to illustrate the key idea.

A MVT can be straightforwardly realized by fully decoding temporally compressed video streams and then using a spatio-temporal multiview video compression algorithm to re-encode the decoded video streams as shown in Fig. 4.2. We call this scheme the Cascaded Pixel Domain Multiview Video Transcoder (CPDMVT). However, it is computationally very expensive. Similar to traditional transcoders, the challenge for a MVT design is how to reduce the computational complexity of a straight-forward implementation and achieve the best possible video quality. Furthermore, unlike traditional transcoders, one unique challenge of a MVT design is how to utilize the coding statistics and parameters of input video streams to efficiently remove the inter-view redundancies.

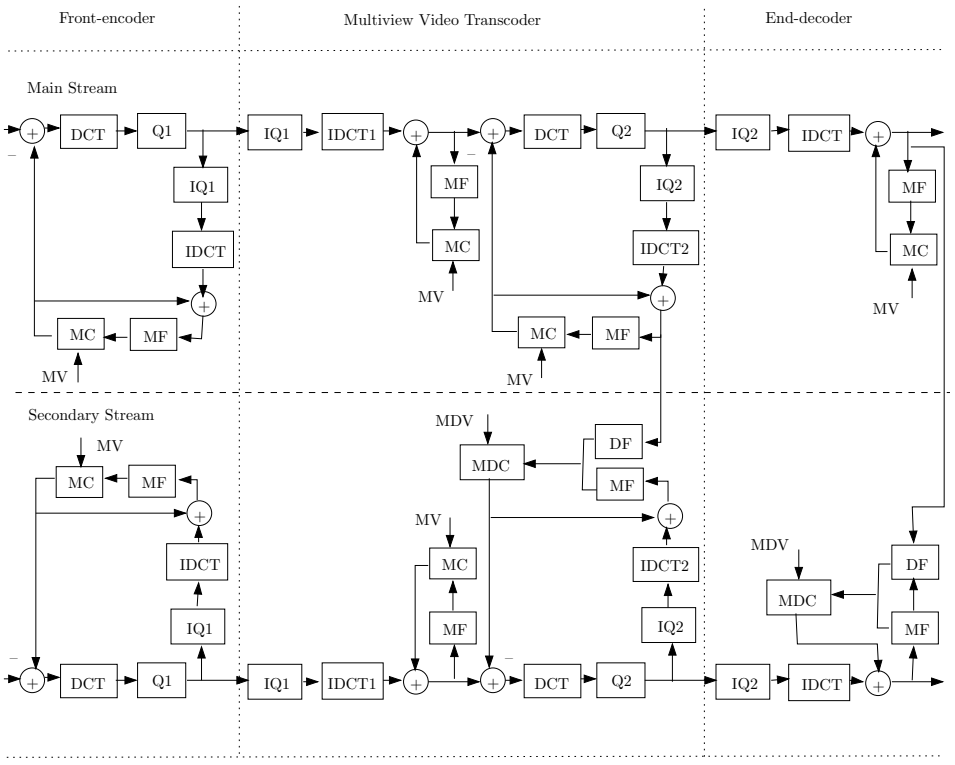


Figure 4.2: System Architecture of A Cascaded Pixel Domain Multiview Transcoder. DCT: Discrete Cosine Transform, IDCT: Inverse Discrete Cosine Transform, Q: Quantization, IQ: Inverse Quantization, MC: Motion Compensation, MDC: Motion and Disparity Compensation, MV: Motion Vector, MDV: Motion and Disparity Vector, MF: Frame Memory for Motion Compensation, DF: Frame Memory for Disparity Compensation.

4.3.2 Main Stream Selection

The main stream is used as a reference stream to predict and reduce inter-view redundancies among other streams. In this chapter, we assume that each camera location and its view direction are known. The least square angle sum (LSAS) technique proposed in [93] is used to choose the main stream. It is defined as

$$SAS_i = \sum_{j=1}^{m} A(S_i, S_j)^2$$

where SAS_i is squared sum, S_i and S_j is the stream i and j. Basically, there are two camera setups for multiview video acquisition, one is the parallel view and the other is the convergent view. For the parallel view, SAS_i is the squared distance sum and $A(\bullet)$ is the distance between the camera locations of S_i and S_j. For the convergent view, SAS_i is the squared angle sum and $A(\bullet)$ is the angle between the view directions of S_i and S_j. In either case, the stream with the least SAS is selected as the main stream. Other streams are secondary streams. Every frame in the main stream is used to predict the corresponding synchronized frames in secondary streams.

4.3.3 Simplified Pixel Domain Multiview Video Transcoder

The straightforward implementation of a CPDMVT is not desirable due to its high complexity. From Fig. 4.3, it is possible to simplify the main stream transcoding since the main stream is only temporally compressed. For the temporal only video transcoding, the motion vector of the incoming stream is normally reused in the outgoing stream. Namely, $MV_n^{m(1)} = MV_n^{m(2)} = MV_n^m$, where MV_n^m is the motion vector in the main stream.

$$I_n^{m(1)} = R_n^{m(1)} + P_n^{m(1)} \tag{4.1}$$

$$P_n^{m(1)} = S(I_{n-1}^{m(1)}, MV_n^m) \tag{4.2}$$

$$\begin{aligned} P_n^{m(2)} &= S(I_{n-1}^{m(2)}, MV_n^m) && (4.3)\\ &= S(I_{n-1}^{m(1)} + E_{n-1}^{m(2)}, MV_n^m) && (4.4) \end{aligned}$$

where $P_n^{m(i)}$ is the predicted pixel values from the previous frame, $R_n^{m(i)}$ is the residual error and $I_n^{m(i)}$ is the reconstructed frame. i can be "1" or "2" that represents the decoder or encoder loop of the main stream. $E_n^{m(2)}$ denotes the quantization error in the encoder loop of the main stream. $S(\bullet, \bullet)$ represents the motion compensation and is a shift operation. The motion compensation (MC) can be deemed as a linear operation, namely

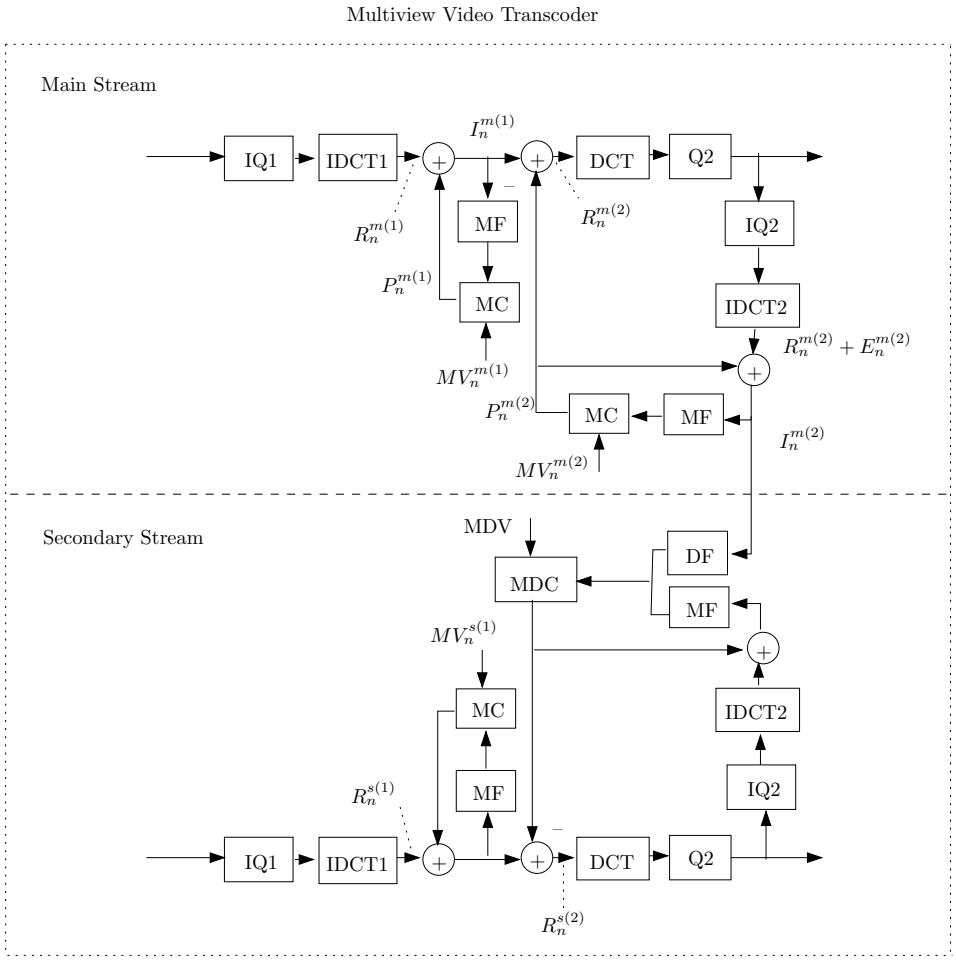


Figure 4.3: Cascaded Pixel Domain Multiview Transcoder

$S(X+Y, MV) = S(X, MV) + S(Y, MV)$. Based on this linearity assumption and Eq. 4.2, Eq. 4.4 can be rewritten as:

$$P_n^{m(2)} = S(I_{n-1}^{m(1)}, MV_n^m) + S(E_{n-1}^{m(2)}, MV_n^m) \quad (4.5)$$

$$= P_n^{m(1)} + S(E_{n-1}^{m(2)}, MV_n^m) \quad (4.6)$$

From Fig. 4.3, Eq. 4.1 and 4.6,

$$R_n^{m(2)} = I_n^{m(1)} - P_n^{m(2)} = R_n^{m(1)} - S(E_{n-1}^{m(2)}, MV_n^m) \quad (4.7)$$

$R_n^{m(2)}$ is the prediction residual error in the encoder-loop of the main stream transcoder and is sufficient to produce the output bit-stream for the main stream. From Eqn. 4.7, $R_n^{m(2)}$ can be obtained by subtracting the incoming residual error, $R_n^{m(1)}$, with the motion-compensated previous quantization error. Eqn. 4.1- 4.7 are well known in the transcoding literature [121, 122], and are presented here for completeness. Based on the Eq. 4.7 and the DCT-domain motion compensation, a Simplified DCT-Domain Transcoder (SDDT) [122, 115] can be easily designed to obtain the transcoded main stream as shown in Fig. 4.4.

However, SDDT cannot be used unless the frame in the transcoded main stream can be reconstructed and put into DF for redundancy elimination in secondary streams. In the following, we will analyze how to generate the reconstructed frame, $I_n^{m(2)}$, for three types of pictures: intra (I), predicted (P), and bidirectionally interpolated (B). Let DF_n denote the nth frame stored in DF.

For I frame, from Fig. 4.3,

$$f_n^I = I_n^{m(2)} = R_n^{m(2)} + E_n^{m(2)} \quad (4.8)$$

where f_n^I is the reconstructed I frame in the encoder loop of the main stream transcoder. The I frame will be directly stored into DF. Namely, $DF_n = f_n^I$.

For P frame,

$$f_n^P = I_n^{m(2)} \quad (4.9)$$

$$= P_n^{m(2)} + R_n^{m(2)} + E_n^{m(2)} \quad (4.10)$$

$$= S(I_{n-1}^{m(2)}, MV_n^m) + R_n^{m(2)} + E_n^{m(2)} \quad (4.11)$$

where f_n^p is the reconstructed P frame in the encoder loop of the main stream transcoder. From Fig. 4.3, $I_{n-1}^{m(2)}$ is the frame stored in MF. Observe that any frame in MF is the same as one in DF. Namely, $I_{n-1}^{m(2)} = DF_{n-1}$. Thus, Eq. 4.11 can be denoted using DF_{n-1}.

$$f_n^P = S(DF_{n-1}, MV_n^m) + R_n^{m(2)} + E_n^{m(2)} \quad (4.12)$$

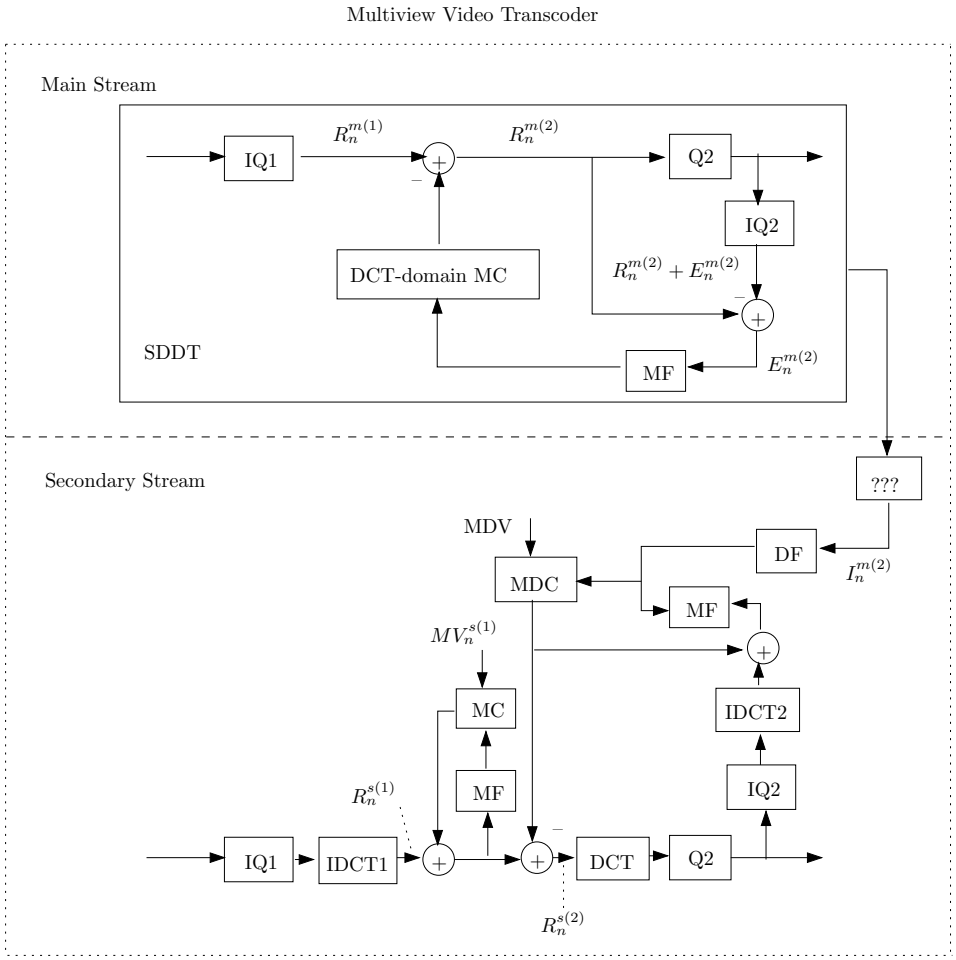


Figure 4.4: Simplified DCT-domain Transcoder (SDDT) for main stream

For B frame, we can get the similar derivation as the P frame.

$$\begin{aligned} f_n^B &= I_n^{m(2)} && (4.13)\\ &= P_n^{m(2)} + R_n^{m(2)} + E_n^{m(2)} && (4.14)\\ &= \alpha S(I_{n-1}^{m(2)}, MV_n^m) + (1-\alpha) S(I_{n-2}^{m(2)}, MV_n^m) + R_n^{m(2)} + E_n^{m(2)} && (4.15) \end{aligned}$$

where α is interpolation constant in the range [0 1]. The difference between B frame and P frame is that B frame needs two previous prediction frames, $I_{n-1}^{m(2)}$ and $I_{n-2}^{m(2)}$. Similarly, $I_{n-1}^{m(2)} = DF_{n-1}$ and $I_{n-2}^{m(2)} = DF_{n-2}$. Thus,

$$f_n^B = \alpha S(DF_{n-1}, MV_n^m) + (1-\alpha) S(DF_{n-2}, MV_n^m) + R_n^{m(2)} + E_n^{m(2)} \quad (4.16)$$

Note that the GOP of a MPEG2 bitstream always starts with I frame that is followed by P and B frame. Thus the first frame stored in DF is always I frame.

$$DF_0 = f_0^I \quad (4.17)$$

From Eq. 4.17, Eq. 4.15 and Eq. 4.12, we can easily derive the following equation.

$$DF_n = S(DF_{n-1}, MV_n^m) + R_n^{m(2)} + E_n^{m(2)} \quad (4.18)$$

Thus we get a multiview video transcoder with simplified main stream transcoder design. This is called Simplified Pixel-Domain Multiview Video Transcoder (SPDMVT) as shown in Fig. 4.5.

4.3.4 Motion and Disparity Compensated Prediction for Secondary Streams

To improve the compression ratio and reduce the distortion under the same bit-rate, each frame in secondary streams is predicted by both MCP and DCP as shown in Fig. 3.3(a). Specifically, I frames in secondary streams are encoded with DCP based on the I frame in the main stream. P and B frames are encoded with MCP of the previous frame and DCP of the synchronized P and B frame in the main stream. Each macroblock can be predicted with one of 6 modes: Forward (F), Backward (B), Disparity (D), Forward and Backward interpolation (FB), Forward and Disparity Interpolation (FD), Backward and Disparity Interpolation (BD). The mode that leads to the minimal Mean Absolute Difference (MAD) is chosen as the encoding mode. Namely,

$$R_n^{s(2)} = min\{R_F, R_B, R_D, R_{FB}, R_{FD}, R_{BD}\} \quad (4.19)$$

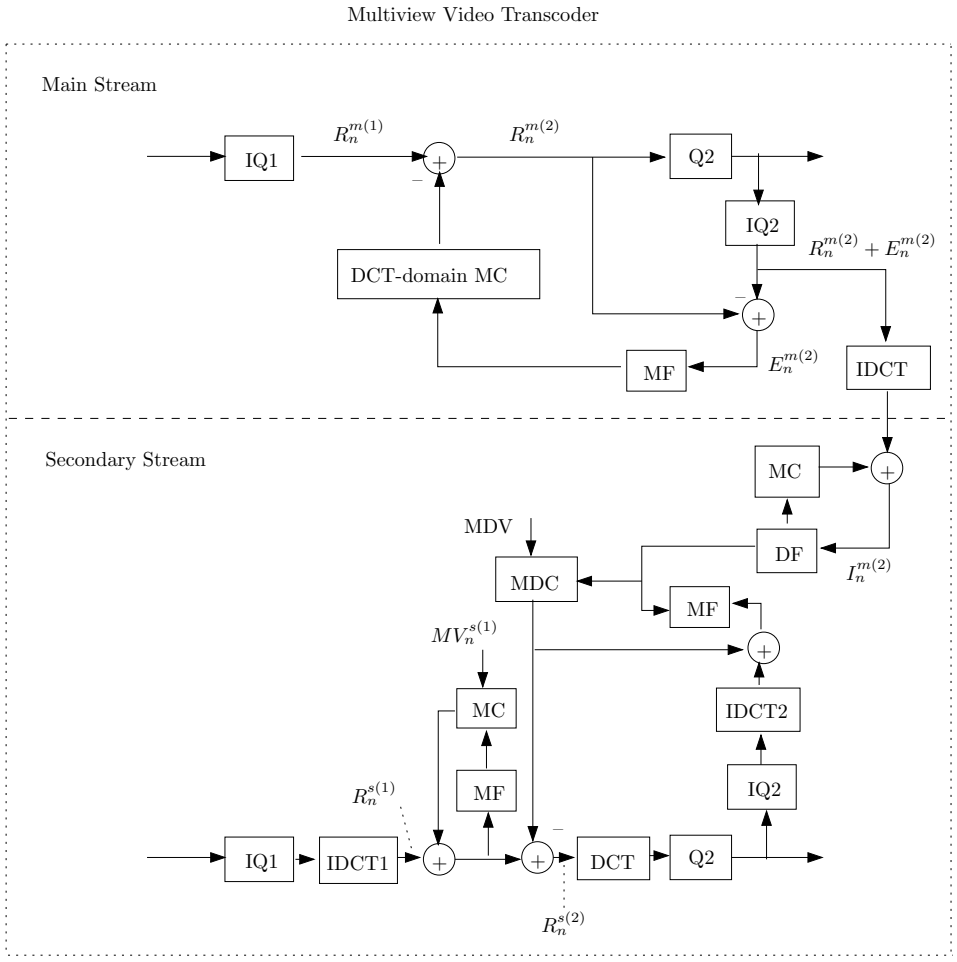


Figure 4.5: Simplified Pixel-Domain Multiview Video Transcoder

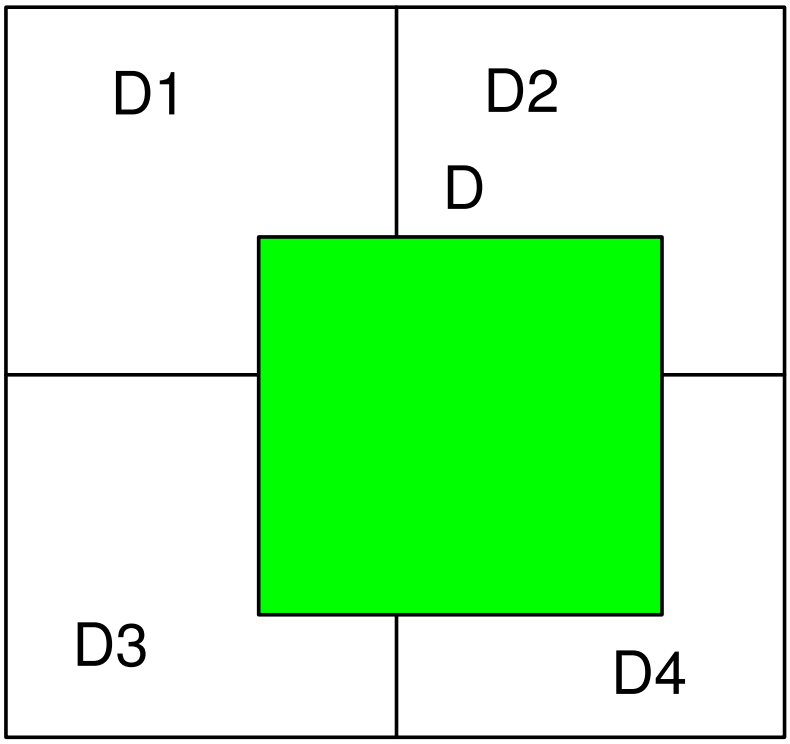


Figure 4.6: Search Window for Disparity Vector. D_i is the disparity vector for macroblock i.

where R_F, R_B, R_D, R_{FB}, R_{FD} and R_{BD} are the residual error of the corresponding macroblock mode.

Since the secondary stream transcoder can reuse the incoming motion vector, $MV_n^{s(1)}$, it is easy to calculate R_F, R_B and R_{FB} using Eq. 4.20.

$$R_t = f_n^s - S(f_{n-1}^s, MV_n^s) \tag{4.20}$$

where f_n^s denotes the nth frame in the secondary stream and $S(\bullet, \bullet)$ represents the motion compensation and is a shift operation.

The challenge is how to efficiently calculate R_D, R_{FD} and R_{BD}. Namely, the main difficulty is how to accurately and efficiently estimate the disparity vector. A simple motion-assisted block matching approach is proposed to estimate the disparity in a GOP (Group of Pictures). The basic idea is to use the disparity information in the adjacent frames to accelerate the estimation of disparity information in the current frame by reducing the search range. The pseudo code of the algorithm is shown in Algorithm 3.

The algorithm first finds the disparity vector of the I frame in a GOP using the full search window. Based on the assumption that the disparity vectors of the corresponding macroblocks between two adjacent frames only have small changes, the search window for

a macroblock in the current frame can be estimated by disparity vectors of 4 intersection macroblocks of the corresponding macroblock in the adjacent frame which can be found using the known motion vector. Fig. 4.6 illustrates a macroblock and its four intersection macroblocks in the neighbor frame. For P frame, the adjacent frame is the previous frame in the encoding order. For B frame, the adjacent frame is either of the previous two frames in the encoding order. Δd is used to account for the possible displacement of disparity vectors due to the movement between two adjacent frames.

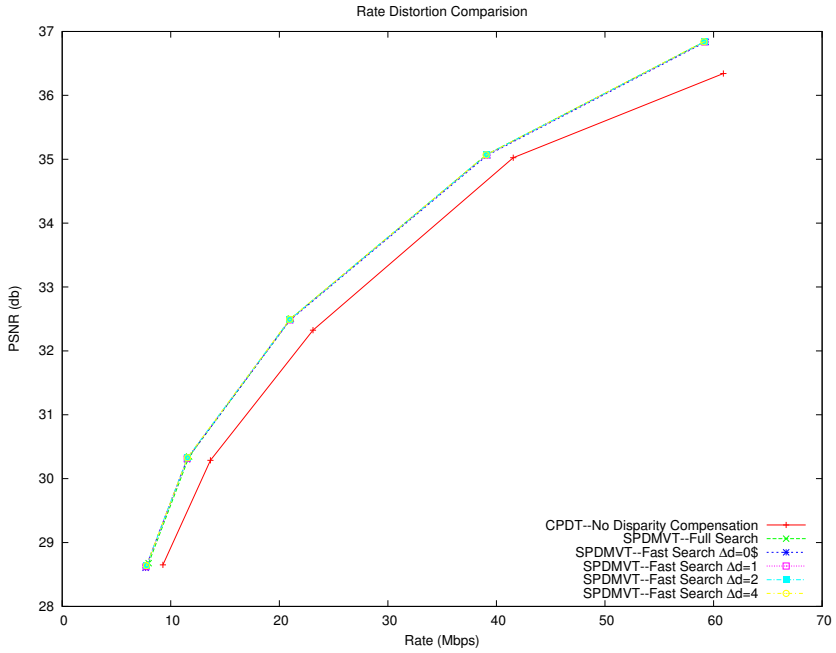
Algorithm 3 Pseudo Code for GOP-based Disparity Estimation

d_i^j: disparity vector for macroblock i in jth frame;
mb_i^j: macroblock i in jth frame;
mv_i^j: motion vector of macroblock i in jth frame;
d_{min}: minimal disparity vector;
d_{max}: maximal disparity vector;
Δd: the empirical bound of disparity vector displacement in two adjacent frames;
$I_c(P)$, $I_s(P)$: pixel intensity in the main stream and the secondary stream

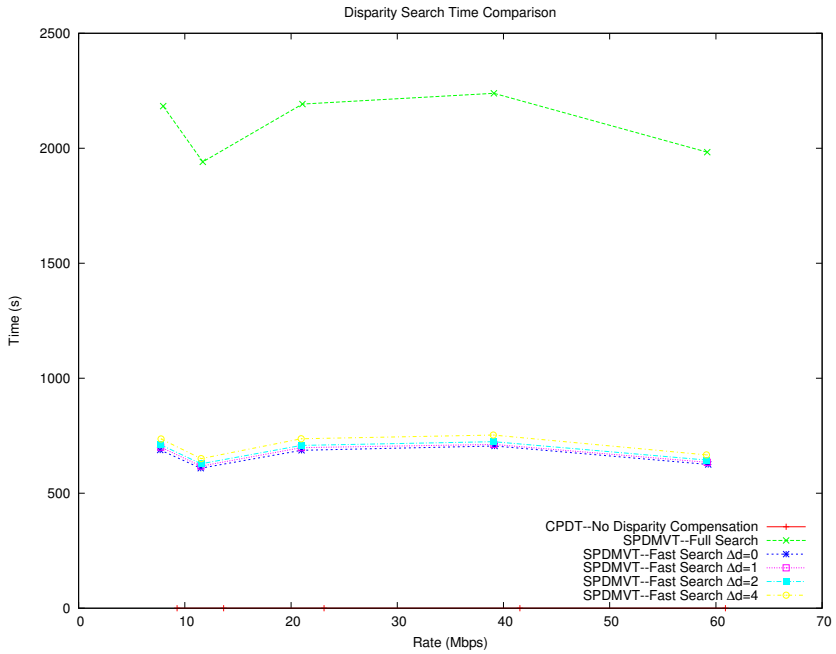
compute the disparity vector for each macroblock in a I frame using full search window;
for each remaining frame j in a GOP **do**
  for each macroblock i in the frame j **do**
    find the corresponding macroblock mb_i^{j-1} based on the motion vector mv_i^j in frame $j-1$;
    find the disparity vector d_i^{j-1} of the four surrounding macroblocks;
    $d_{min} = min\{d_i^{j-1}\}$;
    $d_{max} = max\{d_i^{j-1}\}$;
    search the disparity vector d_i^j based on $d_i^j = \underset{D}{\operatorname{argmin}} \sum_{P \in MB_i^j} |I_s(P) - I_c(P+D)|$, where $D \in [d_{min} - \Delta d, d_{max} + \Delta d]$;
  end for
end for

4.4 Experimental Results

We use both synthetic video and real video to evaluate the performance of the proposed multiview video transcoder. Synthetic video is rendered using POVRAY [112]. Cameras are placed in parallel lines and spaced uniformly in horizontal and vertical direction. This is a parallel camera setup. Seven views shown in Fig. 3.4 are used in the evaluation. Each input video stream is 24-bit RGB video with 640×480 pixels and 25 frames per second, and compressed using MPEG2 encoder at each camera. "Breakdance" and "Ballet" sequences [3] shown in Fig. 3.5 and Fig. 3.6 are used as real video examples. Both sequences

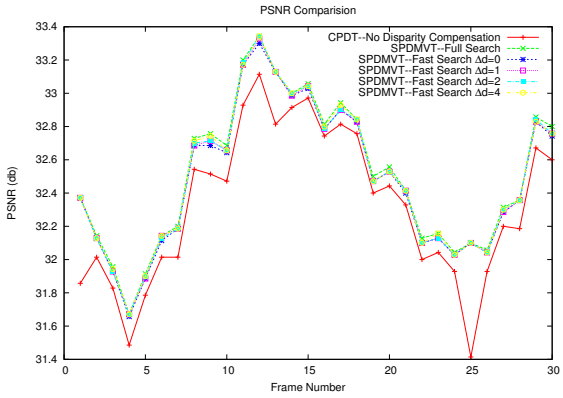


(a) Rate Distortion Comparison

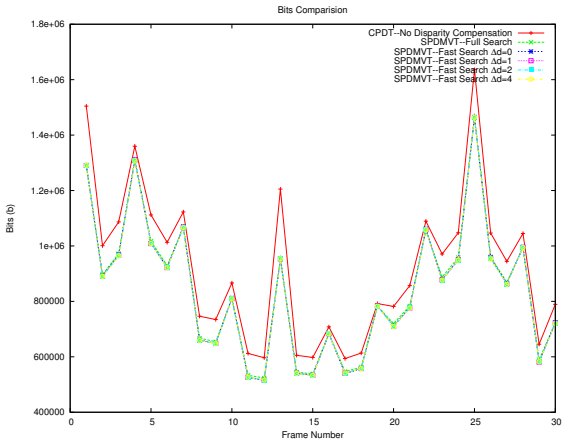


(b) Disparity Search Time Comparison

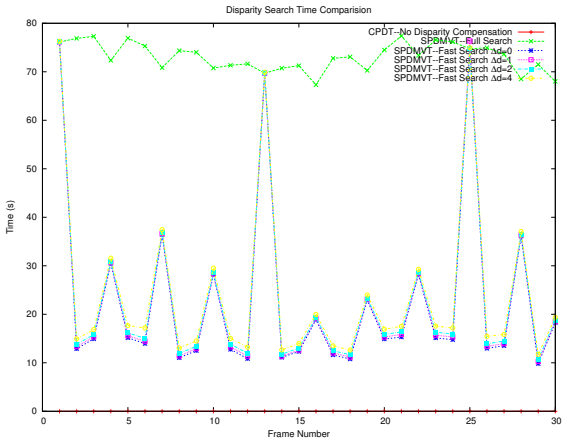
Figure 4.7: Rate-Distortion and Disparity Search Time Comparison for Synthetic Video



(a) PSNR Comparison

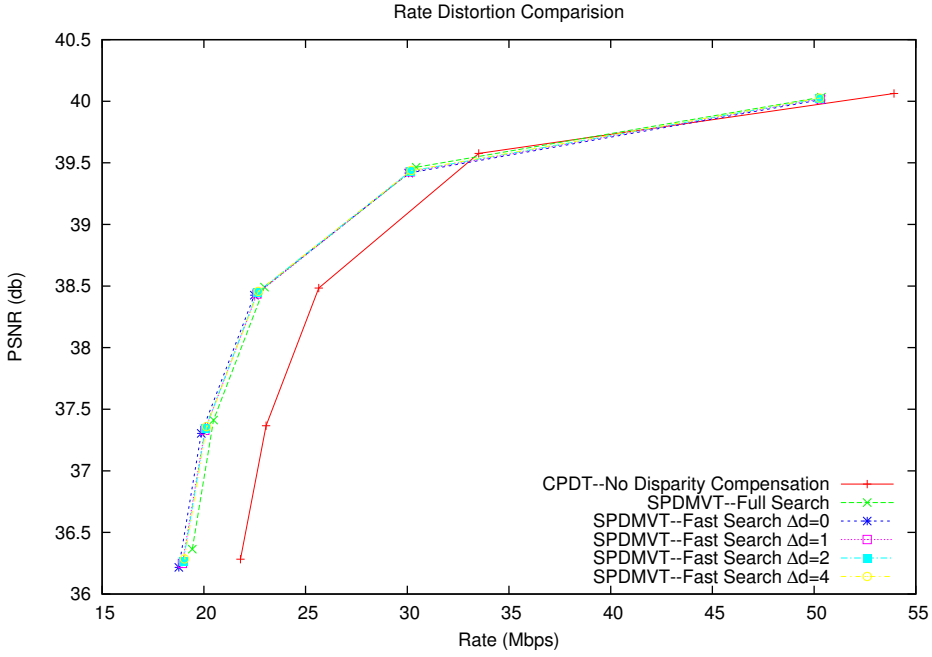


(b) Rate Comparison

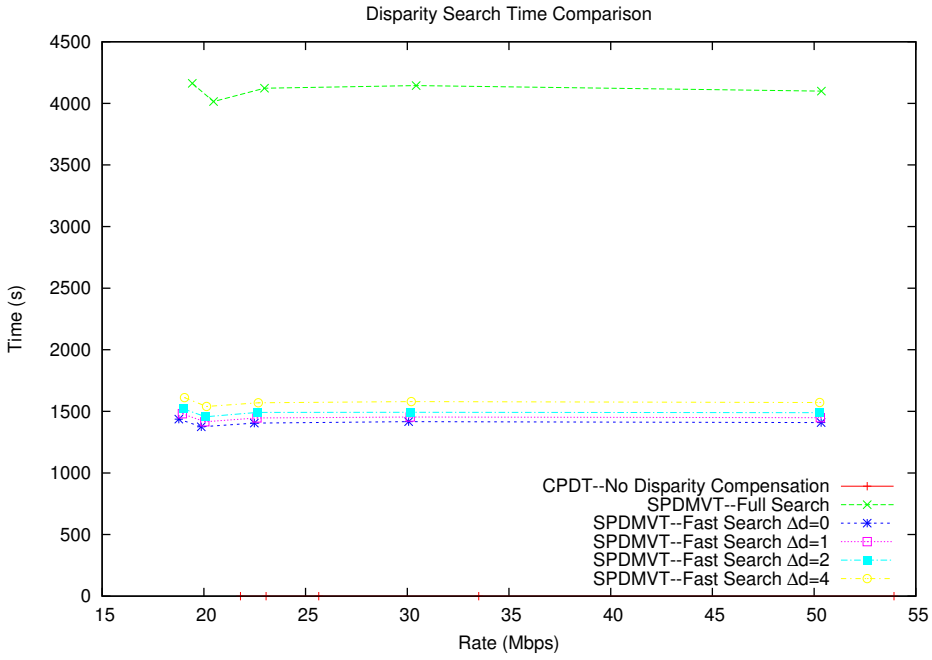


(c) Disparity Search Time Comparison

Figure 4.8: Per-Frame Comparison on PSNR, Rate and Disparity Search Time for Synthetic Video. $Q_1^I = 4$ $Q_1^P = 6$ $Q_1^B = 6$, $Q_2^I = 20$ $Q_2^P = 22$ $Q_2^B = 22$.

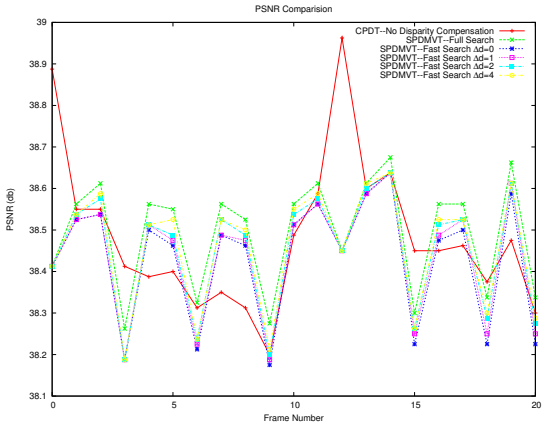


(a) Rate Distortion Comparison

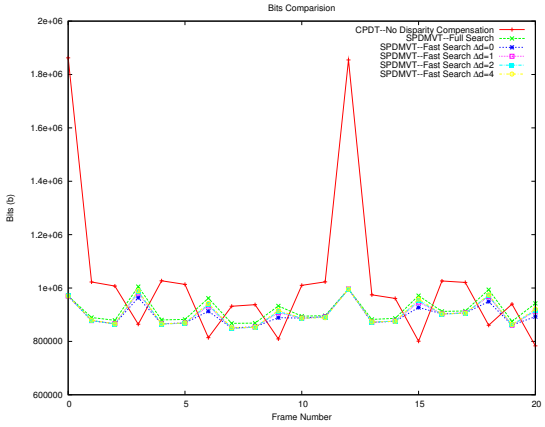


(b) Disparity Search Time Comparison

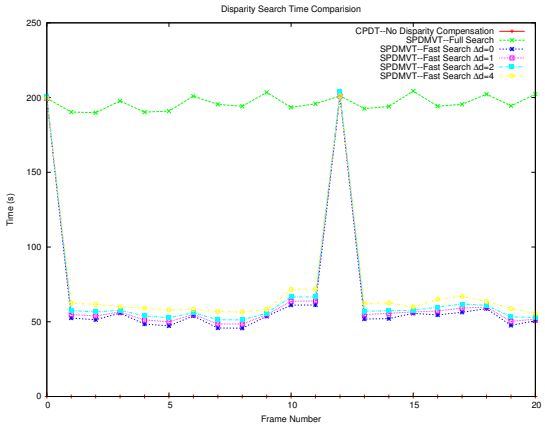
Figure 4.9: Rate-Distortion and Disparity Search Time Comparison for Breakdance Video



(a) PSNR Comparison

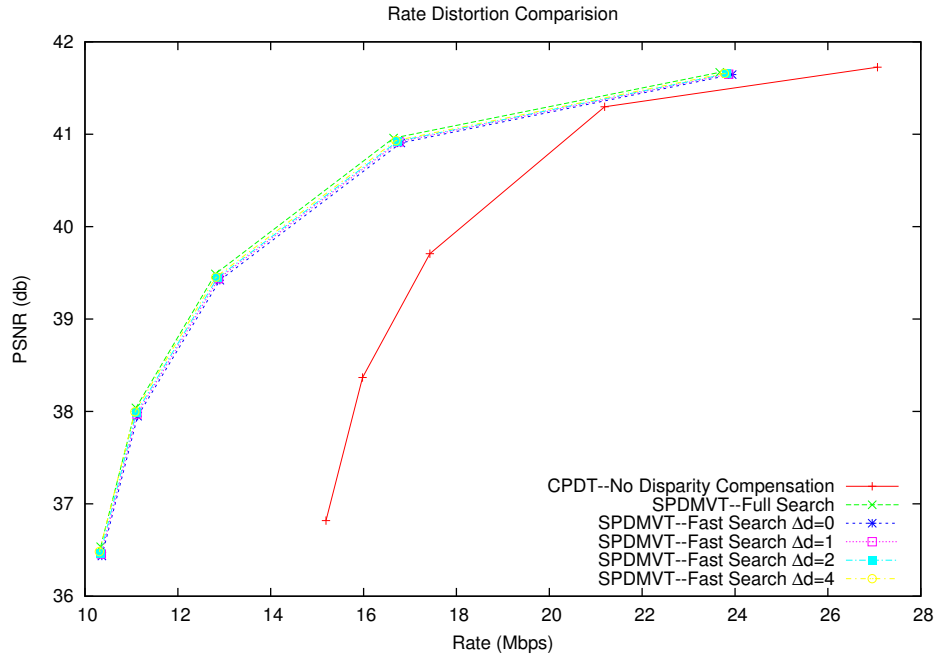


(b) Rate Comparison

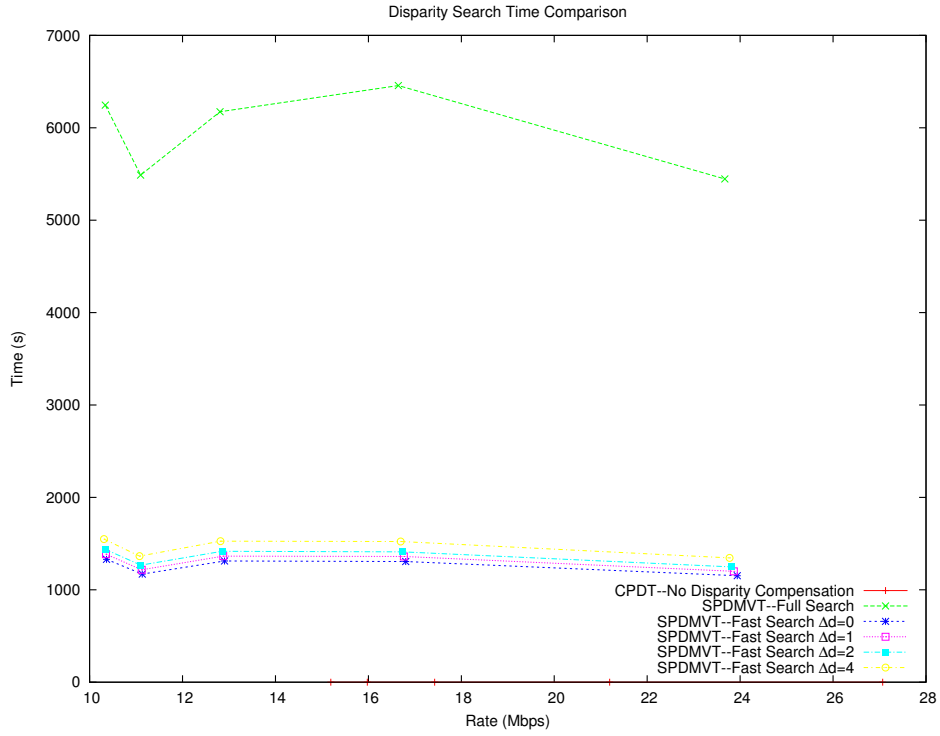


(c) Disparity Search Time Comparison

Figure 4.10: Per-Frame Comparison on PSNR, Rate and Disparity Search Time for Breakdance Video. $Q_1^I = 4$ $Q_1^P = 6$ $Q_1^B = 6$, $Q_2^I = 20$ $Q_2^P = 22$ $Q_2^B = 22$.

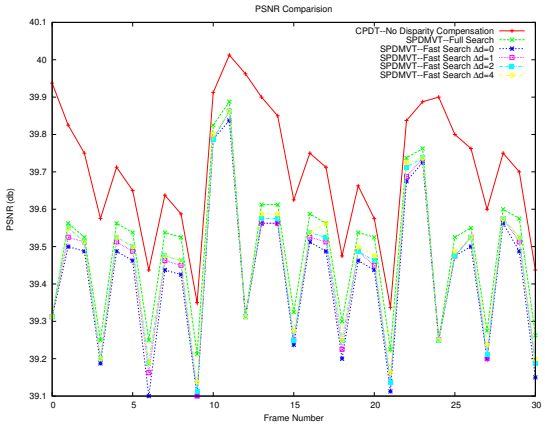


(a) Rate Distortion Comparison

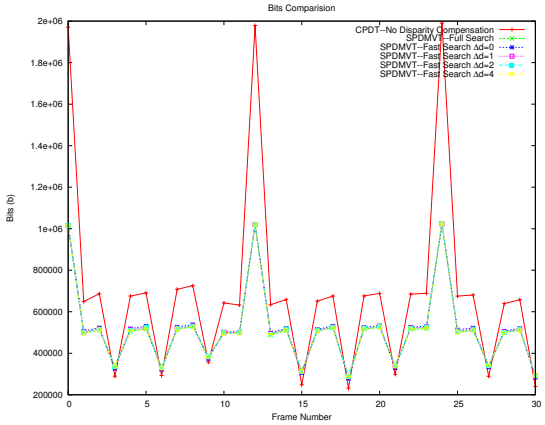


(b) Disparity Search Time Comparison

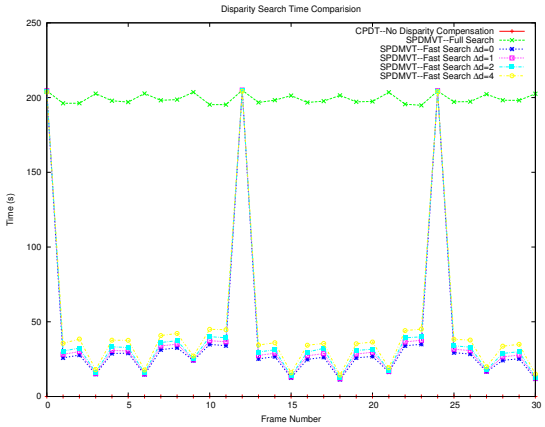
Figure 4.11: Rate-Distortion and Disparity Search Time Comparison for Ballet Video



(a) PSNR Comparison



(b) Rate Comparison



(c) Disparity Search Time Comparison

Figure 4.12: Per-Frame Comparison on PSNR, Rate and Disparity Search Time for Ballet Video. $Q_1^I = 4\ Q_1^P = 6\ Q_1^B = 6$, $Q_2^I = 20\ Q_2^P = 22\ Q_2^B = 22$.

have 8 views and are 24-bit RGB video with 1024×768 pixels.

We first study the performance of CPDT, SPDMVT with the full search window, and SPDMVT with GOP-based fast search window. CPDT does not use disparity search to remove the inter-view redundancy. The maximum disparity search window used in SPDMVT is 60 pixels. The only difference between SPDMVT with the full search window and SPDMVT with GOP-based fast search window is whether the GOP-based fast disparity estimation algorithm is used or not. The input compressed video streams to the multi-view video transcoder is encoded by MPEG2 and the quantization step size in MPEG2 is $Q_1^I = 4$ for I frame, $Q_1^P = 6$ for P frame, and $Q_1^B = 6$ for B frame. Fig. 4.7 and Fig. 4.9 and Fig. 4.11 present the comparison on rate-distortion performance and disparity search time of CPDT, SPDMVT with the full search window, SPDMVT with GOP-based fast search window over Synthesis, Breakdance and Ballet sequence. They show that MVT has a better rate-distortion performance than CPDT since disparity search helps reduce inter-view redundancies. In addition, compared with disparity search using full window, the proposed GOP-based fast disparity search algorithm can significantly reduce the search time while at the same time having a similar rate-distortion performance. The fast disparity search algorithm can save around 68% time for Synthetic video, 65% time for Breakdance video, and 80% time for Ballet video. The time used for disparity search in CPDT is 0 since there is no DCP in CPDT.

We also examine the sensitiveness of the parameter Δd. Fig. 4.7, Fig. 4.9 and Fig. 4.11 indicate that the GOP-based fast disparity search algorithm can accurately estimate the disparity range and a small Δd is enough to achieve similar performance as the disparity search using full window. Fig 4.8, Fig. 4.10 and Fig. 4.12 give a per-frame comparison on PSNR, rate, and disparity search time between CPDT and MVT for Synthetic, Breakdance and Ballet sequence when the quantization step size in CPDT and MVT is $Q_2^I = 20$ for I frame, $Q_2^P = 22$ for P frame and $Q_2^B = 22$ for B frame. PSNR is defined in Eqn. 2.20. Rate is measured by number of bits. Disparity search time is measured by number of seconds. They show that PSNR has slight variation based on video sequences. Compared with CPDT without disparity search, SPDMVT has a greater bit saving in I frame than in P and B frame since there is no motion compensation for I frames in CPDT and DCP can greatly reduce the inter-view redundancies in I frames. On the other hand, compared with SPDMVT with full disparity search, SPDMVT with GOP-based fast disparity search has no saving on search time in I frames since there is no motion prediction for I frames and thus no motion vector can be used to help estimate the disparity search window. Results on other quantization

parameters are similar.

4.5 Conclusion

In this chapter, we propose a novel multiview video transcoder and discuss its design challenges. In particular, a simple heuristic for fast disparity estimation is elaborated and evaluated. The results show that the proposed algorithm can effectively and efficiently compress the multiple compressed synchronized video streams. The GOP-based fast disparity estimation algorithm is able to reduce the computational complexity of disparity estimation and at the same time maintain the similar video quality as the one using exhaustive search.

Chapter 5

Multiview Video Coding Using Semi-Supervised Learning Algorithm

5.1 Introduction

Research on joint multiview video coding initially focuses on how to efficiently compress multiview video texture data based on conventional block-based prediction and transform coding approach by taking advantage of both temporal and inter-view similarity among multiple video streams [28]. Algorithms based on hierarchical B pictures [27] have been found to achieve the best performance. Recently some researchers start developing methods to efficiently compress both multiview video texture and depth data [80, 81]. Multiview-video-plus-depth (MVD) format is a popular data representation format for multiview video applications. The MVD format makes it easy to synthesize virtual views in client machines.

In the past decade, machine learning techniques such as clustering and classification have made significant progress [140]. Video coding techniques such as vector quantization based methods have inherent connection with clustering techniques [141]. Cheng and Vishwanathan [142] recently propose a new method based on semi-supervised learning [143, 140] to compress image and video chrominance data. Inspired by their work, we propose a learning-based multiview video coding (LMVC) scheme to compress multiview videos represented by the MVD format. This work is published in [82]. LMVC is also a joint multiview coding scheme. The basic idea of our scheme is to model the multiview video compression problem as a semi-supervised learning problem and achieve the compression by finding a sparse representation of multiview video. Depth data are compressed by H.264-based schemes. Luminance and chrominance data are compressed by semi-supervised learning algorithms.

The rest of the chapter is organized as follows. Section 5.2 gives an overview of semi-supervised learning techniques. Section 5.3 elaborates on the details of the proposed encoder and decoder. Experimental results are presented in Section 5.4. The chapter concludes in Section 5.5.

5.2 Semi-Supervised Learning

Semi-supervised learning refers to a set of classification techniques that use both labeled and unlabeled data for training. Formally, let $\mathcal{X}$ be the space of observations and $\mathcal{Y}$ the space of labels. $\mathcal{Y}$ is assumed to be a finite subset of $\mathbf{R}$. The semi-supervised learning [143, 140] is formulated as follows: Given a sequence $\{(\mathbf{x_i}, \mathbf{y_i})\}_{i=1}^{m}$ of labeled observations drawn from $\mathcal{X} \times \mathcal{Y}$, $\{\mathbf{x_i}\}_{i=m+1}^{n}$ of unlabeled observations drawn from $\mathcal{X}$, and a loss function $l : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \rightarrow \mathbf{R}$, learn a function $f \in H$ which minimizes the loss on the labeled observations and also generalizes well to new observations. In general, semi-supervised learning techniques can be classified into two categories: transductive learning and inductive learning. Transductive learning techniques can only work on the labeled and unlabeled data and cannot handle new data. Inductive learning techniques are able to learn a model and output a predictive function that is defined on the whole space. Namely, inductive learner can handle the new data. In the past decade, lots of semi-supervised learning methods such as transductive support vector machines, self-training, co-training and graph-based methods have been developed. Detailed taxonomy can be found in the recent survey [143] and book [140]. In this section, we will elaborate on graph-based semi-supervised learning techniques since our compression scheme is based on them.

5.2.1 Graph-based Semi-supervised Learning

Graph-based semi-supervised learning methods construct a graph, $\mathcal{G}$, whose nodes are the observations, and edges encode nearest neighbor relationships. These methods usually assume label smoothness over the graph which means that label values between adjacent nodes are similar. The semi-supervised learning problem is typically modeled as estimating a smooth function that respects neighborhood relations on the graph. Different functions can be estimated based on different choices of the loss function and the regularizer.

Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ where V is a finite set of n vertices denoted by $\{v_1, v_2, \cdots, v_n\}$ and E is a finite set of edges, $E \subset V \times V$. The adjacency matrix of $\mathcal{G}$ is an $n \times n$ real matrix W where w_{ij} is the weight on the edge (v_i, v_j) and can be any non-negative value, namely, $w_{ij} \in [0, \infty)$. The degree matrix, D, is defined as an $n \times n$ diagonal matrix with entries

$d_{ii} = \sum_j w_{ij}$. The graph Laplacian is the matrix $L = D - W$ while the normalized graph Laplacian is $\triangle = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$. The function can be estimated by minimizing the following regularized risk:

$$J(f) = c\|f\|^2_{\mathcal{H}} + \frac{\lambda}{n^2}\|f\|^2_{\mathcal{G}} + \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, f). \tag{5.1}$$

where l is a loss function and $\mathcal{H}$ is a Reproducing Kernel Hilbert Space (RKHS) of functions $f : \mathcal{X} \to \mathbf{R}$, and $\|f\|^2_{\mathcal{H}}$ imposes the smoothness conditions on possible solutions in RKHS, and $\|f\|^2_{\mathcal{G}}$ regularizes the solution to be smooth with respect to the graph, c and f are trade-off parameters for the regularizer, $\|f\|^2_{\mathcal{H}}$ and $\|f\|^2_{\mathcal{G}}$. The defining kernel is denoted by $K : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$, which satisfies $\langle f, K(x, \bullet)\rangle_{\mathcal{H}} = f(x)$. The regularizer $\|f\|^2_{\mathcal{G}}$ is defined as

$$\|f\|^2_{\mathcal{G}} = \mathbf{f}^\top \bigtriangledown_{\mathcal{G}} \mathbf{f} \tag{5.2}$$

where $\mathbf{f}$ is the vector $[f(x_i), \cdots, f(x_m), \cdots, f(x_n)]$, and $\bigtriangledown_{\mathcal{G}}$ denotes a specific regularization function over $\mathcal{G}$. The graph Laplacian L and $\triangle$ are typical choices. Namely,

$$\|f\|^2_{\mathcal{G}} = \mathbf{f}^\top \triangle \mathbf{f} \tag{5.3}$$

or

$$\|f\|^2_{\mathcal{G}} = \mathbf{f}^\top L^2 \mathbf{f} = \|Lf\|^2. \tag{5.4}$$

The loss function l is assumed to be piecewise differentiable and only depend on f via its evaluation at $f(x_i)$. The square loss function is normally chosen.

$$l(x_i, y_i, f) = (f(x_i) - y_i)^2 \tag{5.5}$$

Laplacian Regularized Least Square (LapRLS) algorithm [144] is used in [142] to solve equation 5.1. The solution f can be expressed as

$$f(\cdot) = \sum_{i=1}^{n} \alpha_i k(x_i, \bullet) \tag{5.6}$$

where α denotes the vector $[\alpha_i, \ldots, \alpha_m, \ldots, \alpha_n]$ and has a following closed form solution,

$$\alpha = (I_m K + cmI + \frac{\lambda m}{n^2} \bigtriangledown_{\mathcal{G}} K)^{-1} \mathbf{y}. \tag{5.7}$$

where $I_m \in \mathbf{R}^{n \times n}$ includes the $m \times m$ identity matrix on the top left hand corner and zeros elsewhere, I is the identity matrix, K is the Gram matrix $K_{ij} = k(x_i, x_j)$, and $\mathbf{y}$ denotes the label vector $[y_i, \ldots, y_m, 0, \ldots, 0]$.

Cheng and Vishwanathan [142] drop the regularization term $\|f\|^2_{\mathcal{H}}$ from the objection function 5.1 and learn the function f by minimizing

$$J(f) = \frac{\lambda}{n^2}\|f\|^2_{\mathcal{G}} + \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, f). \tag{5.8}$$

The resulting algorithm is still an inductive algorithm and the learned predictive function f is required to be smooth only on the observed data. It is sufficient for image and video compression applications since f is only used to predict unseen examples that are similar to the observed examples.

5.3 Learning-based Multiview Video Coding

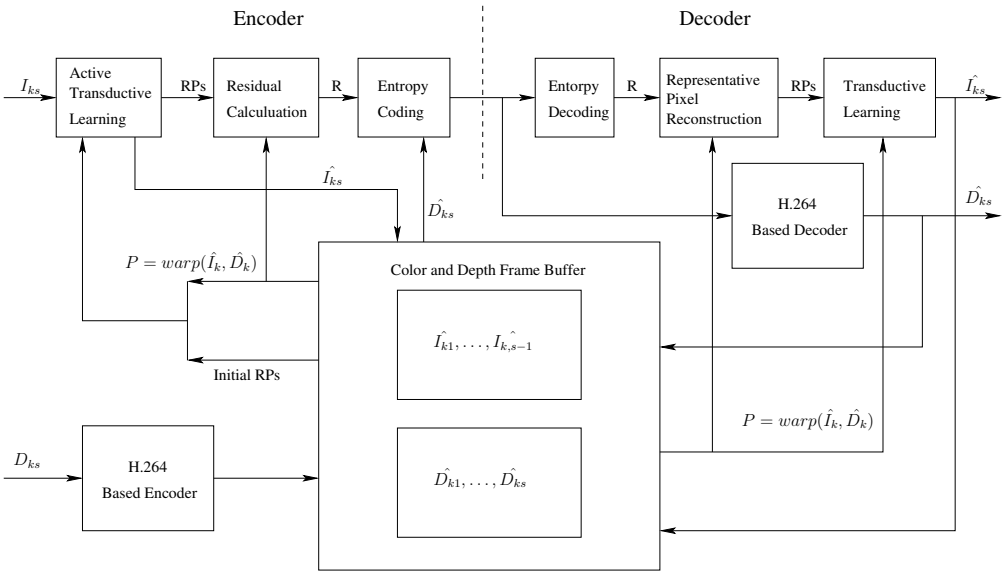


Figure 5.1: The Architecture of the Learning-based Multiview Video Coding Scheme

The proposed learning-based multiview video coding scheme aims at efficiently compressing multiview videos represented with MVD format. The basic idea is to first compress the depth map by the H.264-based schemes and then use an active semi-supervised learning method to compress texture data. LMVC compresses the texture image by choosing a small number of representative pixels (RP) at the encoder and uses the semi-supervised learning to reconstruct the original image based on the representative pixels at the decoder. Fig. 5.1 illustrates the architecture of the proposed learning-based multiview video coding scheme. A texture image includes both luminance and chrominance frames. The generic architecture of LMVC shown in Fig. 5.1 can be used to compress both luminance and chrominance frames. However, there is a slight difference to compress luminance and chrominance

frames. To compress a luminance frame, LMVC needs to first obtain an approximate frame through temporal/inter-view prediction. The approximate frame aims to construct the adjacency graph used in the graph-based semi-supervised learning algorithm. To compress the chrominance frames, LMVC uses the luminance frames to construct the adjacency graph like [142]. A depth map is used in LMVC to help generate approximate frames and reuse the representative pixels. Though LMVC can be used to compress both key pictures and non-key pictures [145], we focus on studying its performance on key pictures. Namely, only inter-view prediction is used to generate approximate frames and decide representative pixel reuse. Specifically, we concentrate on the inter-view prediction structure similar to the prediction structure for key pictures such as KS-IPP in [145]. Fig. 5.2 shows the prediction structure, KS-IPP, where each video stream is predicted by a neighbor video stream.

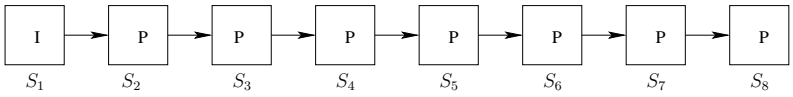


Figure 5.2: The Prediction Structure of the Learning-based Multiview Video Coding Scheme. S_i is the i-th video stream.

Since the eventual bit rate is proportional to the total number of representative pixels transmitted to the decoder, the challenge of LMVC is how to choose the minimum number of representative pixels to reach the target PSNR. Formally, let $\{I_{ij} = \langle Y_{ij}, U_{ij}, V_{ij} \rangle\}$, $i = 1, 2, \cdots, z$ and $j = 1, 2, \cdots, u$ denote a multiview video sequence where Y is the luminance frames, U and V are chrominance frames, u is the number of video sequences captured by u different cameras, and z is the number of images in each video sequence. Let $\{D_{ij}\}$, $i = 1, 2, \cdots, z$ and $j = 1, 2, \cdots, u$ denote the disparity map for image i in video stream j. For a specific image k in the video stream s, to compress the luminance frame, LMVC needs to first get the approximate frame $Y^p_{ks} = warp(\hat{Y_{k1}}, \cdots, \hat{Y_{k,s-1}}, \hat{D_{k1}}, \cdots, \hat{D_{k,s}})$, where $\hat{Y_{kj}}$ and $\hat{D_{kj}}, j = 1, \cdots, s-1$ are reconstructed luminance and depth frames in the video stream j. The approximate frame Y^p_{ks} is used in the semi-supervised learning algorithm to construct the adjacency graph. To compress the chrominance frames, luminance frames are first compressed using H.264-based schemes. The reconstructed luminance frame $\hat{Y_{ks}}$ is then used to construct the adjacency graph. In this chapter, we are interested in using the semi-supervised learning approach as illustrated in Fig 5.1 to compress $I_{ks}, s = 1, 2, \cdots, u$. Given a target distortion level denoted by the target PSNR, SNR_t, and the realized PSNR, SNR_r, the goal of our algorithms is trying to achieve

$|SNR_r - SNR_t| \leq \epsilon$, where ϵ is a preset threshold value, by minimizing the total bit rate. We will elaborate on the details of the algorithm in the following sections.

5.3.1 Transductive Learning

Similar to [142], we use graph-based semi-supervised learning to choose representative pixels (RPs). The learning objective is to minimize the generalized risk denoted by Equation 5.8. However, unlike the inductive learning approach in [142], we use the transductive learning approach since inductive learning is computationally expensive to inverse an $n \times n$ dense matrix in Equation 5.7 to solve the semi-supervised learning problem. In our current implementation, the transductive learning actually minimizes the following objective functions:

$$\sum_{i=1}^{n} \left(f(\mathbf{x_i}) - \sum_{i \sim j} w_{ij} f(\mathbf{x_j}) \right)^2 + \sum_{i=1}^{m} \delta(f(\mathbf{x_i}), y_i). \tag{5.9}$$

where w_{ij} is the weight between the pixel $\mathbf{x_i}$ and its neighbor pixels $\mathbf{x_j}$ and $\delta(f(\mathbf{x_i}), y_i)$ is a loss function which is 0 if $f(\mathbf{x_i}) = y_i$ and ∞ otherwise. The similar objective function is used in [146] to perform colorization on a video sequence. The difference between the Equation 5.9 and the objective function used in [142] is that we use δ loss function rather than the square loss function in [142]. The δ loss function enforces that the predicted values of the learned function at representative pixels are the same as the original ones. Minimizing Equation 5.9 actually is an optimization problem with quadratic cost function and linear constraints. It can be solved efficiently by using a large sparse system of linear equations [146].

To use the graph-based semi-supervised learning method, we need first to build an adjacency graph to denote the adjacency relationship between all labeled and unlabeled pixels. Each pixel has a feature map that includes the pixel value information of a 3×3 local window around each pixel in the predicted frame P_{ks}. Similar to [142, 146], a 4-nearest neighbor adjacency graph is constructed in the feature space based on the weight w_{ij} between any two pixels x_i and x_j. The weight w_{ij} is a normalized correlation function over the feature maps of two pixels [146].

$$w_{ij} \propto 1 + \frac{1}{\sigma_i^2}(\eta(i) - u_i)(\eta(j) - u_i) \tag{5.10}$$

where u_i and σ_i are the mean and variance of the pixel values in the local window around x_i, $\eta(i)$ and $\eta(j)$ are the pixel values of x_i and x_j.

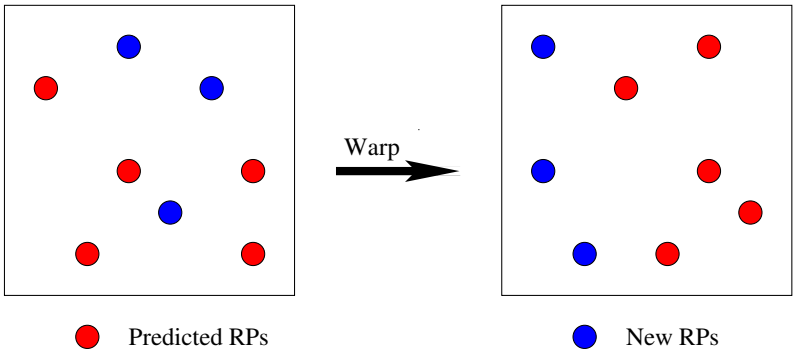


Figure 5.3: Representative Pixel Selection

5.3.2 Representative Pixel Selection

The key challenge of the LMVC scheme is how to choose as few representative pixels (RPs) as possible to achieve the target PSNR. Our basic strategy is to group pixels into clusters and choose representative pixels from each pixel cluster and reuse as many representative pixels in the neighbor frames as possible to reduce total bits used to encode the representative pixels. The representative pixel selection process is an active learning process. The semi-supervised learning algorithm actively chooses the best representative pixels to reach the target PSNR. The algorithm first encodes I_{k1} by using H.264 I frame. Denote the reconstructed frame as $\hat{I_{k1}}$. We use a mean-shift based segmentation algorithm [147] to process the frame $\hat{I_{k1}}$ and obtain an over-segmented image. A deterministic algorithm is used to choose q RPs from each segment and total m_{k1} RPs are selected. Represent each representative pixels, $\{RP_i^{k1}\}, i = 1, \cdots, m_{k1}$, as $([x_{iv}^{k1}, x_{iu}^{k1}], \hat{v_i^{k1}})$, where $[x_{iv}^{k1}, x_{iu}^{k1}]$ is the coordinates of RP_i^{k1} and $\hat{v_i^{k1}}$ is the quantized value of RP_i^{k1} in the frame I_{k1}. To encode the frame I_{k2}, each representative pixel in the frame I_{k1}, RP_i^{k1}, is warped into the frame I_{k2} to get a representative pixel, RP_j^{k2}. Namely, $RP_j^{k2} = ([x_{jv}^{k2}, x_{ju}^{k2}], \hat{v_j^{k2}}) = warp(RP_i^{k1}, \hat{D_{k1}})$. Note that $\hat{v_j^{k2}} = \hat{v_i^{k1}}$. The function $warp$ transforms a pixel in a frame captured by a camera X to a pixel in the frame captured by a camera Y. It is defined in the following equation:

$$warp_{X \to Y}(x, y, d) = \mathcal{K}_Y \mathcal{R}_Y^T \mathcal{R}_X \mathcal{K}_X^{-1}(x, y, 1)^T + d\mathcal{K}_Y \mathcal{R}_Y^T(\mathcal{T}_X - \mathcal{T}_Y) \tag{5.11}$$

where $\mathcal{K}_X$ and $\mathcal{K}_Y$ is the intrinsic calibration matrix for camera X and Y, $\mathcal{R}_X$ and $\mathcal{R}_Y$ are the rotation matrix that describes the orientation of the camera X and Y in a world coordinate system, $\mathcal{T}_X$ and $\mathcal{T}_Y$ are the transition vectors that specify the position for camera X and Y in a world coordinate system, (x, y) is the coordinate of the pixel in the image X and d is the depth.

If the coordinate of RP_j^{k2} is within the boundary of the frame I_{k2}, RP_j^{k2} is used as

a representative pixel. If the signal to noise ratio (SNR), $SNR_{p_j}^{k2}$, for RP_j^{k2} is greater than the target PSNR, SNR_t, namely, $SNR_{p_j}^{k2} \geq SNR_t$, the predicted pixel value, $\hat{v_j^{k2}}$, of RP_j^{k2} is kept as the quantized pixel value. Otherwise, $\hat{v_j^{k2}}$ is assigned a new quantized value which guarantees that its signal to noise ratio is greater than the target PSNR. A flag, f_p^{k2}, is set for each predicted representative pixel to indicate whether the predicted pixel value is used as the pixel label. In this way, we ensure that every predicted representative pixel has correct label and helps the transductive learning algorithm find right labels for unlabeled pixels. By reusing the representative pixels, total m_{k2}^p representative pixels can be used as initial representative pixels to train the semi-supervised learner for the frame I_{k2}. If the semi-supervised learning algorithm cannot learn a good model based on the predicted representative pixels to make the predicted frame reach the target PSNR, the predicted pixel value of every unlabeled pixel based on the current model is evaluated and pixels in high error areas are identified and clustered. The algorithm then chooses q representative pixels from each cluster and assigns those RPs correct labels which ensure that their signal to noise ratios are greater than the target PSNR. The newly chosen representative pixels are added into the labeled pixel set and the semi-supervised learner starts to train a new model. The process repeats iteratively until the realized PSNR, SNR_r, is within a threshold, ϵ, of the target PSNR, SNR_t. Fig. 5.3 illustrates the representative pixel selection process. Finally, m_{k2}^n new representative pixels are selected. Therefore, total $m_{k2} = m_{k2}^p + m_{k2}^n$ representative pixels are needed to compress the frame I_{k2}. To encode the frame I_{k3}, m_{k2} representative pixels in the frame I_{k2} are first warped to m_{k3}^p RPs in the frame I_{k3}. This process goes on until all u frames are encoded.

5.3.3 Residual Calculation

After choosing the right number of representative pixels to reach the target PSNR, residuals need to be computed so that the entropy encoding can achieve a high compression efficiency. Given a frame I_{kj} and its predicted frame P_{kj}, the straightforward way to calculate the residual is to directly subtract the corresponding pixel value, p_i^{kj}, in P_{kj} from the quantized pixel value of a representative pixel, $\hat{v_i^{kj}}$. Namely, $R_i^{kj} = \hat{v_i^{kj}} - p_i^{kj}$. However, the method might not be able to get good compression performance since the best prediction for each representative pixel might be its neighbor representative pixels in the frame I_{kj}. In this section, we propose a neighbor-graph based residual calculation scheme.

Given m_{kj} representative pixels and their coordinates and quantized pixel value, our algorithm first constructs a t-nearest neighbor graph based on the Manhattan distance between

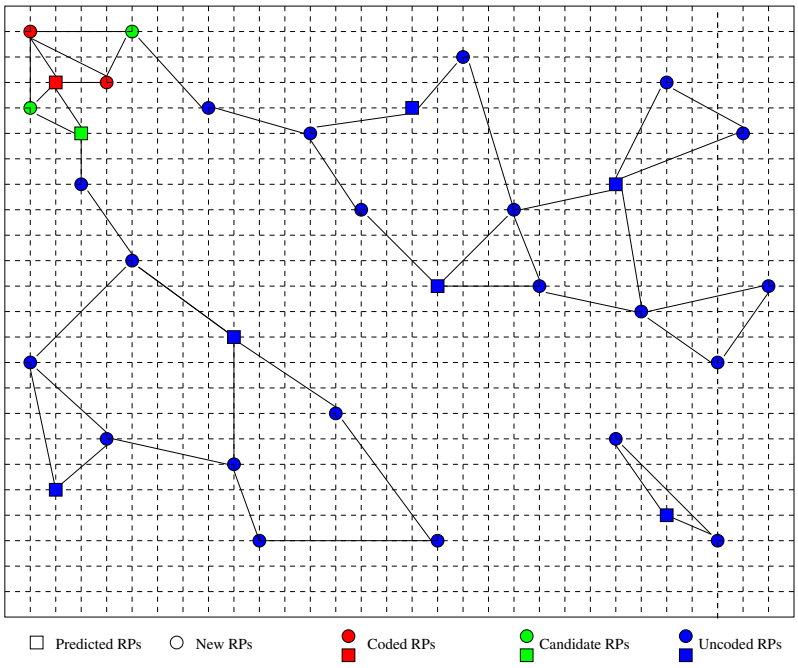


Figure 5.4: The Illustration of Neighbor Graph of RPs and Residual Calculation Process

coordinates of different representative pixels, which is defined in the following equation.

$$d(RP_i, RP_j) = |x_i - x_j| + |y_i - y_j| \tag{5.12}$$

where RP_i and RP_j are two represent pixels with coordinates (x_i, y_i) and (x_j, y_j).

Each representative pixel in the graph has at least t neighbors and its degree is greater than or equal to t. Fig. 5.4 shows a 2-nearest neighbor graph. Each representative pixel maintains an ascending order list of its neighbors based on the Manhattan distance and coordinates. The neighbors are first ordered by Manhattan distance. When two neighbors have the same Manhattan distance, the neighbor with smaller coordinates is in front of the one with larger coordinates. Based on the neighbor graph, the algorithm can decide a unique encoding order to calculate the residual of each representative pixel after the first coded representative pixel is determined. The first coded representative pixel can be randomly chosen or follow some guidelines such that the representative pixel with the smallest coordinate is always coded first as the case shown in Fig 5.4 or the representative pixel with the smallest absolute residual value in the predicted frame is always coded first. The residual of the first coded representative pixel is always calculated by deducting its corresponding pixel value in the predicted frame P_{kj}.

The algorithm maintains three set of representative pixels: coded RP set, *SC*, candidate RP set, *SN*, uncoded RP set, *SU*. *SC* includes the representative pixels whose residual is

already calculated. *SN* includes the representative pixels from which the next coded representative pixel will be selected. *SU* is composed of representative pixels whose residuals are not yet computed. For each representative pixel, the algorithm also maintains a variable, f_m, to indicate from which representative pixel its residual is calculated. For a t-nearest neighbor graph, f_m has $t+1$ values to indicate $t+1$ possible mode. $f_m = 0$ means that the residual is calculated from the corresponding pixel in the predicted frame P_{kj}. $f_m = 1, \cdots, w, \cdots, t$ means that the residual is computed from the wth-nearest neighbor representative pixel. Once the residual of the first representative pixel is coded, it is added into the set *SC*. Then all its uncoded neighbor representative pixels are added into the set *SN*. The next coded representative pixel is chosen from the set *SN* and has the property that its shortest Manhattan distance with one of representative pixels in the set *SC* is smallest among all the representative pixels in the set *SN*. When two or more representative pixels have the same shortest Manhattan distance from the representative pixels in the set *SC*, the representative pixel with the smallest coordinate is chosen to be coded next. The chosen representative pixel calculates its residual with all its neighbor representative pixels in the coded set *SC* and then selects the residual whose absolute value is the smallest as its final residual and sets f_m to indicate the neighbor representative pixel that leads to the smallest residual. The process continues until the residuals of all the representative pixels are calculated or the set *SN* becomes empty. When *SN* is empty, it means that the neighbor graph is disconnected as the case shown in Fig. 5.4 and then the representative pixel with the smallest coordinate in the uncoded set *SU* is chosen to be the next coded representative pixel. Experiments show that the neighbor graph approach has a better compression efficiency than the straightforward one.

5.3.4 Entropy Coding

After calculating the residual for all representative pixels, the coordinates and residuals of representative pixels are encoded by using arithmetic coding. Coordinates are coded first. Only coordinates of new representative pixels in the frame $\{I_{kj}\}, j = 2, \cdots, u$ are coded. The coordinates of representative pixels in the frame I_{k1} that are chosen by the deterministic representative pixel selection algorithm need not be coded since the decoder can use the same segmentation algorithm and deterministic representative pixel selection algorithm to find them. The flag, f_p, which indicates whether the pixel values of the predicted representative pixels are reused or not, are then coded. Finally, the residuals of unreused representative pixels and new representative pixels and their prediction mode f_m are coded.

Note that the coding order of residuals must follow the same order from which residuals are computed. A zero-order word-based arithmetic coding algorithm [148] is used in our implementation.

5.3.5 Decoding

As shown in Fig. 5.1, the decoder first computes the representative pixels based on the reconstructed frame $\hat{I_{k1}}$ by using the same algorithm as the encoder. Then it decodes the coordinates of new representative pixels in the frame I_{k2} and then builds the same neighbor graph based on the coordinates of all representative pixels. Following the algorithm that computes the residuals at the encoder, the decoder can recover pixel values of all representative pixels in the frame I_{k2}. Lastly, based on the representative pixels and the predicted frame, the transductive learning algorithm can reconstruct the original frame $\hat{I_{k2}}$. The process continues until all the frames are decoded.

5.4 Experiments

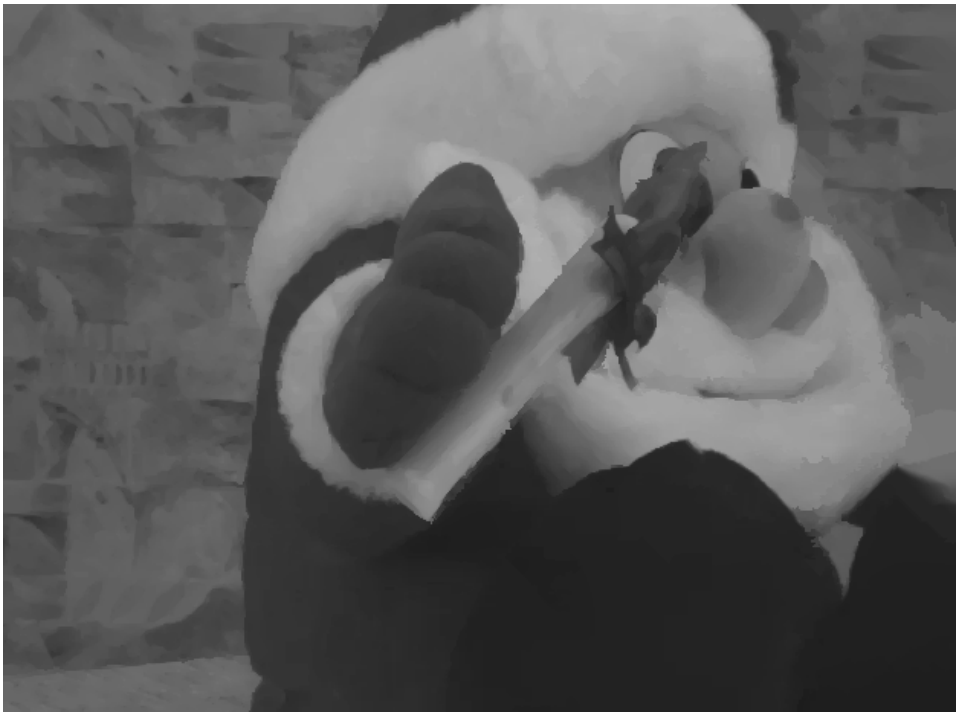
We evaluate the performance of LMVC on compressing luminance frames and chrominance frames and compare its performance with JMVM. The 624×464 "Santa Claus" and 1072×768 "Breakdance" [3] multiview video sequence are used to study the performance of our algorithm. The "Santa Claus" sequence has 9 views and the "Breakdance" sequence has 8 views.

LMVC is first used to compress the luminance frames. The Y components of multiview video sequences are compressed. We first examine whether representative pixel selection algorithm works properly and selects right pixels to compress multiview videos. Fig. 5.5 shows some decoded images and distribution of representative pixels over the Y frame for "Santa Claus" sequence during the training phase. The PSNR threshold, ϵ, used in this experiment is 0.1dB and the target PSNR, SNR_t, is 43.11dB. This shows that representative pixel selection algorithm works well and the algorithm could automatically choose representative pixels along boundaries and around occlusion regions where there are sharp pixel intensity change and great prediction error and thus it is generally hard for the semi-supervised learner to assign them correct labels.

Fig. 5.6 shows the evolution of PSNR and the number of RPs during the course of training for view 2 of the "Santa Claus" sequence. It has similar characteristics to the results described in [142]. Namely, initially it takes a relatively small number of representative pixels to quickly improve the realized PSNR, SNR_r, however, it takes a large number of

(a) Original Y Frame

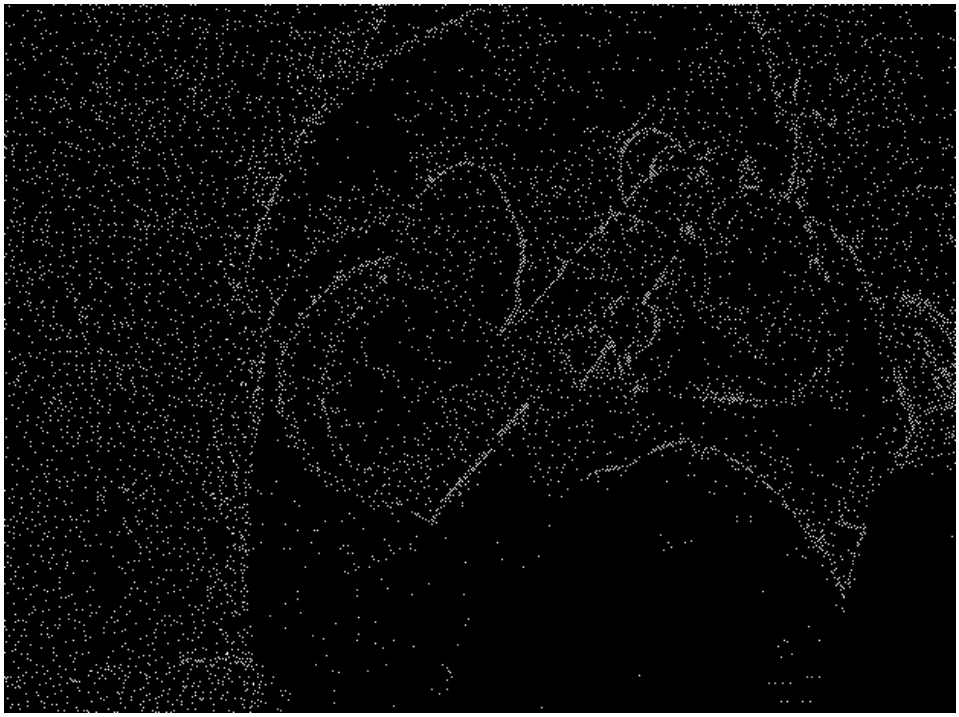
(b) Depth



(c) Decoded Y Frame, Iter = 1

(d) RPs, Iteration = 1

(e) Decoded Y Frame, Iter = 51

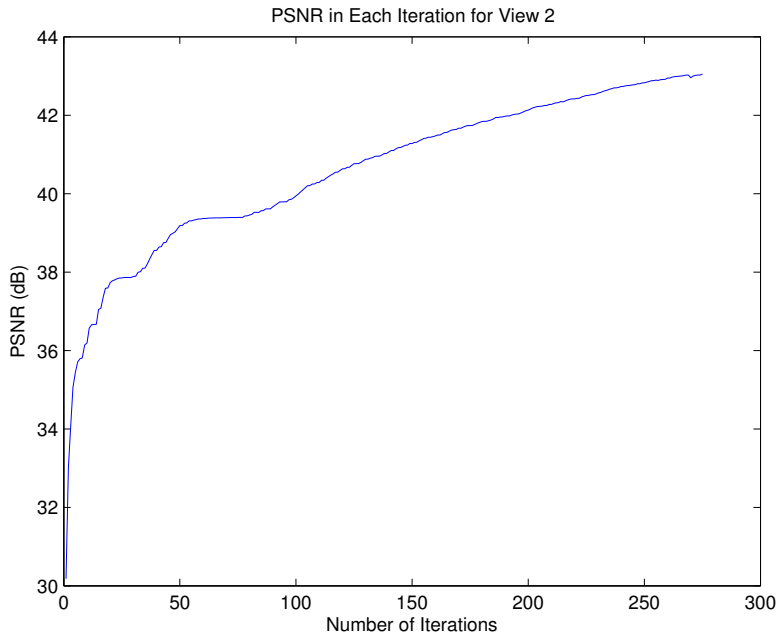


(f) RPs, Iter = 51

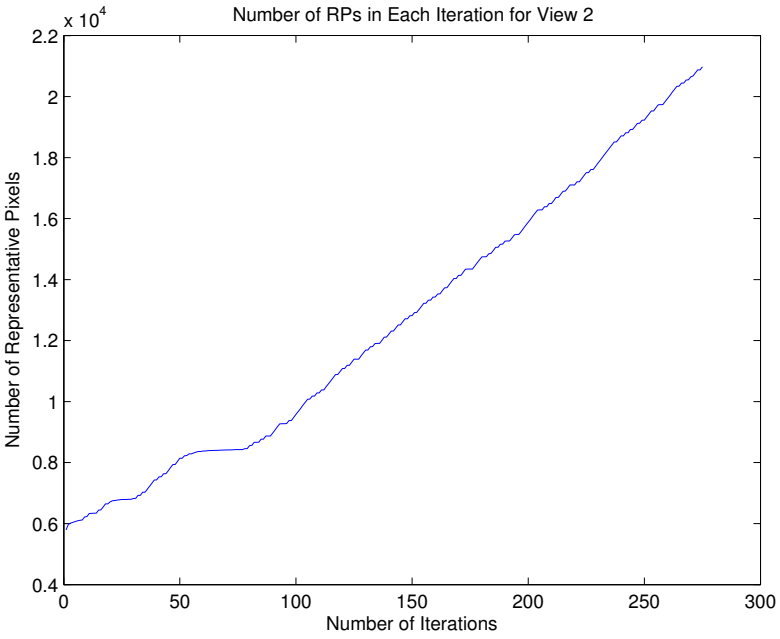
(g) Decoded Y Frame, Last Iter

(h) RPs, Last Iter

Figure 5.5: Compressing Luminance Frames of “Santa Claus” by LMVC



(a) PSNR Evolution



(b) Number of RPs Evolution

Figure 5.6: PSNR and Number of RPs Evolution Example When Compressing Y Frames of "Santa Claus" Sequence

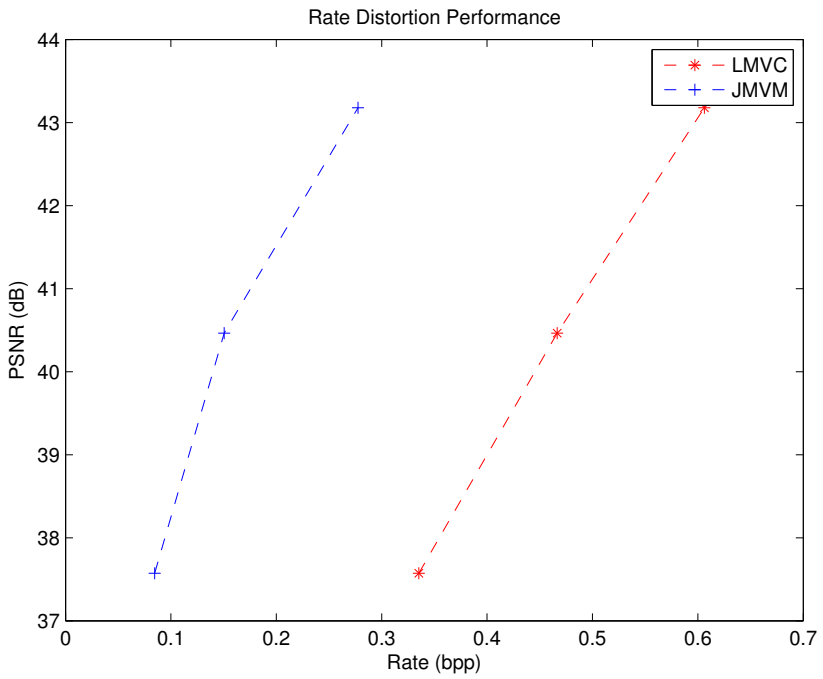


Figure 5.7: Rate-Distortion Performance of LMVC on luminance frames of "Santa Claus" Sequence

representative pixels in later iterations but with only moderate improvement of signal to noise ratio. Results for other views are similar.

Fig. 5.7 shows the rate-distortion performance of LMVC on compressing luminance frames of the "Santa Claus" sequence. There is still a large gap between LMVC and JMVM. The reason for the poor performance of LMVC on luminance frames might be because the predicted frame based on frames in the neighbor views is not good enough for LMVC to construct a good adjacency graph to predict the unknown pixel values based on the known representative pixels. Results for the "Breakdance" video is similar.

We also evaluate the performance of LMVC on compressing chrominance frames, namely, the U and V component of an image. In this case, the luminance frame is used to construct the adjacency graph. Fig. 5.8 shows the decoded U frames and representative pixels of "Breakdance" sequence during the training phase. The target PSNR, SNR_t, in this experiment is 42.8dB and PSNR threshold, ϵ, is 0.1dB. The process to compress the V frames is similar.

Fig. 5.9 shows the evolution of PSNR for U frames and number of RPs when LMVC is used to compress the chrominance frames of "Breakdance" sequence. Similar to the case of compressing luminance frames, PSNR can improve very fast initially as the number of

(a) Original U Frame

(b) Depth

(c) Decoded U Frame, Iter = 1

(d) RPs, Iteration = 1

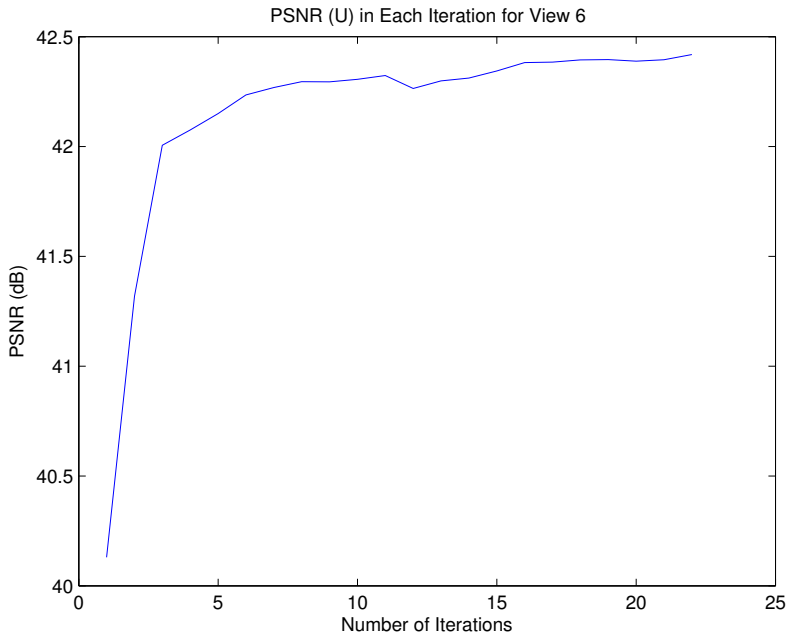
(e) Decoded U Frame, Iter = 11

(f) RPs, Iter = 11

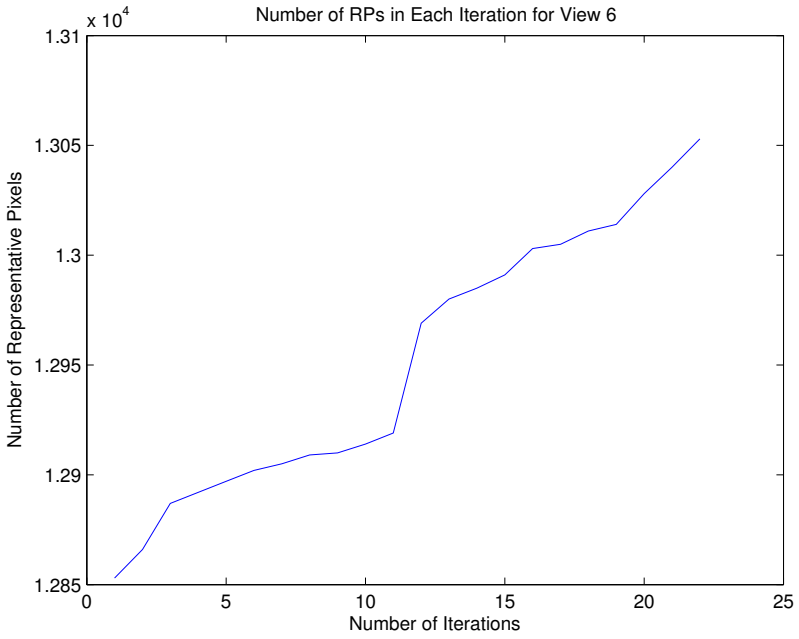
(g) Decoded U Frame, Last Iter

(h) RPs, Last Iter

Figure 5.8: Compressing Chrominance Frames of “Breakdance” Sequence by LMVC



(a) PSNR Evolution for U Frames



(b) Number of RPs Evolution

Figure 5.9: PSNR and Number of RPs Evolution Example When Compressing Chrominance Frames of "Breakdance" Sequence

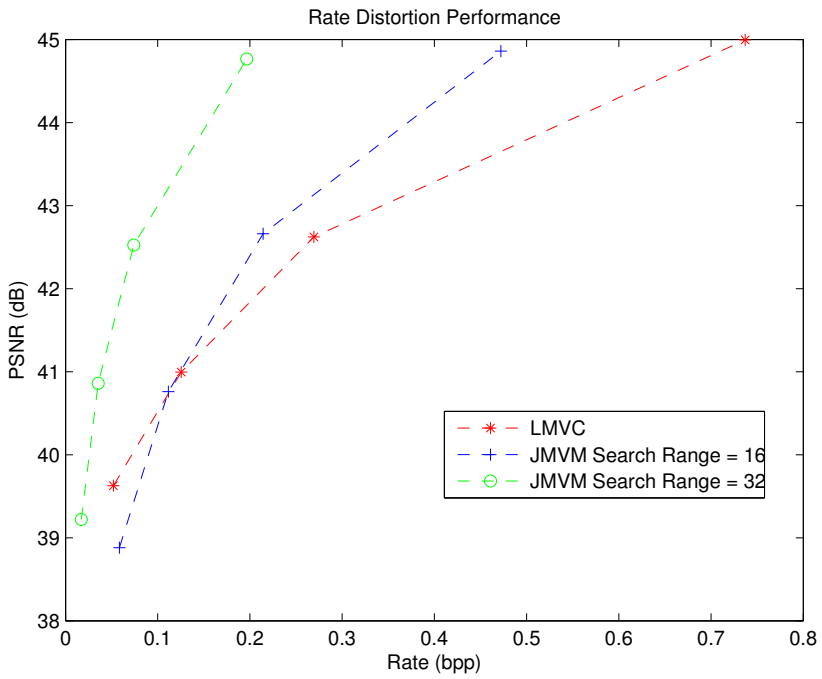


Figure 5.10: Rate-Distortion Performance of LMVC on chrominance frames of "Santa Claus" Sequence

representative pixels increases. A relative large number of representative pixels can only get moderate PSNR improvement at later iterations.

Rate-distortion performance of LMVC on compressing chrominance frames is also compared with JMVM. Fig. 5.10 and Fig. 5.11 show the rate-distortion performance of LMVC on compressing chrominance frames of "Santa Claus" sequence and "Breakdance" sequence. LMVC has a large performance gap with JMVM when the target PSNR is high and can achieve a comparable performance with JMVM when the target PSNR is low. The reason might be that the algorithm can learn a good prediction function from a small number of representative pixels to achieve a relative low target PSNR but need a very large number of representative pixels to learn an accurate prediction function for the high target PSNR.

5.5 Conclusion

In this chapter, we propose a novel learning-based multiview video compression framework. LMVC intends to efficiently compress multiview video represented by the MVD format. We model the multiview video compression as a semi-supervised learning problem and find ways to solve it efficiently. LMVC roots at the sound learning theory and is

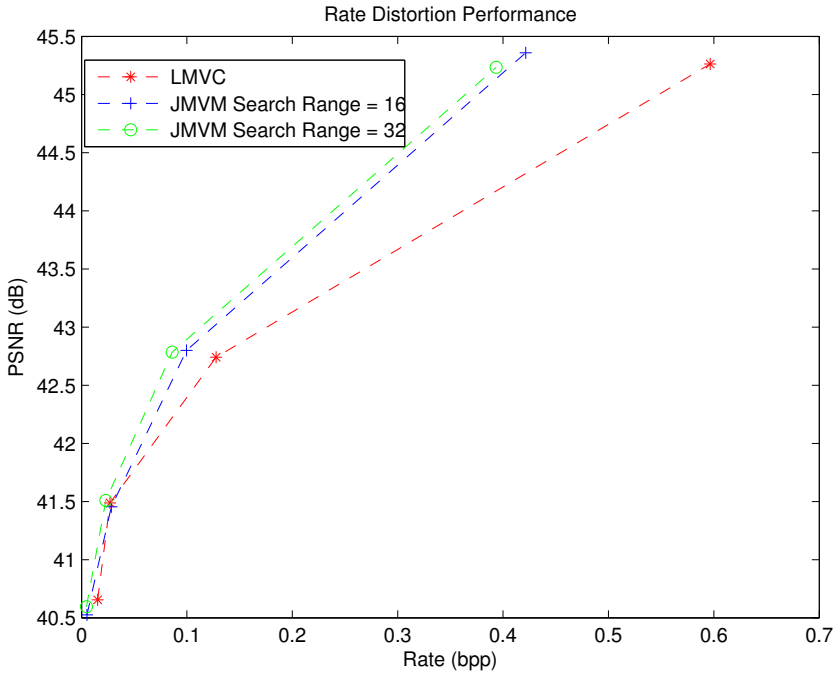


Figure 5.11: Rate-Distortion Performance of LMVC on chrominance frames of "Breakdance" Sequence

significantly different from previous multiview video coding approaches. The results show that our scheme still has a significant gap with JMVM when it is used to compress luminance frames. In the case of compressing chrominance frames, there is also a large gap with JMVM when the target PSNR is high. LMVC can achieve a comparable performance as JMVM when the target PSNR is relatively low. The rate-distortion performance of LMVC depends on how good the adjacency graph is. When the label values between connected nodes in the adjacency graph is similar, the performance is good as the case for chrominance frame compression. Otherwise, the performance is bad as the case for luminance frame compression.

Chapter 6

Symmetric Distributed Source Coding

6.1 Introduction

As discussed in Chapter 2, distributed source coding is an underlying scheme for distributed multiview video coding. It also has broad applications in wireless sensor networks [149]. Distributed source coding is based on the theorem proved by Slepian and Wolf in 1970s [40]. The fascinating aspect of distributed source coding is that efficient compression of two or more sources can be achieved by separate encoding and joint decoding. The achievable rate region of distributed source coding is shown in Fig. 2.4. As we mentioned in Section 2.1.3, distributed source coding methods can be grouped into two classes: asymmetric distributed source coding (ADSC) and symmetric distributed source coding (SDSC). ADSC corresponds to the corner point in the achievable rate region shown in Fig. 2.4. SDSC refers to the schemes that can achieve an arbitrary rate point in the achievable rate region.

Slepian and Wolf's seminal work inspires other researchers to produce dozens of ensuing theoretical papers [150]. However, practical distributed source coding algorithm was not developed until Pradhan and Ramchandran's work in 1999 [151]. Since distributed source coding is dual to channel coding [42], powerful channel codes such as lattice codes, convolutional codes, Turbo codes and low density parity check codes (LDPC), can all be used to realize distributed source coding. Algebraic binning is the basic idea to construct the practical distributed source codes, which was first introduced by Wyner in the seventies [152]. The majority of early research effort is focused on asymmetric distributed source coding. A survey in 2004 [149] summarizes the early results. With asymmetric distributed source codes, symmetric distributed source coding can be achieved straightforwardly through time

sharing or source splitting approaches [153, 154]. However, these methods are hard to deploy in real world because time sharing requires accurate time synchronization and source splitting induces practical performance loss and increases implementation complexity.

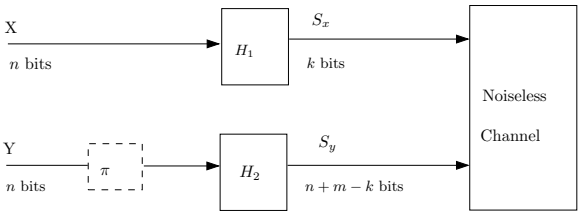
Although it appears difficult to design one channel code to achieve the entire Slepian-Wolf rate region at first sight, several researchers have proposed excellent solutions to realize symmetric distributed source coding [155, 156, 157, 158, 159, 160]. These approaches can be classified into two classes: parity-based approaches and syndrome-based approaches. [160, 158] are parity-based approaches, where a decoder recovers the original source bits by processing the information bits and parity bits generated by an encoder. Garcia-Frias and Zhao [160] propose one of earliest papers on symmetric distributed source coding, which lets two correlated sources X and Y pass through a Turbo code encoder and achieves the arbitrary rate by puncturing the systematic bits and parity bits. Sartipi and Fekri [158] use non-uniform LDPC codes and rate-compatible LDPC codes to achieve the entire rate region. [155, 156, 157, 159] are syndrome-based approaches where a decoder deciphers the original source from part of source bits and syndrome bits generated by an encoder. [155, 156, 157] share the similar idea that a decoder first retrieves the syndrome of the difference pattern between X and Y, and then decodes the difference pattern from the syndrome by using a powerful Turbo or LDPC code, and finally solves linear equations to obtain all the original information bits by exploiting the unique property of linear channel codes. Schonberg [159] uses the sum-product algorithm in an extended Tanner graph to decode original source bits. [156, 157] needs one linear channel code to encode both sources. [155, 159], which both have roots in [161], construct independent subcodes from one main channel code and use them to realize the symmetric distributed source coding.

The aforementioned symmetric distributed source coding approaches are elegant and able to create capacity-approaching codes. However, it is still difficult to use them in the real world. There are two main disadvantages in previous symmetric distributed source coding schemes. First, the decoding procedure in previous syndrome-based approaches [155, 156, 157, 159] is a dependent process. Namely, the decoding of sources are dependent to each other. In other words, given two sources X and Y, to recover X (Y), the decoding process needs to iteratively utilize the partial information of Y (X). The dependent decoding procedure propagates the decoding error and make it hard to design capacity-approaching code. Unlike these syndrome-based approaches, the parity-based scheme in [158] proposes a decoding process where the decoding of sources are independent to each other. Second, all previous approaches implicitly assume that an encoder knows the exact bit correspon-

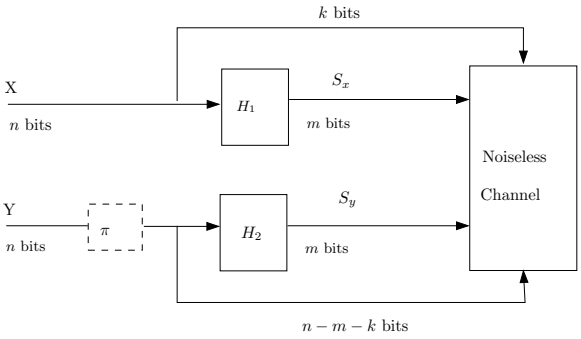
dence between correlated sources, X and Y. However, this assumption is not true in most cases due to various reasons. For example, in the camera sensor network, pixel correspondence between two correlated images is not known at the encoder and can only be inferred at the decoder since there is no communication channel between two encoders. Without the knowledge of bit correspondence, many algorithms [156, 158, 155] will fail to work since bit mismatch problem makes it impossible for the decoder to correctly recover the syndrome of the difference pattern between X and Y, and thus the decoder is unable to decipher the difference pattern, $X \oplus Y$, though they offer mathematically provable capacity-approaching codes. The decoder normally knows the bit correspondence. Though authors do not specifically address the bit mismatch problem, the algorithm [159] can successfully decode the original mismatched sources because the process to recover the bit correspondence can be easily integrated into the decoding process as shown in Fig. 6.1(a) and 6.1(c), where bit correspondence between sources is modeled as a mapping, π, which specifies the bit mapping between two sources. However, the code partitioning technique makes it difficult to design capacity-approaching codes since it requires both the main code and two subcodes to be good codes [159]. Specifically for LDPC code, authors in [159] give a future research direction to possibly use a nested LDPC design algorithm to construct capacity-approaching codes. if not impossible, it is still hard to realize such a nested code design procedure.

In this chapter, we strive to design novel symmetric distributed source coding schemes to address the shortcomings of previous schemes. First, inspired by the work of Sartipi and Fekri [158], we propose a syndrome-based symmetric distributed source coding scheme that has an independent decoding procedure. Similar to [158], non-uniform parallel channels are used to model the correlation between two sources. The scheme is called Syndrome-based Nonuniform Symmetric Slepian-Wolf Coding scheme (SNS-SWC). Second, inspired by the work [156, 159], we propose an enhanced symmetric distributed source coding approach that combines both advantages of the two previous approaches. Namely, at the encoder we use one capacity-approaching code to encode the correlated sources, which helps us circumvent the difficulty of code construction in [159]; at the decoder we use the message-passing algorithm on the extended Tanner graph, which makes it easy to handle the bit mismatch problem. We call the approach Enhanced Extended Tanner Graph (EETG) method. We also put forward a simple heuristic to construct the extended Tanner graph to achieve better decoding performance. Furthermore, we introduce a general framework to handle the bit mismatch problem. This work is published in [162]

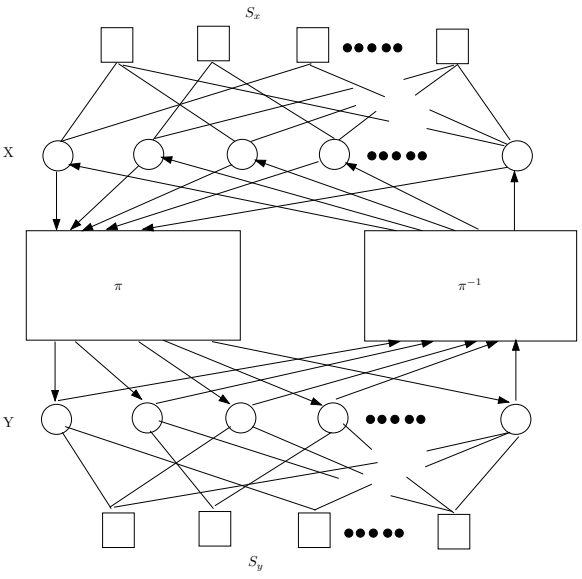
The remainder of the chapter is organized as follows. Section 6.2 gives an overview of



(a) Encoder Using Two Subcodes. H_1 and H_2 are the parity check matrix of respective subcodes.



(b) Encoder Using One Channel Code. For EETG, H_1 and H_2 are the parity check matrix of one LDPC code and its permutation equivalent code. For SSIF, $H_1 = H_2$.



(c) Decoder Using Extended Tanner Graph

Figure 6.1: Illustration of Encoder and Decoder Using Extended Tanner Graph. π is a mapping, which is used to change the bit order of a source.

three previous symmetric distributed source coding approaches [158, 156, 159] in detail. SNS-SWC is presented in Section 6.3. In Section 6.4, we elaborate on the details of EETG. Simulation results are presented in Section 6.5. Conclusion is drawn at Section 6.6.

6.2 Background

In this section, we first give some definitions related to bit correspondence that will be used in the later sections. Then we give an overview of the symmetric distributed source coding schemes proposed in [158, 159, 156]. The scheme in [158] is called "Parity-based Nonuniform Symmetric Distributed Source Coding scheme (PNS-DSC)". The approach in [159] is termed "Two-machine Algorithm (TM)", and the method in [156] is called "Symmetric SF-ISF (Syndrome Former - Inverse Syndrome Former) Framework (SSIF)". These approaches are general and can be realized with any linear channel code. To facilitate easy exposition, we use LDPC codes as an example to explain the basic ideas. The virtual correlation channel is BSC.

6.2.1 Bit Correspondence

Given two n-bit sources, X and Y, bit correspondence is defined as a mapping between bit X_i and its correlated bit, Y_j. Namely, let π be the mapping, then $\pi(i) = j$. Given a bit location, i, which satisfies $1 \leq i \leq n$, and a mapping, π, let $\gamma = |i - \pi(i)|$. If $0 \leq \gamma \leq n$, the mapping π is called arbitrary mapping. If $0 \leq \gamma \leq t < n$, the mapping π is called bounded mapping. if the mapping is independent of time and source blocks and an encoder knows the mapping as a priori knowledge, the encoder is said to be aware of bit correspondence between two sources. Otherwise, the encoder is oblivious to the bit correspondence. In PNS-DSC, TM and SSIF, the authors assume that the mapping between two sources is an identity mapping and an encoder knows the mapping. Namely the mapping, π, shown in Fig. 6.1 has the form $\pi(i) = i$.

6.2.2 Parity-based Nonuniform Symmetric Distributed Source Coding

PNS-DSC [158] achieves the entire Slepian-Wolf rate region by sending parity bits. Fig. 6.2 illustrates its architecture. The basic idea of PNS-DSC is to use two systematic LDPC codes to encode two correlated sources X and Y. The correlation channel is assumed to be BSC (Binary Symmetric Channel). The k-bit source X is encoded by a systematic LDPC code of rate $r_X = \frac{k}{k+|P_1|}$, where $|P_1|$ denotes the cardinality of the parity bits P_1. The encoder then sends the parity bits P_1 and the first qk bits of the information bits to the communication

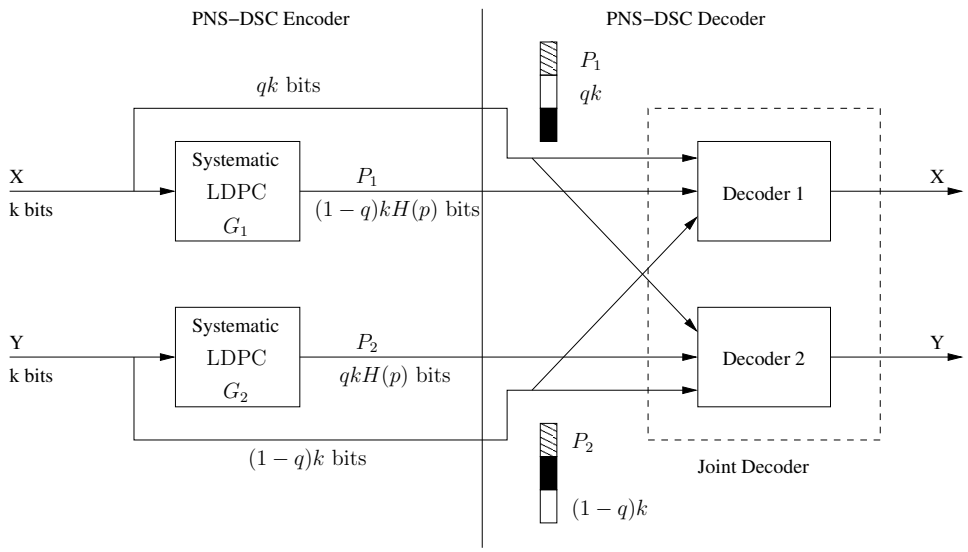


Figure 6.2: Architecture of Parity-based Nonuniform Symmetric Distributed Source Coding Scheme

channel. Thus, the compression rate of X is $R_X = \frac{qk+|P_1|}{k}$. Similarly, the source Y can be encoded by a systematic LDPC code of rate $r_Y = \frac{k}{k+|P_2|}$ and the encoder sends the corresponding parity bits P_2 and the complement $(1-q)k$ information bits in Y to the communication channel. The compression rate of Y is $R_Y = \frac{(1-q)k+|P_2|}{k}$. To losslessly recover the source X and Y at the decoder, the cardinality of P_1 and P_2 should be at least $(1-q)kH(p)$ and $qkH(p)$. Namely, $|P_1| = (1-q)kH(p)$ and $|P_2| = qkH(p)$. Thus the compression rates of X and Y are $q + (1-q)H(p)$ and $1 - q + qH(p)$. A single rate-compatible channel code can be used to realize the channel codes for compressing X and Y through puncturing.

The source in PNS-DSC is encoded through the systematic generator matrix and parity bits are sent to the communication channel. The decoder of PNS-DSC includes two independent sub-decoders which is responsible for decoding the source X and Y respectively. Each sub-decoder uses a standard message-passing algorithm to decode the original source based on the received parity bits and partial information bits.

6.2.3 Two-Machine Algorithm

The Two-Machine algorithm realizes any point in the entire rate region by creating two subcodes from a main code. Both the main code and subcodes need to be capacity-approaching codes to avoid practical performance loss.

Fig. 6.1(a) illustrates its encoder architecture. Given the parity check matrix of two

subcodes, encoding is realized by straightforward multiplication of the parity check matrix and the sources. Namely, $S_x = H_1 X$ and $S_y = H_2 Y$. The resulting syndromes, S_x and S_y, are transmitted to the decoder.

Fig. 6.1(c) shows the extended Tanner graph used by the message-passing decoding algorithm. The message-passing algorithm for each single Tanner graph is exactly the same as the typical message-passing algorithm used to decode LDPC codes. The only difference is that messages are also passed between two Tanner graphs to exchange extrinsic information. The exact formulas to calculate the extrinsic information can be found in [159]. A column dropping procedure is proposed to create parity check matrices of subcodes from the parity check matrix of the main code. The column-dropping procedure might not guarantee to generate capacity-approaching subcodes. Further research effort is needed to refine the column dropping procedure [159].

6.2.4 Symmetric SF-ISF Framework

SSIF uses one channel code to achieve an arbitrary rate pair. Its encoder structure is illustrated in Fig. 6.1(b). SSIF requires two encoders using the same parity check matrix, which is the syndrome former of a LDPC code. Let $H = H_1 = H_2$ and its size is $m \times n$. Then the sum rate of two sources is $m + n$. Each encoder transmits its syndrome, m bits, and complementary subset of first $n - m$ bits. Different rates between two sources are achieved by adjusting what subsets of source bits to transmit.

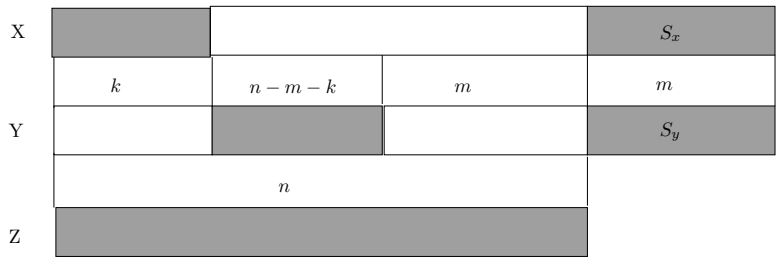


Figure 6.3: Illustration of the Second Decoding Step in SSIF Decoder. The gray areas represent the bits known before the second step decoding.

The decoder in SSIF proceeds in two steps. Given two binary sources, X and Y. Let Z be the difference patten of X and Y. Namely $Z = X \oplus Y$. In the first step, the syndrome corresponding to Z can be obtained through $S_z = S_x \oplus S_y$. Then S_z is passed through a inverse syndrome former, which is H^{-1} for LDPC codes, to get the noise codeword of the difference patten, Z. Z can be recovered after passing the noise codeword into a channel decoder corresponding to H. In the second step, with the knowledge of the difference

pattern, Z, X_{k+1}^{n-m} and Y_1^k can be recovered through the following equations:

$$X_{k+1}^{n-m} = Y_{k+1}^{n-m} \oplus Z_{k+1}^{n-m} \tag{6.1}$$

$$Y_1^k = X_1^k \oplus Z_1^k. \tag{6.2}$$

To decipher the rest of bits, SSIF partitions H into two sub-matrices:

$$H_{m\oplus n} = [A_{m\oplus(n-m)}, B_{m\oplus m}]$$

where B is square matrix and must have full rank. Since

$$S_x = HX = [A, B] \begin{bmatrix} X_1^{n-m} \\ X_{n-m+1}^n \end{bmatrix} = AX_1^{n-m} \oplus BX_{n-m+1}^n$$

we can obtain the remaining m source bits using

$$X_{n-m+1}^n = B^{-1}(S_x \oplus AX_1^{n-m})$$

After recovering all bits of X, the m remaining bits of Y can be recovered through $Y_{n-m+1}^n = X_{n-m+1}^n \oplus Z_{n-m+1}^n$. Fig. 6.3 illustrates the second decoding step.

From the decoding process of SSIF, it is obvious that the performance gap between SSIF and the theoretical limit solely depends on how well the channel code performs on the equivalent virtual correlation channel between two sources. In addition, SSIF imposes stringent requirement on bit correspondence at the encoder. The bit mismatch problem will cause the decoder to fail.

6.3 Syndrome-based Non-uniform Symmetric Slepian-Wolf Code

In this section, we propose a novel approach to realize symmetric Slepian-Wolf coding. We call the proposed approach the Syndrome-based Nonuniform Symmetric Slepian-Wolf Coding scheme (SNS-SWC). It is inspired by PNS-DSC [158]. The basic idea of the proposed approach is to design two nonuniform LDPC codes [163] and let each source transmit a complement set of source bits and syndrome bits. Its architecture is illustrated in Fig. 6.4.

Given two correlated n-bit sources, X and Y, Slepian-Wolf theorem dictates that compression rates $R_X \geq H(X|Y)$, $R_Y \geq H(Y|X)$ and $R_X + R_Y \geq H(X,Y)$. To facilitate exposition of the basic idea, the virtual correlation channel between two sources is assumed to be a BSC (Binary Symmetric Channel). In this case, $R_X \geq H(p)$, $R_Y \geq H(p)$ and

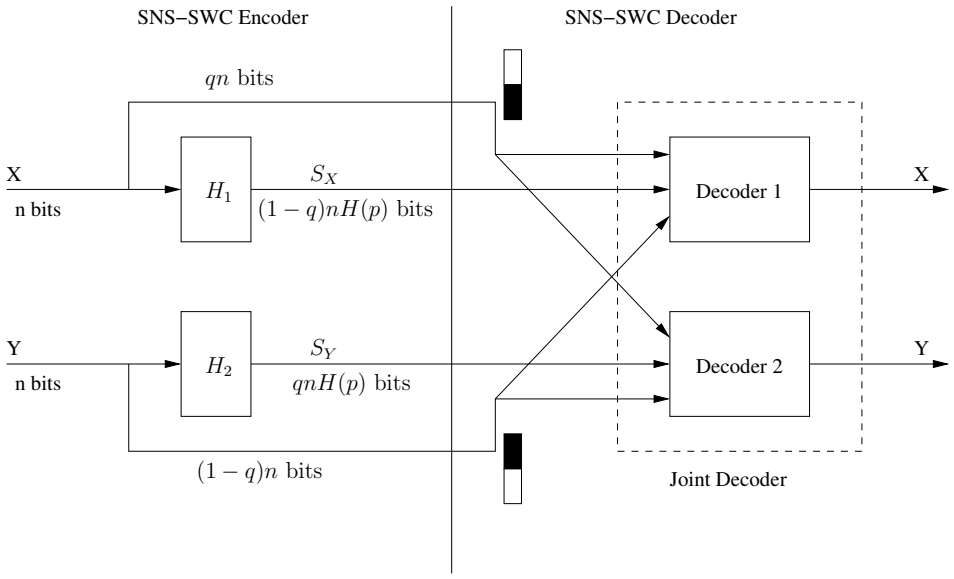


Figure 6.4: The Architecture of the Syndrome-based Nonuniform Symmetric Slepian-Wolf Coding Scheme

$R_X + R_Y \geq 1 + H(p)$, where p is the crossover probability $P(X \neq Y|X)$. We intend to design a symmetric Slepian-Wolf coding scheme to achieve the following compression rates:

$$R_X = \frac{n_X + (n - n_X)H(p)}{n} \tag{6.3}$$

$$R_Y = \frac{n - n_X + n_X H(p)}{n} \tag{6.4}$$

where n_X is the number of source bits directly transmitted by source X. Let $q = \frac{n_X}{n}$ represent the fraction of transmitted source bits. Then we have

$$R_X = q + (1 - q)H(p) \tag{6.5}$$

$$R_Y = 1 - q + qH(p). \tag{6.6}$$

Thus the rate of LDPC codes, r_X and r_Y, that realize the proposed symmetric Slepian-Wolf

coding scheme should satisfy the following constraints:

$$
\begin{aligned}
r_X &= \frac{k_X}{n} && (6.7)\\
&= \frac{n-(n-n_X)H(p)}{n} && (6.8)\\
&= 1-(1-q)H(p) && (6.9)\\
r_Y &= \frac{k_Y}{n} && (6.10)\\
&= \frac{n-n_X H(p)}{n} && (6.11)\\
&= 1-qH(p) && (6.12)
\end{aligned}
$$

where k_X and k_Y is the number of information bits of LDPC codes used to compress source X and source Y. Obviously, as q varies from 0 to 1, the compression rates of two sources, R_X and R_Y, change in the range $[H(p)\ 1+H(p)]$. Namely, the scheme can achieve an arbitrary rate in the Slepian-Wolf rate region. In the case of $q = 1/2$, the encoder uses the equal rate to compress source X and Y, we have $R_X = R_Y = \frac{1+H(p)}{2}$ and $r_X = r_Y = 1 - \frac{H(p)}{2}$.

6.3.1 LDPC Code Design

The LDPC code used in SNS-SWC can be deemed as an LDPC code for two parallel channels as shown in Fig. 6.5, where one part of the code bits are transmitted to the decoder losslessly through a perfect channel and while the other part of code bits are not transmitted, their information can be inferred from the perfectly available complement bits of the other source based on the virtual correlation channel between them. Let $Z_i, i = 1, 2$, denote the random variable that is equal to the log-likelihood ratio (LLR) of a received bit from the ith channel. The LLR distribution of a received bit can be represented by a single channel by using the two channels. Namely

$$P_{Z_X}(z) = qP_{Z_1}(z) + (1-q)P_{Z_2}(z) \quad (6.13)$$

$$P_{Z_Y}(z) = (1-q)P_{Z_1}(z) + qP_{Z_2}(z) \quad (6.14)$$

where Z_1 is the perfect channel and Z_2 is the BSC.

The proposed LDPC code has similar format to PNS-DSC [158]. They both are LDPC codes for a nonuniform channel [163] which are composed of two parallel channels. Therefore, the proposed LDPC code profile can be designed using either the classical single-channel density evolution algorithm [53] or the density evolution algorithm of nonuniform channels in [163]. The difference between the two approaches is that our approach is a syndrome-based approach and treats the n source bits as the code bits and uses the parity

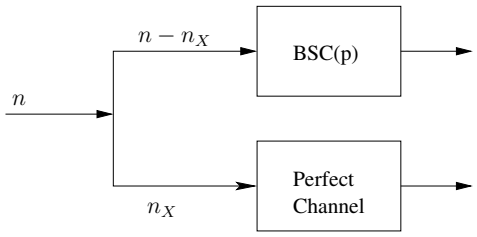


Figure 6.5: The Illustration of Two Parallel Channels

check matrix to encode the sources while PNS-DSC [158] is a parity-based approach and treats the n source bits as the information bits and uses the systematic generator matrix to encode the sources.

6.3.2 Encoding

The left part of Fig. 6.4 illustrates the encoder structure of the proposed symmetric Slepian-Wolf code. Each encoder transmits a complement set of source bits and the corresponding syndrome bits. Different rates of two sources are achieved by adjusting the proportion of source bits to be perfectly transmitted.

6.3.3 Decoding

The decoder structure is shown in the right part of Fig. 6.4. The decoder includes a separate LDPC decoder for each source X and Y. The architecture makes it possible to avoid dependent decoding. Hence, it prevents error propagation that will happen in previous syndrome-based schemes [156, 155, 159, 162]. The decoder of X receives the first part of the source bits perfectly. To reconstruct the whole sequence, the decoder tries to recover the remaining bits based on the part of source bits of Y which is perfectly received by the decoder. The decoder considers the missing bits of X as the output of a BSC with the crossover probability p whose input is perfectly available bits of Y. The decoder of Y follows the same algorithm. The decoding algorithm in each LDPC decoder is the standard message passing algorithm. The only difference between the decoding algorithm in SNS-SWC and the original LDPC decoding algorithm is the initialization of LLR values. The initial LLR values of all bits used in the message passing algorithm are set based on the type of channel to which the bit belongs. The bits passing through the perfect channel have their LLR values set as $\pm\infty$. The LLR values of the bits passing through the BSC(p) channel are equal to $\pm log\frac{1-p}{p}$.

6.4 Enhanced Extended Tanner Graph

In this section, we discuss the proposed symmetric distributed source coding approach (EETG) that can handle the bit mismatch problem at encoder. The idea is simple and intuitive. EETG takes advantage of both benefits of the Two-Machine algorithm and SSIF. It simplifies the code design and relaxes the bit correspondence requirement at an encoder. EETG is realized by LDPC codes.

6.4.1 Encoder Design

The design objective is to find a symmetric distributed source coding approach that make it easy to construct a capacity-approaching code and can handle potential bit mismatch problem at the encoder. It turns out that turbo-like iterative decoding is the only option since mapping and inverse mapping operation can be naturally integrated into such a decoder. Because the two-machine algorithm [159] is essentially an iterative decoding algorithm, we thus decide to use the message-passing algorithm in an extended Tanner graph as the decoding algorithm. However, it is difficult to construct good subcodes if we follow the code partitioning philosophy [161]. Inspired by the observation that the work [159, 155] both root back to the code partitioning idea [161] and [156, 155] share the same ingredient to realize the symmetric distributed source coding while [156] does not use code partitioning, we realize that one channel code without partitioning should be able to achieve similar performance when it is used in the iterative message-passing algorithm in an extended Tanner graph. Without much thought, it is evident that a channel code and its permutation equivalent code should be used. It is well known in algebraic coding theory that a channel code and its permutation equivalent code have the same weight distribution and thus same error correction capability. Fig. 6.1(b) illustrates the encoder structure of the proposed approach. H_2 is formed by permutation of the columns of H_1. It is almost the same as the encoder used in [156]. The only difference is that two parity check matrices in EETG are specially designed permutation equivalent matrices while they are the same matrix in [156]. Compared with two same matrices, permutation equivalent parity check matrices make it easy to reduce the the number of short cycles that go across two Tanner graphs and thus can improve the decoding performance. Like SSIF, EETG achieves the entire Slepian-Wolf region by adjusting which subsets of source bits are transmitted.

6.4.2 Decoder Design

Given a capacity-approaching channel code, the key to the decoder design is to construct a good extended Tanner graph to improve the performance of message-passing algorithm. In the case of LDPC codes, given a LDPC code ensemble profile (λ, ρ), the question is how we should choose a parity check matrix and its permutation equivalent matrix to construct a good extended Tanner graph. The naive approach would be to randomly choose a code from the code ensemble as the parity check matrix and get a permutation equivalent matrix by randomly permuting the columns of the known parity check matrix and then construct the extended Tanner graph to decode the sources. Experiment results in Section 6.5 show that this method performs poorly. We propose a simple heuristic to construct a parity check matrix and its permutation equivalent parity check matrix from a given channel code ensemble profile. Algorithm 4 gives the pseudo code to construct the parity check matrix and its permutation equivalent parity check matrix. Fig. 6.6 illustrates the structure of the constructed extended Tanner graph.

We construct the extended Tanner graph by observing the following basic guidelines: (1) reduce as many short cycles as possible; (2) let the transmitted source bits associate with variable nodes with large degree; (3) let the extrinsic information propagate into ambiguous nodes as soon as possible. The heuristic approach described in Algorithm 4 is a specific realization of the above principles. Variable nodes in A consist of the set of information variable nodes since their initial log-likelihood ratio (LLR) is $\pm\infty$ or $\pm log\frac{1-p}{p}$, where p is the crossover probability of the BSC virtual correlation channel, and includes most information about their original bits. Variable nodes in C consist of the set of internal reachable variable nodes since extrinsic information can be directly obtained from variable nodes in A from the same Tanner graph. Variable nodes in D consist of the set of external variable nodes since they can only obtain their extrinsic information initially from the other Tanner graph. The variable nodes in E consists of the set of isolated variable nodes since they only obtain their extrinsic information after all other nodes have their extrinsic information.

The rationale to partition H_1 into A and B is inspired by the second decoding step of SSIF. We let the initially most ambiguous bits, whose initial LLR is 0, associate with B and hope that if all $n - m$ bits in A is known, the rest m bits can be quickly decoded. In addition, the partition makes the asymmetric case naturally fit into the scheme. In the asymmetric case, one node transmits $n-m$ source bits and m syndrome bits in our scheme; the other node transmits m syndrome bits. Since our construction guarantees that $n - m$

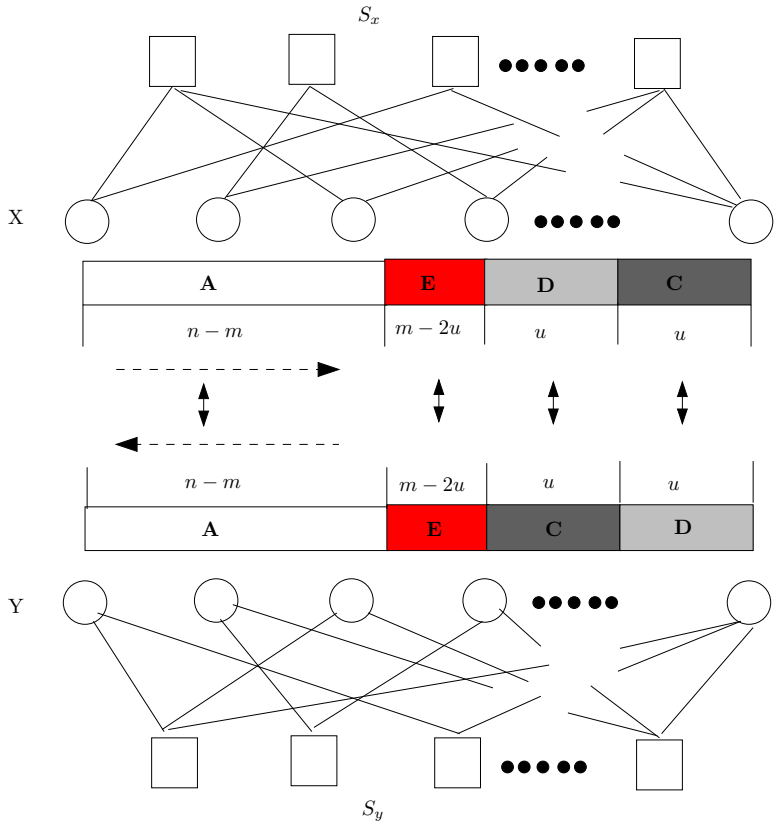


Figure 6.6: Illustration of Permutation Equivalent Parity Check Matrix Construction. A represents the set of information variable nodes. C represents the set of internal reachable variable nodes. D represents the set of external reachable variable nodes. E represents the set of isolated variable nodes. The dashed arrow indicates the degree decreasing direction of variable nodes in A for each matrix (Tanner graph).

Algorithm 4 Pseudo Code to Construct A Parity Check Matrix and Its Permutation Equivalent Parity Check Matrix

λ : variable node degree distribution
ρ : check node degree distribution
H_1 : parity check matrix
H_2 : permutation equivalent parity check matrix
n : the code length
m : the syndrome length

Randomly construct a Tanner graph as H_1 based on the channel code profile (λ, ρ);
Partition H_1 into two matrices $A_{m\times(n-m)}$ and $B_{m\times m}$. Make sure that B is full rank and the degree of variable nodes in B is as small as possible;
Partition B into three matrices $C_{m\times u}$, $D_{m\times u}$, and $E_{m\times(m-2u)}$. Suppose that all $n-m$ variable nodes in A are known, make sure that all u variable nodes in C are successfully decoded under BEC assumption and no variable nodes in D and E can be decoded using message-passing algorithm in the Tanner graph of H_1. Suppose that all $n-m$ variable nodes in A and $2u$ nodes in C and D are known, make sure that all $m-2u$ variable nodes in E can be successfully decoded under BEC assumption using message-passing algorithm in the Tanner graph of H_1;
Sort the columns in A based on variable node degree so that the degree of variable node associated with each column is in non-increasing order from left to right;
$H_1 = [AEDC]$;
Sort the columns in A based on variable node degree so that the degree of variable node associated with each column is in non-decreasing order from left to right;
Randomly permute columns in C D E;
$H_2 = [AECD]$;
Construct the extended Tanner graph based on H_1 and H_2;

bits correspond to A and B is full rank, the n original source bits can be easily decoded by solving m linear equations. The other source can then be decoded using sum-product algorithm in a Tanner graph. The reason that we let the variable nodes in B have low degree is to mitigate the negative effect of the most ambiguous bits on the decoding of other bits. According to the message update rule, shown in Eqn. 2.13, of the sum-product algorithm in check nodes, if the message in one of the neighbor nodes is zero, the output message of the check node is zero. Therefore, if the degree of variable nodes in B is large, the extrinsic information propagation will be slow and some bits are hard to decode. Similarly, our intention to sort the variable nodes in A based on their degree is to let the transmitted source bits, whose LLR is $\pm\infty$, to associate with large degree variable nodes and thus maximize their positive impact to decode other source bits. The exchange of C and D in the parity check matrix and its permutation matrix aims to accelerate the propagation of extrinsic information since variable nodes in C can get extrinsic information directly from internal iteration and variable nodes in D initially have no way to get extrinsic information from internal iterations. The Binary Erasure Channel (BEC) assumption is used in the pseudo code because the extrinsic information propagation from known variable nodes to the ambiguous variable nodes is much like the decoding of erasure bits in a BEC channel.

6.4.3 Handling The Bit Mismatch Problem

Though our approach relaxes the stringent requirement on bit correspondence at the encoder, it still cannot totally eliminate all the problems created by lack of knowledge of bit correspondence at an encoder. The major problem created by the bit mismatch is that there is no way for two encoders to send non-overlapped source bits if the mapping between two sources is arbitrary and unknown at the encoder. Overlapped source bits will induce performance loss at the decoder since less information is available for the decoder to decipher the original source bits.

However, the parity check matrices and the extended Tanner graph can be modified to avoid the sending of overlapping bits by encoders if the mapping is bounded and can be described by the following equation.

$$\pi(i) = \begin{cases} j & j \in [1\ k_1 + m_1], i \in [1\ k_1]; \\ j & j \in [n - k_1 - m_1\ n], i \in [n - k_2\ n]. \end{cases} \tag{6.15}$$

where k_1 and k_2 are length of source bits sent by two separate sources and satisfy $k_1 + k_2 = n - m$, and $0 \leq m_1 \leq m$.

For example, as shown in Fig. 6.7, we can move the matrix B to the middle portion of

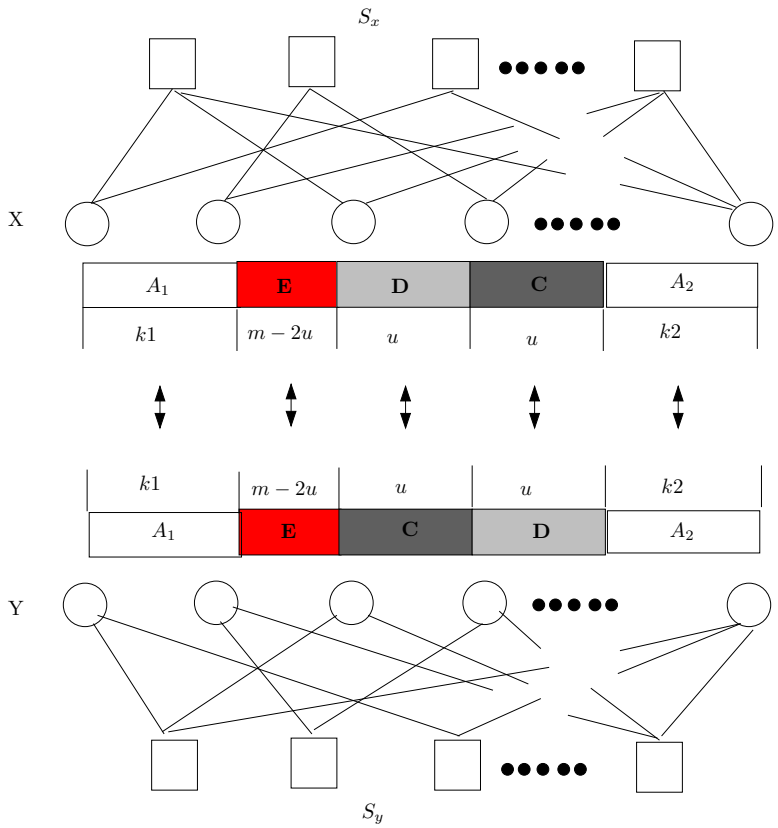


Figure 6.7: Handle The Bit Mismatch Problem

the parity check matrix H and let one encoder transmit the first k_1 source bits and let the other decoder transmit the last $k_2 = n - m - k_1$ bits. Then no bit overlapping will occur. Thus the proposed approach can tolerate some degree of bit mismatch and is more suitable to be used in the real world.

6.5 Simulation Results

In this section, we present simulation results to evaluate the performance of SNS-SWC and EETG. Results on different code length and rate pairs are compared since they have a significant impact on the performance of distributed source coding schemes. Bit Error Rate (BER), which is the ratio between the number of unsuccessfully decoded bits to the code length, is used as a performance metric. In our simulation, two independent and identically distributed (i.i.d) binary sources, X and Y, are generated with a BSC(p) correlation, where p is the switching probability.

We first present the performance evaluation results for SNS-SWC. Based on the ratio of source bits directly transmitted by two sources, the rate of LDPC codes can be calculated using Eqn. 6.9 and Eqn. 6.12. LDPC codes are then designed using the density evolution algorithm [53] for the nonuniform channel that is comprised of a BSC channel and a perfect channel.

We first evaluate the performance of compressing both sources at different rate pairs. Fig. 6.8 gives the log-scale bit error rate (BER) when the code length is 10000. Each data point is an average result with one million simulated bits. Results show that symmetric rate pairs have similar performance to the asymmetric rate pair. It gives the expected performance that the smaller the number of different bits between X and Y and their entropy $H(X, Y)$, the lower the BER. We then compare the performance of SNS-SWC for different code lengths. Fig. 6.9 shows the results for rate pairs $(0.75, 0.75)$ and $(0.8, 0.7)$. The graph gives the expected results that BER decreases as the bit length increases.

We next present the performance evaluation results for EETG. We demonstrate the feasibility and efficiency of EETG using a rate $\frac{1}{2}$ irregular (n, k) LDPC code. The degree distribution pair in example 2 of [53] is used to generate LDPC codes.

We first show the effectiveness of the heuristic structural code construction approach. Fig. 6.10 compares the performance of the code generated by the random construction and code generated by structural construction when the code length is 10000 and rate pair is $(0.75, 0.75)$. Results indicate that the code generated by the random construction has a

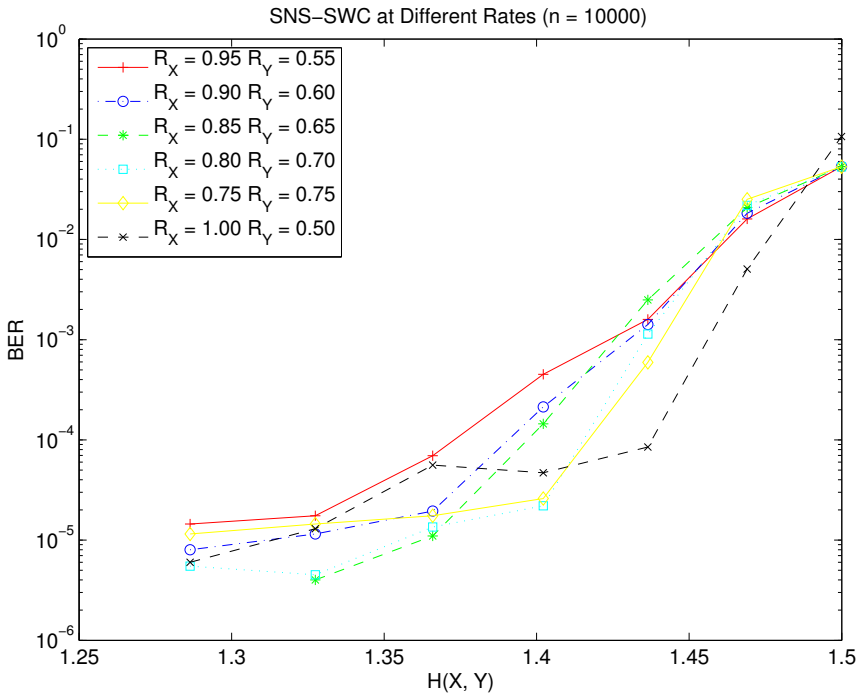
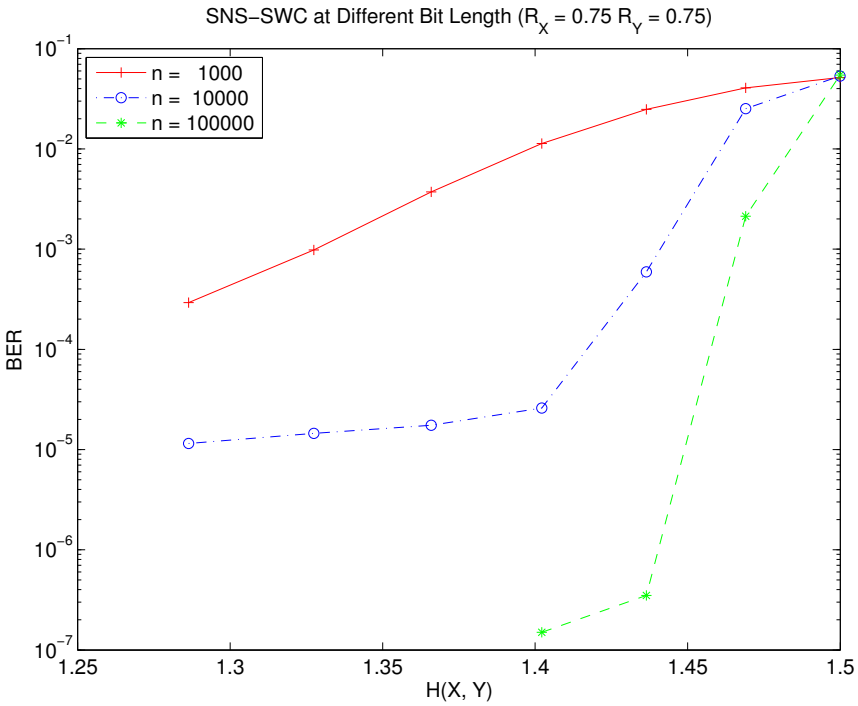


Figure 6.8: Simulation Results under Various Rate Pairs for SNS-SWC. The code length is 10000.

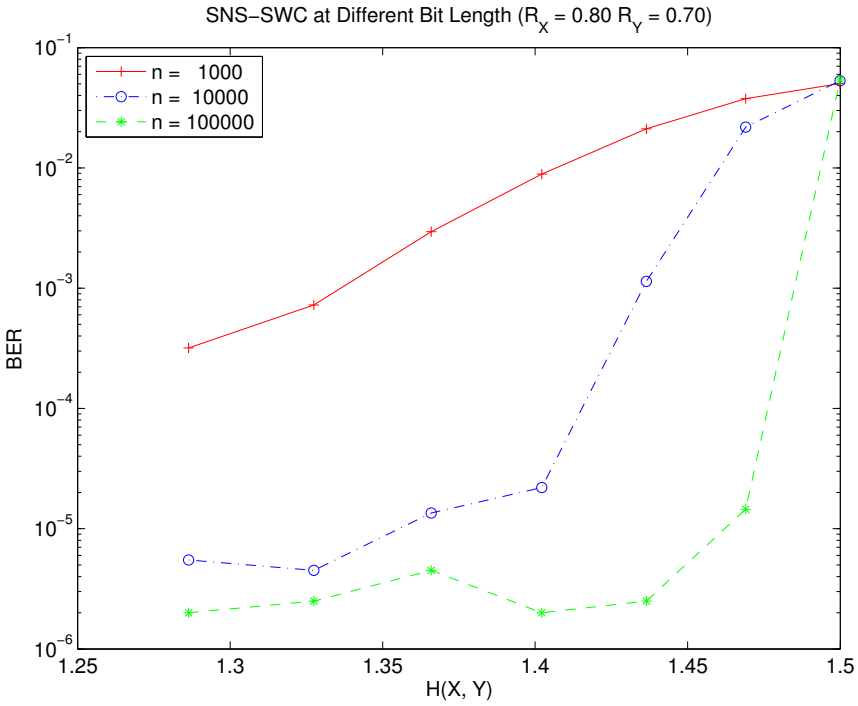
consistent error rates regardless of the joint entropy rate. The code created by structural construction is significantly better than the code generated by random construction. For each data point, one million bits are simulated and the results are averaged. Experiments on other code length and rate pairs have the similar results.

We then study the performance of compressing both sources at different rate pairs. Fig. 6.11 shows the log-scale bit error rate (BER) when the code length is 10000. Results indicate that asymmetric rate pair $(1, 0.5)$ has better performance than other symmetric rate pairs. The reason might be that the iterative message-passing algorithm is sub-optimal in extended Tanner graph. The results of symmetric rate pairs are comparable to other symmetric distributed coding approach using an iterative decoding procedure in extended Tanner graph such as [159, 160].

We also compare the performance of EETG for different code lengths. Fig. 6.12 shows the BER for different code lengths under two symmetric rate setup. Fig. 6.12(a) illustrates the BER for different code lengths when R_X and R_Y are both 0.75. Fig. 6.12(b) gives the BER for different code lengths when R_X is 0.9 and R_Y is 0.6. The results indicate that the performance becomes better when the code length increases. However, when the joint entropy of two sources are greater than a bound, for example, 1.4 in the presented case, the



(a) $R_X = 0.75$ $R_Y = 0.75$



(b) $R_X = 0.8$ $R_Y = 0.7$

Figure 6.9: Simulation Results under Various Code Length for SNS-SWC

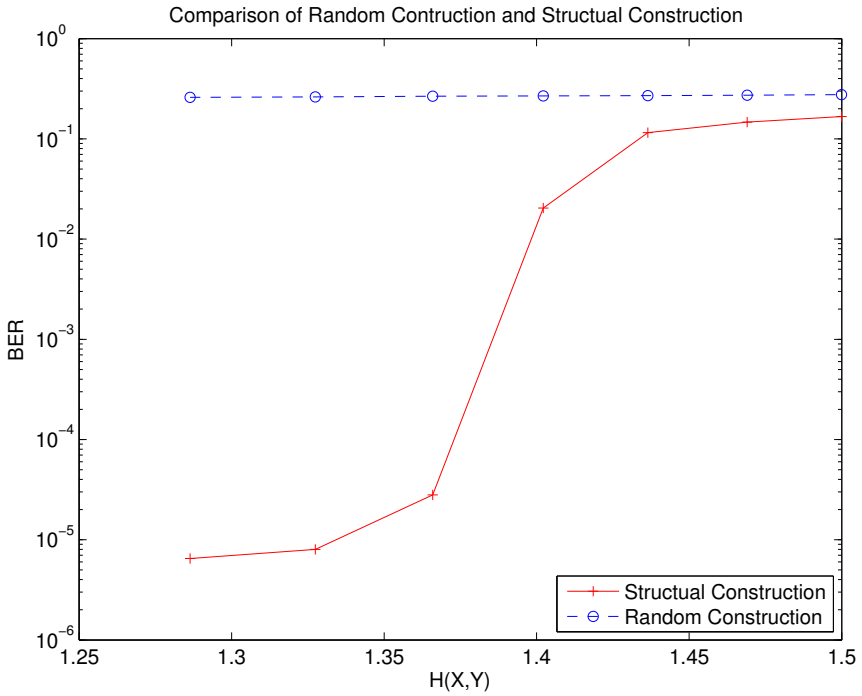


Figure 6.10: Random Code Construction vs. Structural Code Construction for EETG. The code length is 10000. The rate pair is (0.75,0.75).

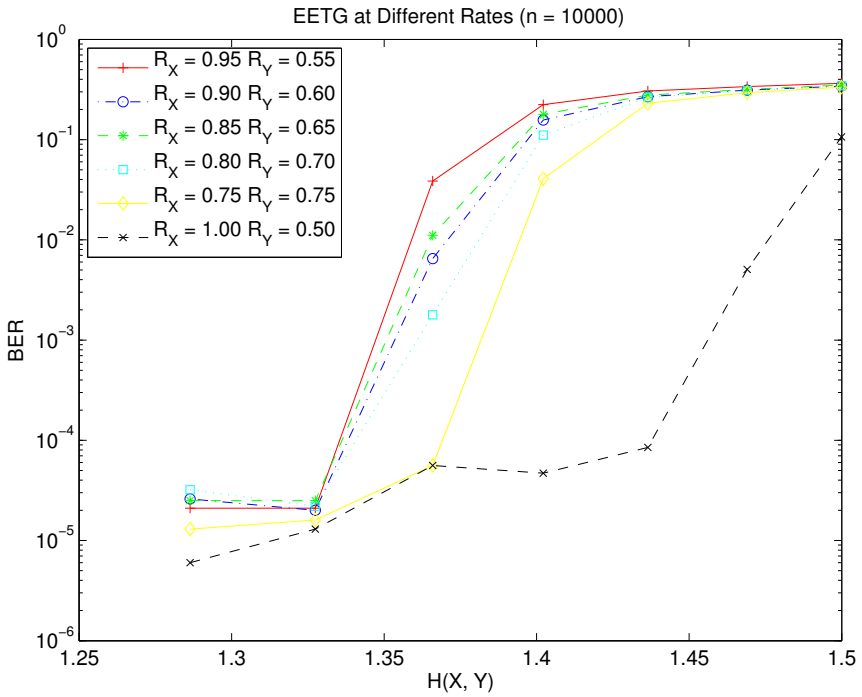
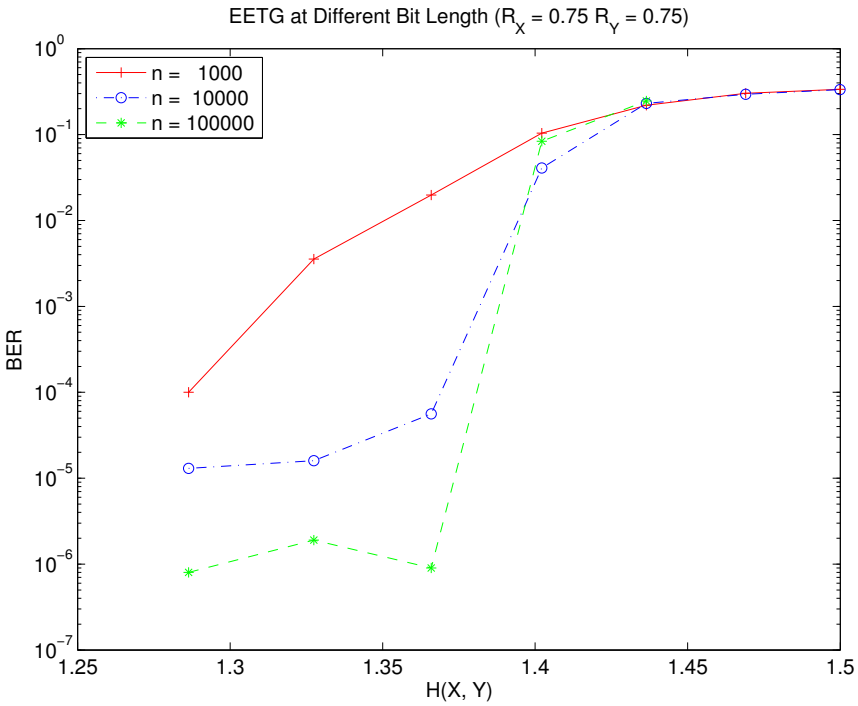
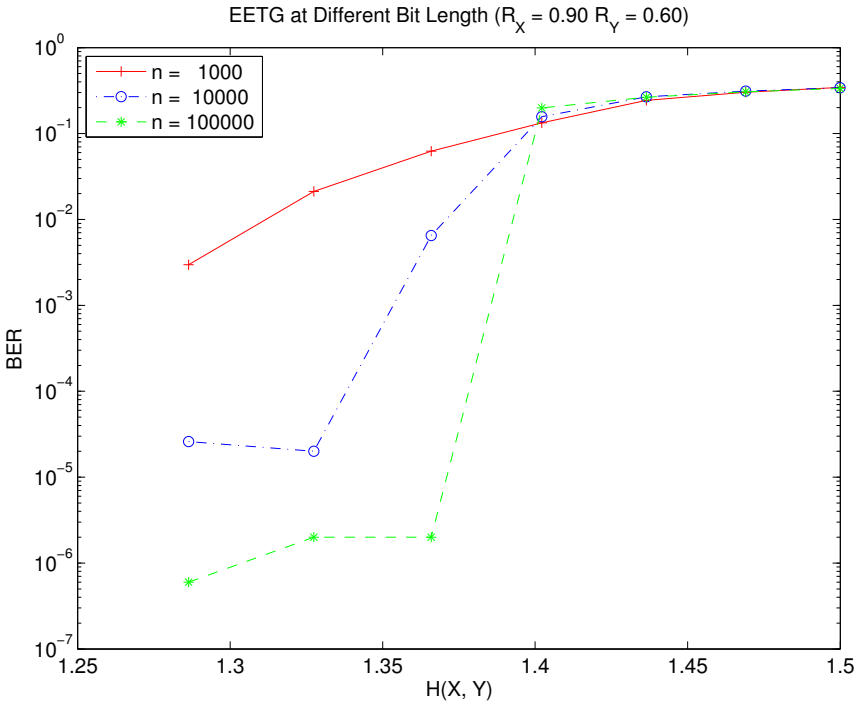


Figure 6.11: Simulation Results under Various Rate Pairs for EETG. The code length is 10000.

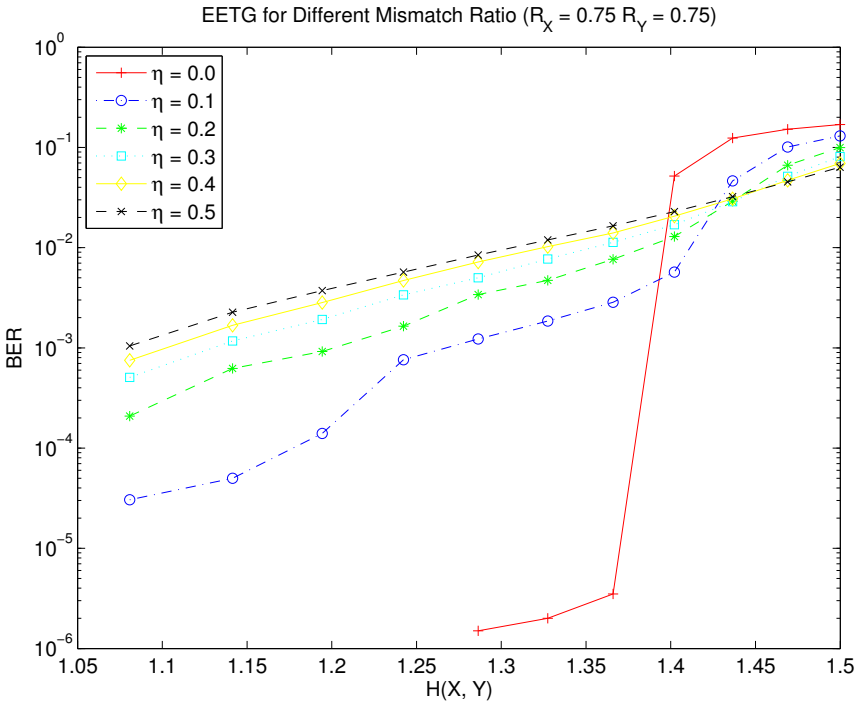


(a) $R_X = 0.75$ $R_Y = 0.75$

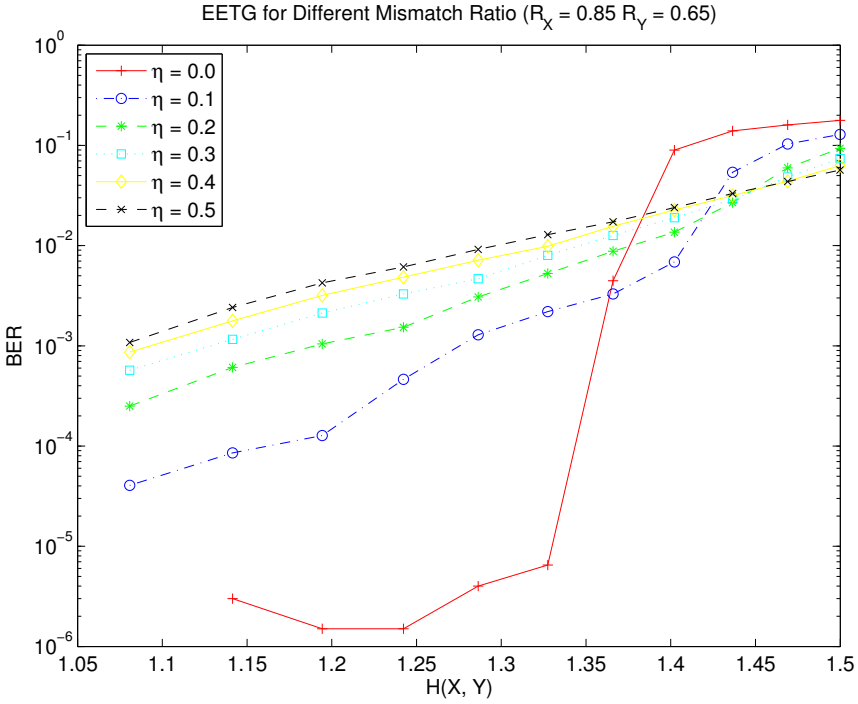


(b) $R_X = 0.9$ $R_Y = 0.6$

Figure 6.12: Simulation Results under Various Code Length for EETG



(a) $R_X = 0.75$ $R_Y = 0.75$



(b) $R_X = 0.85$ $R_Y = 0.65$

Figure 6.13: Simulation Results under Various Mismatch Ratio for EETG. The code length is 10000.

performance is generally poor and is independent of code length. The reason might be that the sum-product decoding algorithm in the extended tanner graph is not powerful enough to losslessly recover the original sources when joint entropy of two sources is near the rate of the chosen channel code.

EETG is able to handle mismatched bits at the encoders by avoiding sending overlapped bits. To evaluate its ability to handle mismatched bits, we use the following equation to specify the mapping between two sources.

$$\pi(i) = \begin{cases} j & j \in [1\ k_1 + \eta m], i \in [1\ k_1]; \\ j & j \in [n - k_2 - \eta m\ n], i \in [n - k_2\ n]. \end{cases} \tag{6.16}$$

where k_1 and k_2 are length of source bits sent by two separate sources and satisfy $k_1 + k_2 = n - m$, and η is the mismatched ratio and is used to measure the degree of mismatched bits between two sources and $0 \leq \eta \leq 0.5$.

Eqn. 6.16 satisfies Eqn. 6.15 and can guarantee that no overlapped bits are sent. Fig. 6.13(a) and Fig. 6.13(b) give the BER for different mismatched ratio when the rate pairs, R_X and R_Y, are $(0.75, 0.75)$ and $(0.85, 0.65)$. The code length is 10000. Results show that there is a consistent performance gap between the case with mismatched bits ($\eta > 0$) and the case without mismatched bits ($\eta = 0$). The reason is that the code design algorithm presented in Algorithm 4 has no way to guarantee that the initial LLRs of variable nodes with large degree are known when there is unknown mapping between two sources. In other words, some variable nodes with large degree become the most ambiguous nodes in the extended Tanner graph. This makes the successful decoding difficult. In addition, the larger the mismatched ratio η, the worse the performance. It is understandable since the larger mismatched ratio makes more variable nodes with large degree become the most ambiguous nodes. In the case of mismatched bits at encoders, EETG still needs to pay the cost of performance degradation for the unknown mapping information between two sources even if it avoids sending overlapped bits.

6.6 Conclusion

In this chapter, we propose two enhanced syndrome-based symmetric distributed source coding schemes that addresses the shortcomings of previous approaches. The idea is simple and effective. Both SNS-SWC and EETG can achieve the whole Slepian-Wolf rate region. SNS-SWC has two independent sub-decoders and thus is able to simplify the decoding process and avoid the error propagation. EETG simplifies the code construction

procedure and relaxes the stringent requirement of bit correspondence at an encoder. A simple code construction heuristic is put forward to construct good extended Tanner graphs used in EETG from a single LDPC code profile. The performance of both SNS-SWC and EETG will get better as the bit length increases. Regardless of the bit length, EETG has a consistent performance gap with the capacity. When there is mismatched bits at encoders, the decoding performance of EETG becomes a little worse.

Chapter 7

Symmetric Distributed Multiview Video Coding

7.1 Introduction

Research in multiview video coding initially focuses on joint multiview video coding (JMVC). JMVC can achieve a high compression ratio. However, JMVC demands communication between cameras to achieve compression. Recently distributed multiview video coding (DMVC), which is rooted in the distributed source coding theory [149], has gained lots of attention. Distributed source coding that exploits the statistics of source signals at the decoder to achieve the compression is essentially different from joint source coding such as methods standardized by MPEG and H.26x which achieves the compression by exploiting the source statistics at the encoder. Practical distributed video coding algorithms have been developed in the past several years and [99, 39] summarize recent advances on distributed video coding. In theory, DMVC holds the promise to achieve the same compression performance as JMVC while demanding no communication between cameras. One particular application scenario for DMVC in MVSN is to reduce the cost of video acquisition subsystem as shown in Fig 7.1. Each video camera directly compresses the video using a DMVC encoder and sends the compressed video to the server for joint decoding. The server can then use a DMVC to MPEG/H.26x based JMVC transcoder to encode the video in the standard format and transmit compressed video over a network to receivers. The configuration could be more cost-effective than the one using multiview video transcoder as illustrated in Fig 4.1. Several distributed multiview video codecs [100, 101, 102, 164, 165, 166] have recently been proposed. However, those approaches are based on asymmetric Slepian-Wolf codes which can not achieve the whole Slepian-Wolf rate region and thus limit the rate allocation options between the encoders.

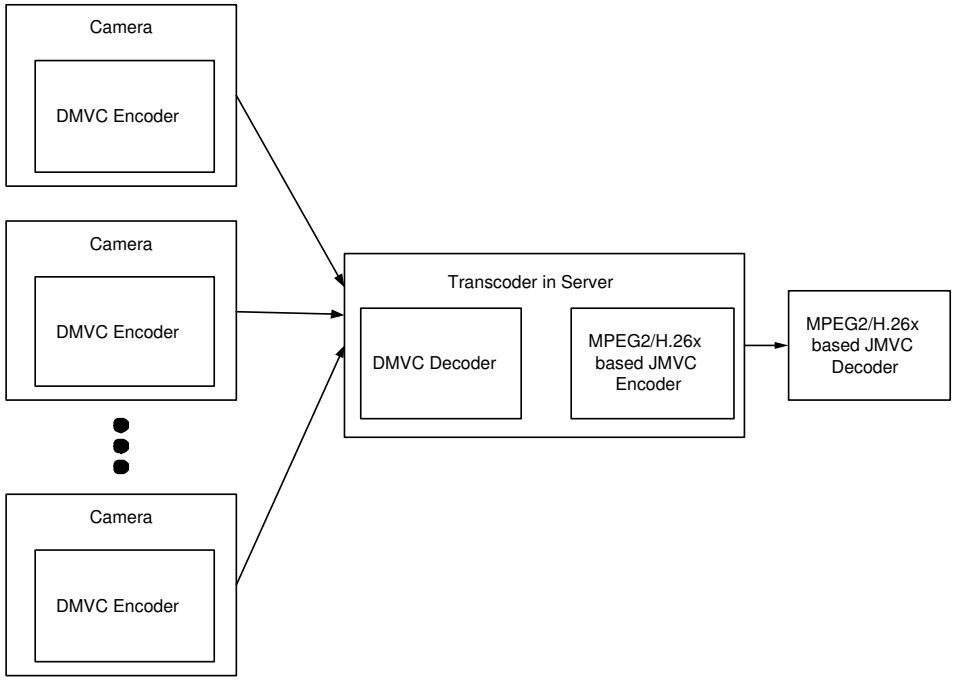


Figure 7.1: Illustration of A DMVC Application Scenario in Video Acquisition Subsystem

Thirumalai et. al [167], Taglisacchi et. al [168] and Yang et.al [105] use source splitting and asymmetric Slepian-Wolf codes to realize symmetric distributed multiview video coding. However, source splitting might incur performance loss. Recently several researchers have designed a capacity-approaching symmetric Slepian-Wolf code which can achieve the whole Slepian-Wolf rate region [155, 156, 158]. Though these solutions are elegant, it is difficult to use them to design symmetric distributed multiview video codec since they implicitly assume that an encoder knows the exact bit correspondence between correlated sources. In the case of distributed multiview video coding, pixel correspondence between two correlated images is not known at the encoder and can only be inferred at the decoder since there is no communication channel between two encoders. Without the assumption of bit correspondence at the encoders, those approaches fail to decode the original source at the decoder. Though the approaches proposed in [159, 162] can handle the bit mismatch problem, it is still elusive to design capacity-approaching codes.

The key to address the pixel mismatch is for the decoder to know the disparity map between two correlated images. There exists two generic approaches to handle the pixel mismatch problem in previous DMVC schemes. The first method is that the decoder estimates a rough disparity map from the rough images transmitted by the encoder [100, 165]. The second approach is to let the decoder learn the disparity in the decoding process using

unsupervised learning algorithm [169, 170]. In this chapter, we put forward a symmetric distributed multiview video codec (SDMVC) that uses the symmetric Slepian-Wolf code SNS-SWC proposed in Section 6.3. It is published in [106]. The scheme estimates the disparity between two correlated images at the decoder and then generates the side information and uses the two independent sub-decoder to recover the original images. Thus, the scheme is able to handle the pixel mismatch problem at the encoder. In addition, the proposed symmetric multiview video codec can realize flexible rate allocation between two encoders and outperform separate H.264 coding of two stereo video sequences.

The rest of chapter is organized as follows. Section 7.2 describes the details of the proposed symmetric distributed multiview video codec. Experimental results are presented in Section 7.3. The chapter concludes in Section 7.4.

7.2 Implementation Details of the Symmetric Distributed Multiview Video Codec

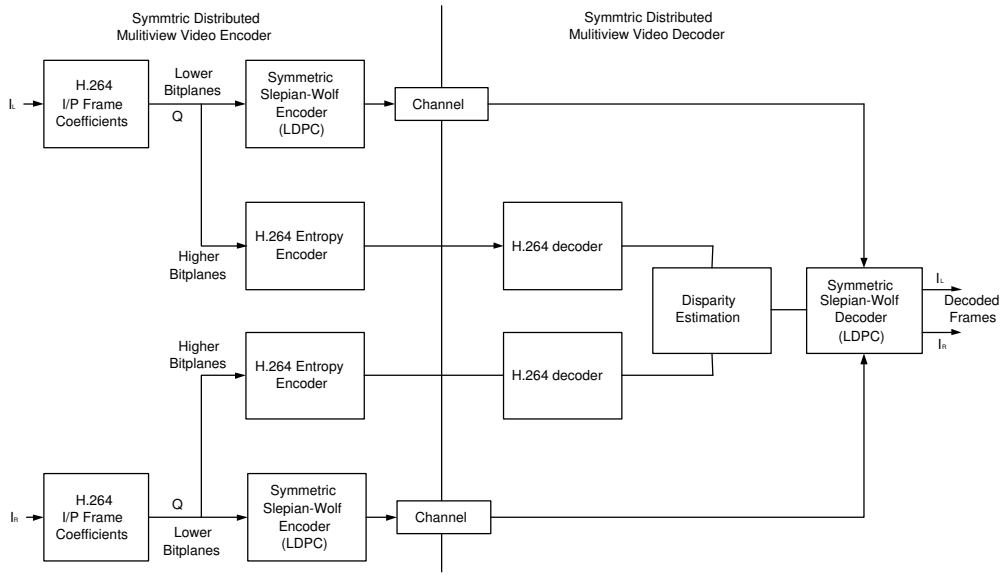


Figure 7.2: The Architecture of The Symmetric Distributed Multiview Video Coding Scheme

DMVC is the application of multiterminal source coding theory on multiview videos. Since the multiterminal source coding theory [43, 44, 171, 172, 173, 45, 46] is not fully developed even for jointly Gaussian sources for the cases with more than two terminals, similar to previous distributed multiview video coding scheme, we also focus on compressing stereo videos. For compressing multiview video with more than two views, it is can

be straightforwardly realized by arranging two neighbor views as a group and using the proposed scheme to distributedly encode them. Though the approach is not a theoretically sound method, it is useful in practice. In this thesis, we concentrate on addressing key challenges to distributedly compress two videos. The architecture of the proposed symmetric distributed multiview video coding (SDMVC) scheme is illustrated in Fig. 7.2. The proposed SDMVC uses the SNS-SWC discussed in Section 6.3 as the underlying distributed source coding scheme. We call our symmetric multiview video coding scheme SDMVC-SNS. Let $I_L = \{I_{L_1}, I_{L_2}, \cdots, I_{L_n}\}$ and $I_R = \{I_{R_1}, I_{R_2}, \cdots, I_{R_n}\}$ be the left and right stereo video sequences, respectively. Pixels in both left and right videos are first processed using standard H.264 algorithms such as intra-prediction or inter-prediction by motion search to get corresponding residual coefficients. Residual coefficients in the frames of both left and right video sequences are first transformed and quantized. Then the higher bit planes of residual DCT coefficients are coded by the H.264 entropy encoder. The lower bit planes of the residual DCT coefficients are compressed by the SNS-SWC introduced in Section 6.3. Since bit planes of a DCT coefficient is used as input source bits in SNS-SWC, bit correspondence is decided by the correspondence of pixels that generate corresponding DCT coefficients. At the decoder, the higher bit planes of DCT coefficients are first used to construct the low quality frames $\hat{I_{L_i}}$ and $\hat{I_{R_i}}, i = 1, 2, \cdots, n$. Then a rough disparity map is estimated from the reconstructed low quality frames. The pixel correspondence can be inferred from the disparity map and then the side information $I^s_{L_i}$ and $I^s_{R_i}$ for I_{L_i} and I_{R_i} ($i = 1, 2, \cdots, n$) can be estimated. Finally the lower bit planes can be decoded by using the side information and the higher bit planes. The proposed scheme can be used to compress residual coefficients in I, P or B frames. We will elaborate on the details of the algorithms below.

7.2.1 Handling The Pixel Mismatch Problem at The Encoder

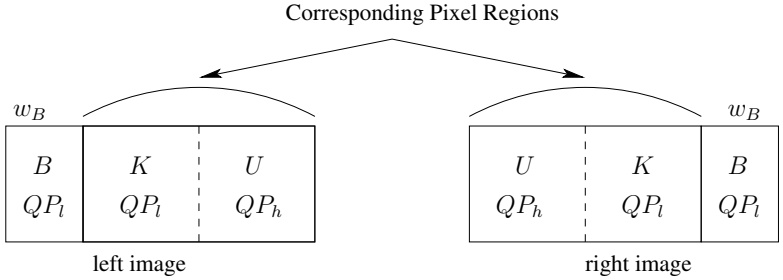


Figure 7.3: Handling The Pixel Mismatch Problem at The Encoder

As we know from the architecture of SNS-SWC elaborated in Section 6.3, the decoding process of SNS-SWC is essentially two separate asymmetric Slepian-Wolf decoding procedures. This property makes it possible for our scheme to handle the bit mismatch problem at the encoder while other approaches such as [156, 155] will fail. The output of the SNS-SWC encoder composes of two parts, syndrome bits and a subset of the source bits. The aim of the SNS-SWC decoder is to recover the unknown part of source bits. The unknown source bits can be recovered correctly if we can successfully estimate the side information and then calculate the correct initial LLR values for those unknown source bits. We use Fig. 7.3 to illustrate how we carefully design our SDMVC-SNS encoder to handle the pixel mismatch problem at the encoder. For simplicity, we assume that there is only horizontal disparity between the left and right image and the maximal disparity is d_{max}, which is known a priori. Pixels in the left or right frame are partitioned into 3 regions: boundary pixel region (B), known pixel region (K) and unknown pixel region (U). Lower bit planes of DCT coefficients in the region K and U are coded by SNS-SWC. Pixels in the region B are not used in Slepian-Wolf coding since they can be occluded with a very high probability and thus can not find corresponding pixels in the other image. The width of the region B, w_B, should be greater than d_{max}. In addition, since a macroblock is a basic coding unit in H.264, w_B should be divided by the macroblock width (16 pixels). Since the corresponding pixels for pixels in the region U of an image are located in the region B and K of the other image which are known at the decoder, the side information for pixels in the region U can be estimated. Thus the bit planes for DCT coefficients in region U can be recovered even though the SNS-SWC encoder is oblivious to the pixel correspondence information between the left and right image.

Table 7.1: Percentage of DCT coefficients whose magnitude is less than 4 for Breakdance video

Frame Type	I Frame	P Frame
$QP_l = 4$	87.75%	90.74%
$QP_l = 10$	97.27%	98.49%
$QP_l = 16$	99.31%	99.71%

7.2.2 Multi-Level Bit Plane Slepian-Wolf Coding

For each 4×4 block in H.264, its residual pixel values are transformed by DCT and quantized. The least significant bit planes of quantized coefficients are coded by SNS-SWC. Bit

Table 7.2: Percentage of DCT coefficients whose magnitude is less than 4 for Ballet video

Frame Type	I Frame	P Frame
$QP_l = 4$	92.46%	95.24%
$QP_l = 10$	98.26%	99.63%
$QP_l = 16$	99.41%	99.91%

planes for every frequency in 4×4 block are separately coded. The length of the source bits to be encoded is $n = \frac{H \times (W - d_{max})}{16}$ where W and H is the width and height of an image. The rate allocation between two encoders is controlled by the fraction of source bits are directly transmitted to the decoders. Let γ be the fraction of source bits directly sent by the encoder for I_L. The fraction of source bits sent by the encoder for I_R is $1 - \gamma$. The LDPC code rate can be computed based on the Eqn. 6.9 and Eqn. 6.12. The number of bit planes for quantized DCT coefficients in H.264 is essentially controlled by the quantization parameter (QP). Suppose that k lower bit planes are coded by the SNS-SWC. Quantization step size in H.264 doubles for every increment of 6 in QP [174]. Given two quantization parameters, QP_l and QP_h, let $QP_l = QP_h - 6k$, we can use them to quantize the DCT coefficients and get the lower k bit planes that are encoded by SNS-SWC. Since the magnitudes of a significant portion of quantized coefficients are small as shown in Table 7.1 and Table 7.2, the sign of the coefficients quantized by QP_l is different from the sign of the coefficients quantized by QP_h. Therefore, the sign bit plane also needs to be encoded by SNS-SWC. Namely, $k + 1$ bit planes are encoded by SNS-SWC.

As shown in Fig. 7.3, QP_l is used to quantize pixels in the region B and K and QP_h is used to quantize pixels in the region U. For a 4×4 block, let $c_i, i = 0, \ldots, 15$ denote its original pixel values and v_i^p be its predicted pixel values. The unquantized residual coefficients, R_i, are equal to $c_i - v_i^p$. Thus the quantized residual coefficients, R_i^q, are $Q(DCT(R_i), QP)$, where $Q(\bullet, \bullet)$ is a quantization function and $DCT(\bullet)$ is the discrete cosine transform function. At the decoder the reconstructed residual coefficients, $\hat{R}_i$, are $\hat{R}_i = IDCT(IQ(R_i^q, QP))$, where $IQ(\bullet, \bullet)$ is an inverse quantization function and $IDCT(\bullet)$ is an inverse discrete cosine transform function. The reconstructed pixel values, $\hat{c}_i$, are $\hat{c}_i = v_i^p + \hat{R}_i$. v_i^p can be intra-predicted from the reconstructed neighbor blocks or inter-predicted by blocks in adjacent frames, $\bar{c}$, at the encoder. In particular, if a 4×4 block is in the unknown region U, two versions of reconstructed pixels are computed, $\bar{c}_j^h = v_j^p + IDCT(IQ(R_j^{q_h}))$ and $\bar{c}_j^l = v_j^p + IDCT(IQ(R_j^{q_l}))$. When it is used to intra-predict other blocks in the same frame, $\bar{c}_j^h$ is used; when it is used to inter-predict

other blocks in adjacent frames, $\bar{c}_j^l$ is used. The reason is that the decoder needs to know the same $\bar{c}$ to reconstruct a low quality frame to estimate the disparity map. For the blocks in the unknown region U, the decoder only has the coefficients quantized by QP_h and thus if other blocks in the same frame are predicted by the blocks in the region U, $\bar{c}_j^h$ is only correct pixel value that the decoder is capable to accurately reconstruct. If the blocks in the unknown region U is used to inter-predict blocks in adjacent frames, the higher bit planes is already successfully decoded at the decoder and thus $\bar{c}_j^l$ can be used to reconstruct the blocks in adjacent frames.

A straightforward way to encode the lower bit planes is to first quantize the residual coefficients, R_i, using a higher quantization parameter QP_h and transmit the resulting coefficients, $R_i^{q_h}$, to the decoder. $\hat{R}_i^{q_h} = IDCT(IQ(Q(DCT(R_i), QP_h), QP_h))$ is reconstructed in the decoder and helps estimate the rough disparity map. Then the same coefficient, R_i, is requantized using a lower quantization parameter QP_l and lower bit planes are extracted from the resulting coefficients, $R_i^{q_l}$, and then are encoded by SNS-SWC and transmitted to the decoder. However, there are two major flaws in the above method: (i) Since the quantization step size does not strictly double for every increment of 6 in QP, the bit planes of $R_i^{q_h}$ are not exactly equal to the higher bit planes of $R_i^{q_l}$. Therefore, even though the decoder can successfully decode the lower bit plane of $R_i^{q_l}$, the concatenation of $R_i^{q_h}$ and the lower bit planes at the decoder, let $\hat{R}_i^{q_l}$ denote the value, is very likely not equal to $R_i^{q_l}$ at the encoder. Namely, $\hat{R}_i^{q_l} \neq R_i^{q_l}$ happens with a high probability. Although the problem can be addressed by designing strictly embedded quantizers so that every quantization threshold using QP_h is also a threshold using QP_l as it is proposed in [105], careful quantizer design is needed. (ii) Even if $\hat{R}_i^{q_l}$ at the decoder is equal to $R_i^{q_l}$ at the encoder. The approach is not efficient and leads to sub-optimal compression performance since the lower bit planes of pixels in the known region K need to be separately entropy coded and transmitted to the decoder. Overhead bits are greatly increased.

We use a simple method to address those two problems and encode the lower bit planes. First, for each 4×4 block, the residual coefficients, R_i, are quantized by QP_l to get $R_i^{q_l}$. The lower bit planes of $R_i^{q_l}$ are extracted to calculate the syndrome bits which are transmitted to the decoder. For pixels in the region K, the lower bit planes of $R_i^{q_l}$ need not be separately entropy coded since the $R_i^{q_l}$ for blocks in the region K can be entropy coded by using the standard H.264 entropy coding method. In this way, we avoid using more overhead bits to encode the lower bit planes of pixels in the region K. For pixels in the region U, the lower bit planes of $R_i^{q_l}$ need not be transmitted and are sim-

ply discarded. We can also guarantee that both the encoder and the decoder can recover the same higher bit planes of DCT coefficients in the region U. With the combination of lower bit planes recovered by the SNS-SWC, the decoder can reconstructed the same coefficients as the encoder. The coefficient in the unknown region U is reconstructed using $\tilde{R}_i^{q_h} = IDCT(IQ(bitshift(DCT(R_i), -k), QP_h))$, where $bitshift(\bullet, \bullet)$ is a bit shift function. One critical point worthy of note is that compared with the previous approach, $\tilde{R}_i^{q_h} \neq \hat{R}_i^{q_h}$. Therefore, the reconstructed approximation of the original pixel value, $\tilde{c}_i = v_i^p + \tilde{R}_i^{q_h}$, is unequal to $\hat{c}_i = v_i^p + \hat{R}_i^{q_h}$, which is reconstructed in the previous approach. since the the reconstructed pixel value is used to help estimate the rough disparity map, the quality of disparity map might be affected. However, since $\tilde{R}_i^{q_h}$ is the same as $\hat{R}_i^{q_h}$ in most time, there is not much difference in the disparity estimated by two methods.

7.2.3 Disparity Estimation and Side Information Generation

To handle the bit mismatch problem at the encoder in SNS-SWC, the decoder needs to first find the correspondent pixels for pixels in the region U. Namely the decoder needs to estimate the disparity map for pixels in the region U. Since the correspondent pixels for the pixels in the region U of an image are located in the region K and B of the other image that is known at the decoder, the stereo matching algorithm can be used to estimate the disparity map for pixels in region U and find the correspondent pixels. The region-tree based stereo matching algorithm [175] is used in the implementation to estimate the disparity map between two stereo video sequences. Once we have the disparity map, the side information can be generated by warping the other frame, for example, $I_{L_i}^s = warp(I_{R_i}, D_i)$, where $warp$ is defined in Eqn. 5.11, D_i is the disparity map between two frames, I_{L_i} and I_{R_i}. However, not every pixel has a matched pixel due to occlusion. For occluded pixels, we use their neighbor pixel values to estimate their real pixel values. Denote the predicted frame as $I_{L_i}^p$ and $I_{R_i}^p$. The side information of the DCT coefficients in the region U can be calculated by transforming and quantizing the residual frame $I_{L_i}^s - I_{L_i}^p$ for the left stream or $I_{R_i}^s - I_{R_i}^p$ for the right stream.

7.2.4 LDPC Code Design

The capacity-approaching LDPC codes need to be designed for the virtual correlation channel of bit planes between two correlated coefficients, X and Y. Let b_i^j ($i = 1, 2, \cdots, n$, $j = X, Y$) denote the ith bit plane of X or Y. The virtual correlation channel between the ith bit plane can be modeled by the conditional probability mass function $P(b_i^X | b_{i-1}^X, \cdots, b_1^X, b_0^X, Y)$.

We use a Gaussian channel to approximate the virtual correlation channel between two correlated bit planes. Thus the LDPC code profile for the nonuniform channel that is comprised of a perfect channel and a Gaussian channel is designed by using the Differential Evolution method [53].

7.2.5 SNS-SWC Decoding

To successfully decode the symmetric Slepian-Wolf code and recover the lower bit planes of pixels in region U, we need to estimate the initial LLR value of each bit plane of unknown pixels. The most significant bit planes are decoded first and they are used to help decode the least significant bit planes. Suppose that $b_0^X, b_1^X, \cdots, b_{i-1}^X$ are known $i-1$ higher bit planes. To decode an ith bit plane, we need to calculate the value of $log\frac{P(b_i^X=0|b_{i-1}^X,\cdots,b_1^X,b_0^X,Y)}{P(b_i^X=1|b_{i-1}^X,\cdots,b_1^X,b_0^X,Y)}$. It is calculated by the following equation :

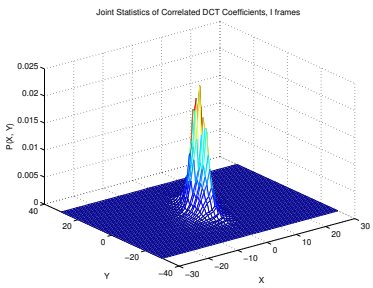
$$log\frac{P(b_i^X=0|b_{i-1}^X,\cdots,b_1^X,b_0^X,Y)}{P(b_i^X=1|b_{i-1}^X,\cdots,b_1^X,b_0^X,Y)} = log\frac{P(b_i^X=0,b_{i-1}^X,\cdots,b_1^X,b_0^X,Y)}{P(b_i^X=1,b_{i-1}^X,\cdots,b_1^X,b_0^X,Y)} \quad (7.1)$$

$$= log\frac{\sum_{b_i^X=b_i^Z=0,b_j^X=b_j^Z,j<i} P(Z,Y)}{\sum_{b_i^X=b_i^Z=1,b_j^X=b_j^Z,j<i} P(Z,Y)} \quad (7.2)$$

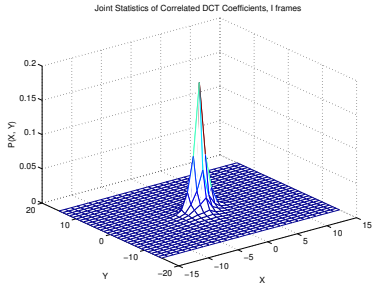
where $P(Z,Y)$ is the joint statistics of correlated DCT coefficients. $P(Z,Y)$ can be estimated based on the joint statistics of previous decoded frames. Fig. 7.4 and Fig. 7.5 give example of estimated joint statistics of correlated coefficients for I frames and P frames.

7.3 Experimental Results

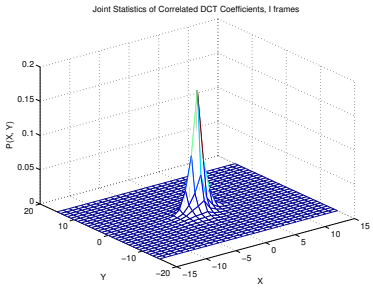
The goal of the proposed symmetric distributed multiview video coding scheme is to have a better rate-distortion performance than separate H.264 coding scheme and achieve flexible rate allocation between two encoders. We conduct various experiments to evaluate the performance of SDMVC-SNS. The Y-components of the 1024x768 Microsoft "Breakdance" and "Ballet" multiview video sequence [3] are used to evaluate the performance of our symmetric distributed multiview video codec. Since the underlying distributed source coding scheme, SNS-SWC, can only achieve the whole Slepian-Wolf region in the case of two sources. Only video streams from two central views, camera 4 and camera 5, are used. The width of the region B, w_B, is set to 64. LDPC code length is thus 46080. We use 10 frames in our experiments. The number of bit planes coded by SNS-SWC is 3, including 2 least



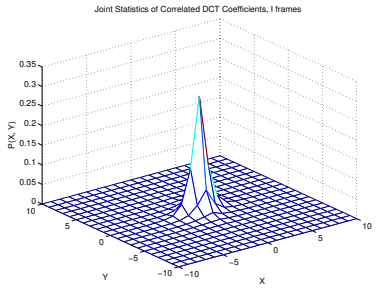
(a) Frequency 0



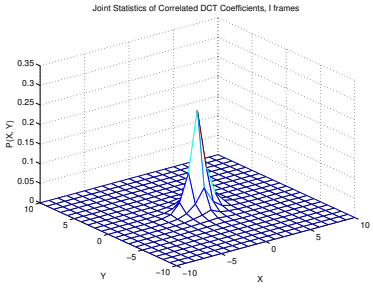
(b) Frequency 1



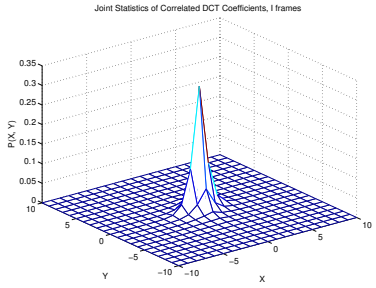
(c) Frequency 2



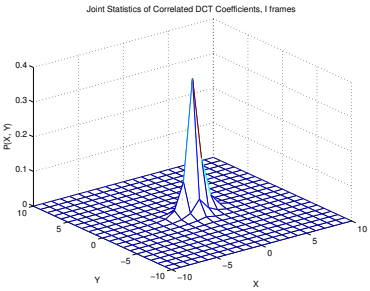
(d) Frequency 3



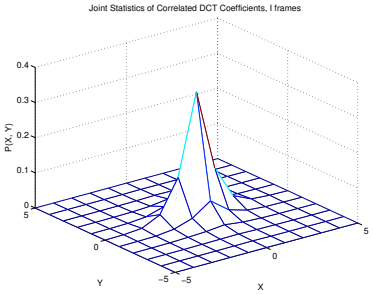
(e) Frequency 4



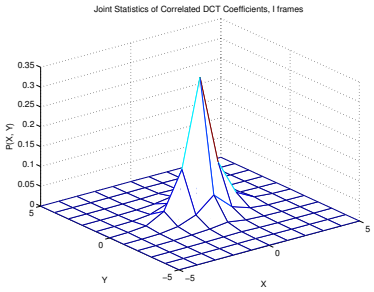
(f) Frequency 5



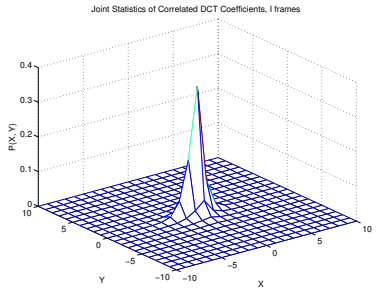
(g) Frequency 6



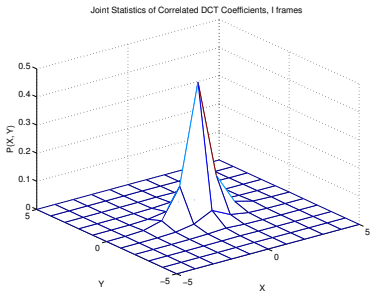
(h) Frequency 7



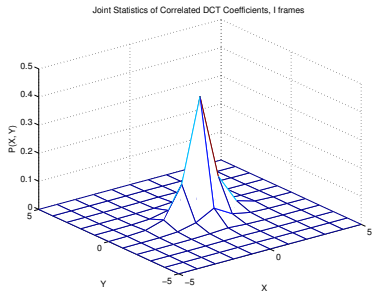
(i) Frequency 8



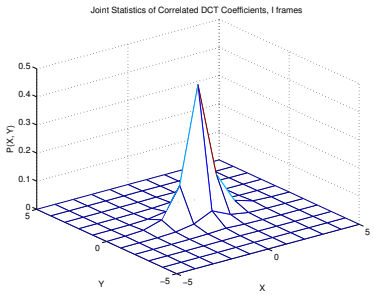
(j) Frequency 9



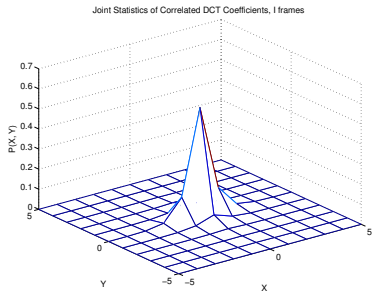
(k) Frequency 10



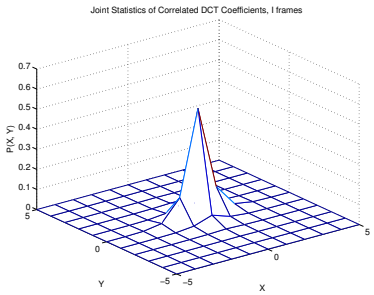
(l) Frequency 11



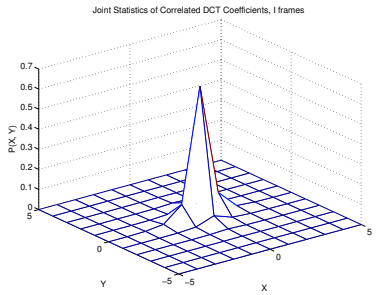
(m) Frequency 12



(n) Frequency 13

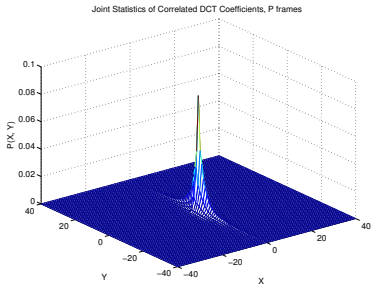


(o) Frequency 14

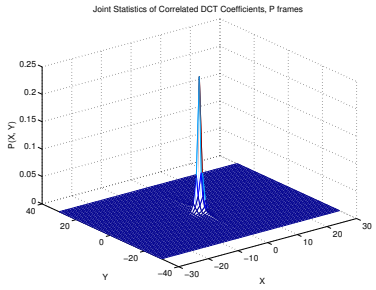


(p) Frequency 15

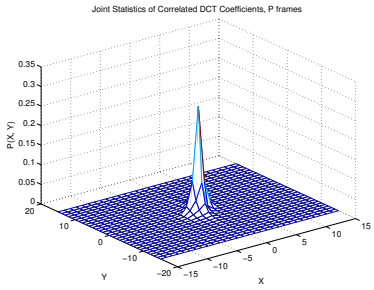
Figure 7.4: Joint Statistics of Correlated Coefficients for Each Frequency in a 4×4 Block of I frames in "Breakdance" Sequence, $QP_l = 10$



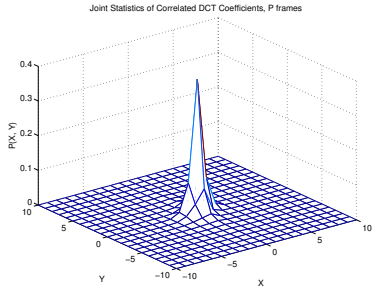
(a) Frequency 0



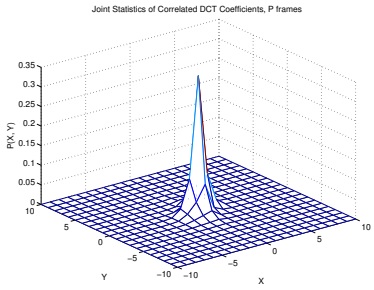
(b) Frequency 1



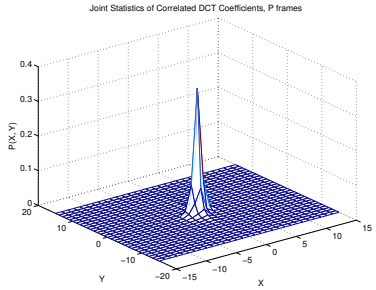
(c) Frequency 2



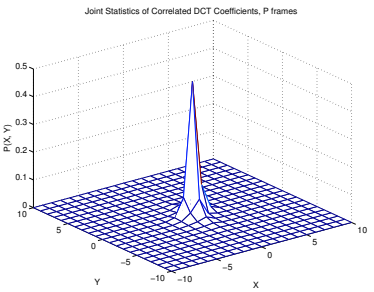
(d) Frequency 3



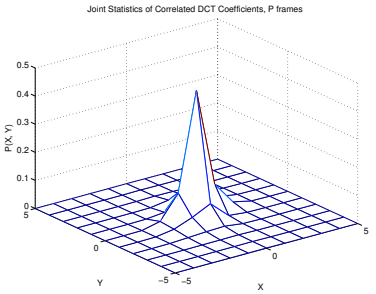
(e) Frequency 4



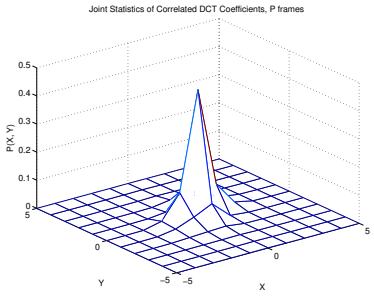
(f) Frequency 5



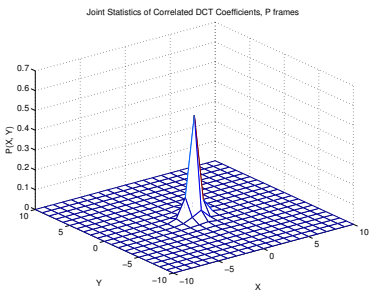
(g) Frequency 6



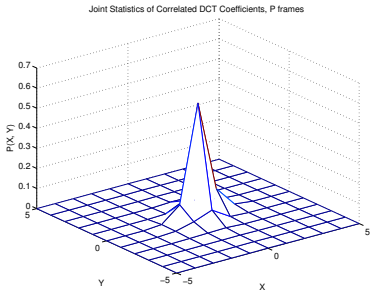
(h) Frequency 7



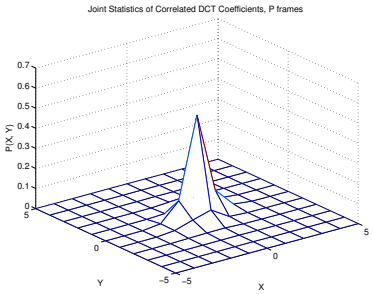
(i) Frequency 8



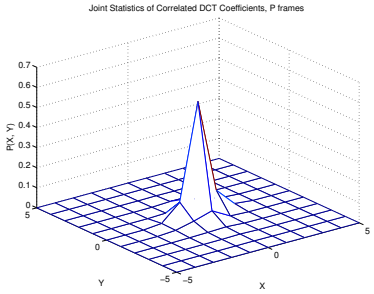
(j) Frequency 9



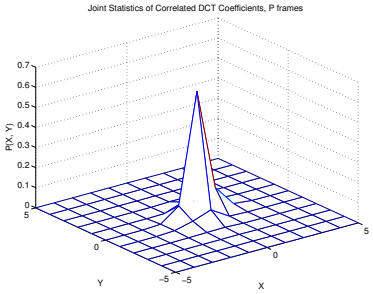
(k) Frequency 10



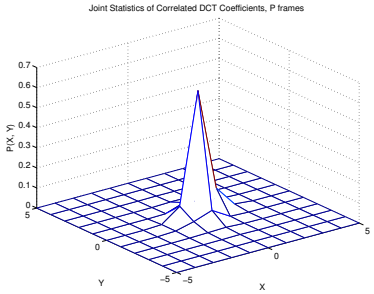
(l) Frequency 11



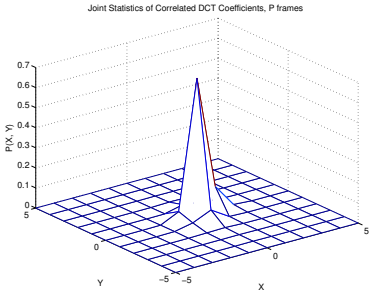
(m) Frequency 12



(n) Frequency 13



(o) Frequency 14



(p) Frequency 15

Figure 7.5: Joint Statistics of Correlated Coefficients for Each Frequency in a 4×4 Block of P frames in "Ballet" Sequence, $QP_l = 10$

significant bit planes of DCT coefficients and the sign bit plane. In the experiments, we vary the rate of two encoders by changing the fraction of source bits, γ, sent by the encoder for camera 4. Two cases, $\gamma = 0.4$ and $\gamma = 0.5$, are considered. Two sequences are coded by two H.264 sequence structures, $II \ldots I$ and $IP \ldots P$. For P frames, all the inter-frame motion search modes are enabled and maximum search range is 16 pixels. III is used to refer the first structure with only I frames. IPP refers to the second structure including both I and P frames.

We first vary the quantization parameters in H.264 to evaluate the rate-distortion performance of the proposed SDMVC-SNS. QP_l is set to 4, 10 and 16 respectively. H.264/AVC reference software JM73 [176] is used in our implementation. Fig. 7.6 and Fig. 7.10 show an example of the original images of camera 4 and camera 5 for Breakdance and Ballet video. Fig. 7.7 and Fig. 7.11 illustrate the reconstructed images when the quantization parameter $QP_l = 10$ and $QP_h = 22$. Fig. 7.8 and Fig. 7.12 gives the corresponding original depth map of Breakdance and Ballet video. Fig. 7.9 and Fig. 7.13 show the estimated rough depth map based on the reconstructed images. The results are compared with separate H.264 coding scheme (H.264). We also compare with the case that the DCT coefficients in the unknown region of frames for camera 4 and 5 are compressed by using LDPC codes optimized for a Gaussian channel. We refer the case as asymmetric Slepian-Wolf Coding (ASWC). ASWC is a hypothetical case and is used to evaluate the performance of LDPC codes designed for SNS-SWC.

Fig. 7.14 and Fig. 7.16 show the rate-distortion comparison for III sequence structure of Breakdance and Ballet video when $\gamma = 0.4$ and $\gamma = 0.5$. Rate is the sum rate of two video streams. Fig. 7.15 and Fig. 7.17 give the rate-distortion comparison for IPP sequence structure of Breakdance and Ballet video when $\gamma = 0.4$ and $\gamma = 0.5$. They indicate that in the high rate case, SDMVC-SNS has a better rate-distortion performance than H.264 and in the low rate case, SDMVC-SNS has an inferior performance than H.264. The higher the bit rate, the better the rate-distortion performance. It is reasonable since most coefficients are not 0 at high rate case and account for a significant portion of total bits used for coding, SDMVC-SNS can thus save many bits by exploiting the joint statistics between correlated coefficients. In the low rate case, most coefficients are 0 and there is not much correlation left for exploitation by SDMVC-SNS. In terms of bit rate saving, in the high rate ($QP_l = 10$, $QP_l = 7$, $QP_l = 4$), SDMVC-SNS with III structure for the Breakdance video can save around 0.6%, 2%, and 1.6% of total bit rate for both rate allocation cases. SDMVC-SNS with IPP structure for the Breakdance video can save around 1.2%, 2.5%, and 2.1% of total

(a)

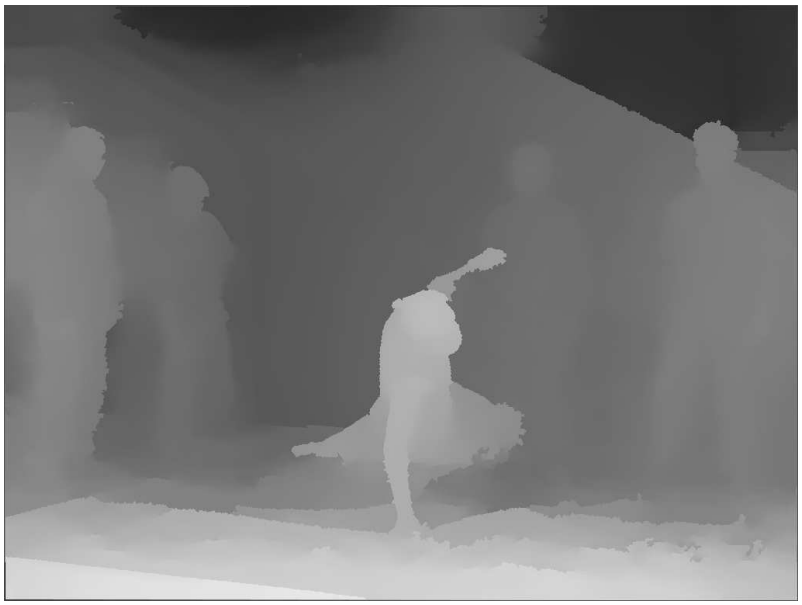
(b)

Figure 7.6: Original Images, Breakdance Video

(a)

(b)

Figure 7.7: Reconstructed Images, Breakdance Video, $QP_l = 10$, $QP_h = 22$



(a)

(b)

Figure 7.8: Original Depth Map, Breakdance Video

(a)

(b)

Figure 7.9: Depth Map, Breakdance Video, $QP_l = 10$, $QP_h = 22$

(a)

(b)

Figure 7.10: Original Images, Ballet Video

(a)

(b)

Figure 7.11: Reconstructed Images, Ballet Video, $QP_l = 10$, $QP_h = 22$

(a)

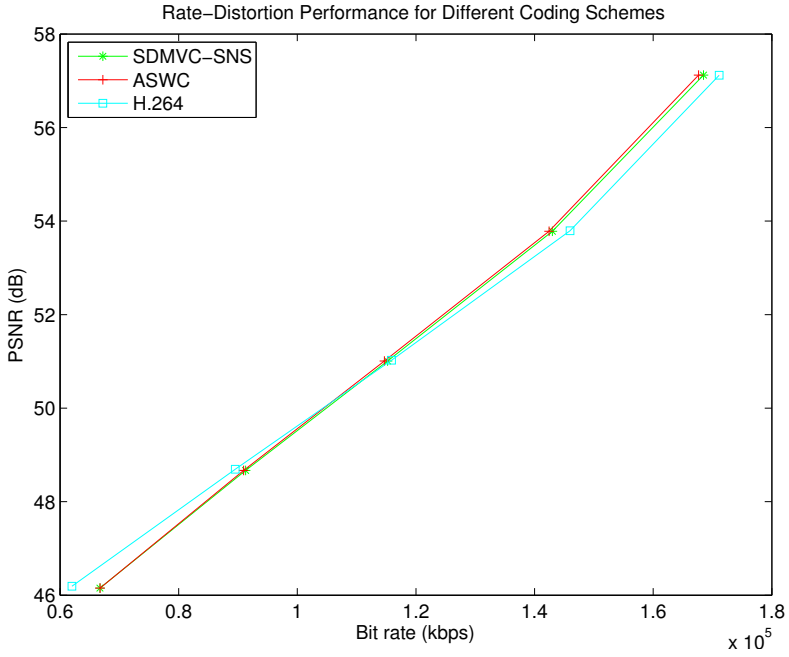
(b)

Figure 7.12: Original Depth Map, Ballet Video

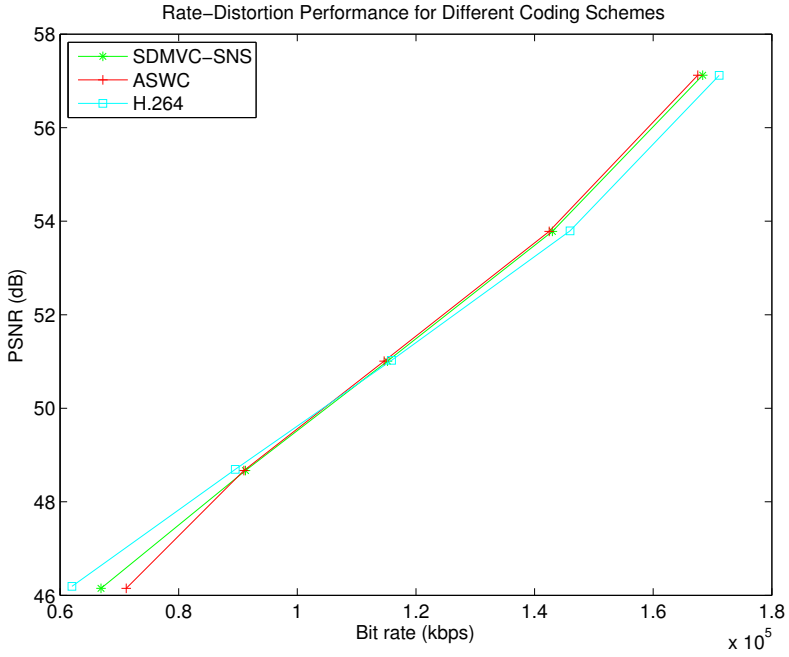
(a)

(b)

Figure 7.13: Depth Map, Ballet Video, $QP_l = 10$, $QP_h = 22$

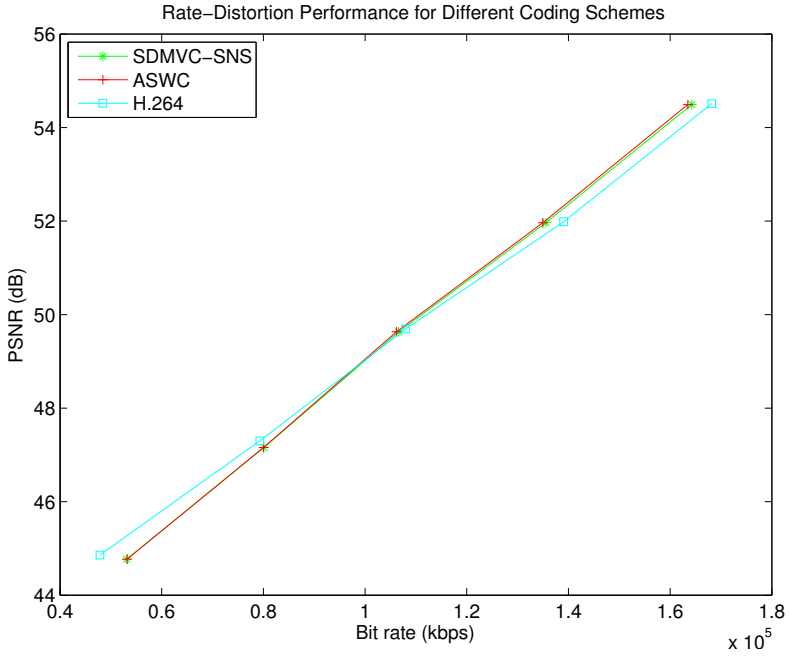


(a) III, $\gamma = 0.4$

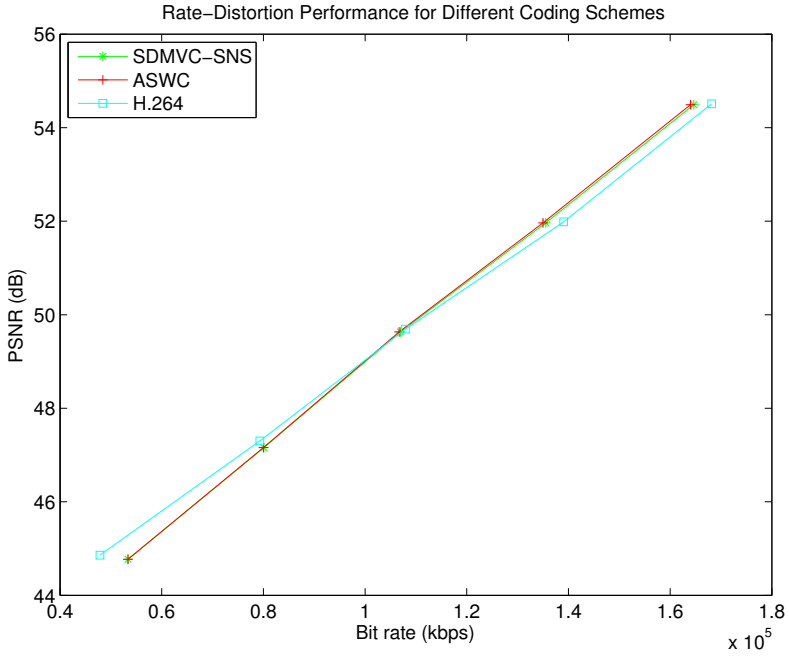


(b) III, $\gamma = 0.5$

Figure 7.14: Rate-Distortion Comparison for Breakdance Video, III

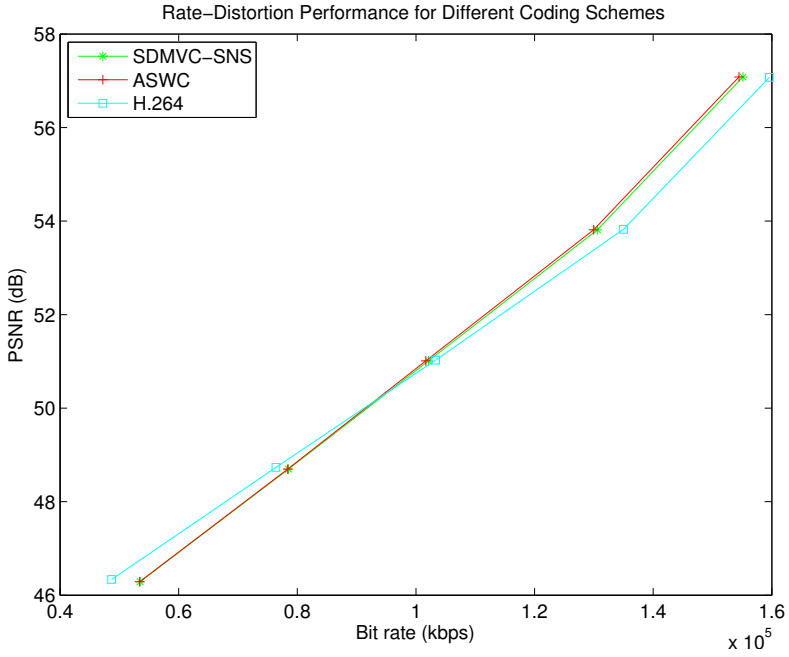


(a) IPP, $\gamma = 0.4$

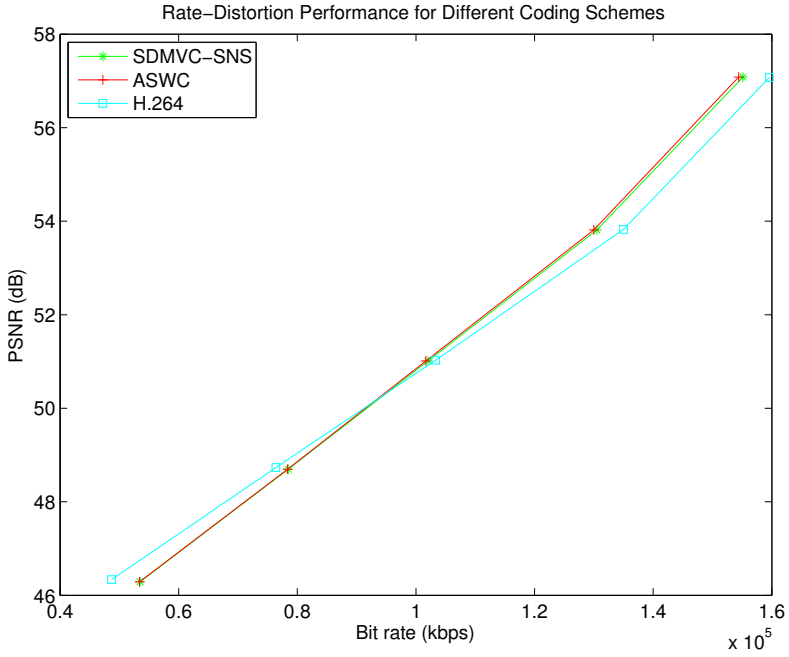


(b) IPP, $\gamma = 0.5$

Figure 7.15: Rate-Distortion Comparison for Breakdance Video, IPP

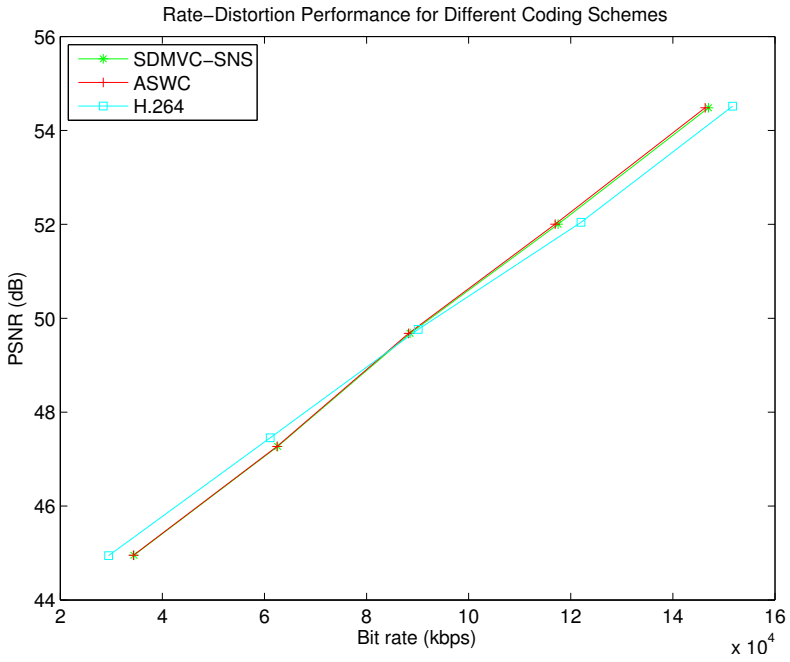


(a) III, $\gamma = 0.4$

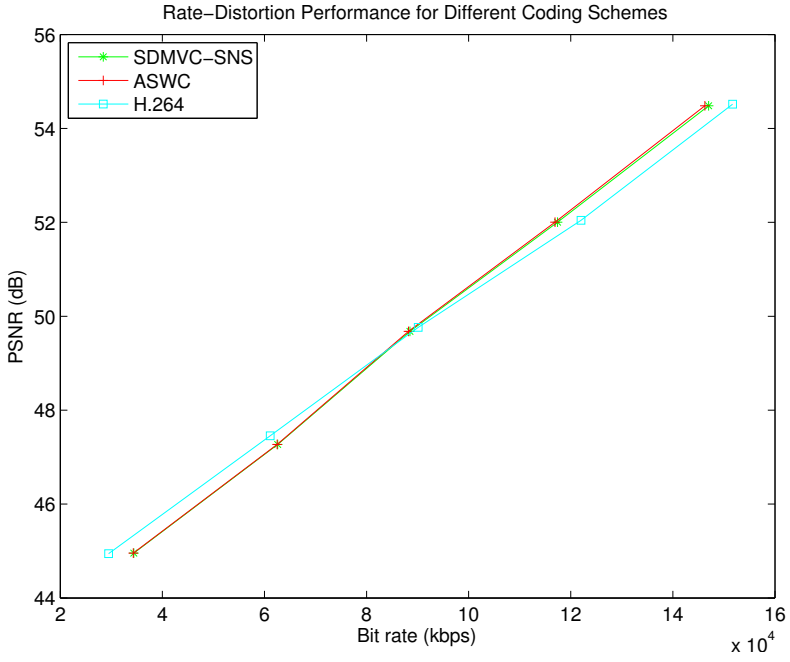


(b) III, $\gamma = 0.5$

Figure 7.16: Rate-Distortion Comparison for Ballet Video, III



(a) IPP, $\gamma = 0.4$



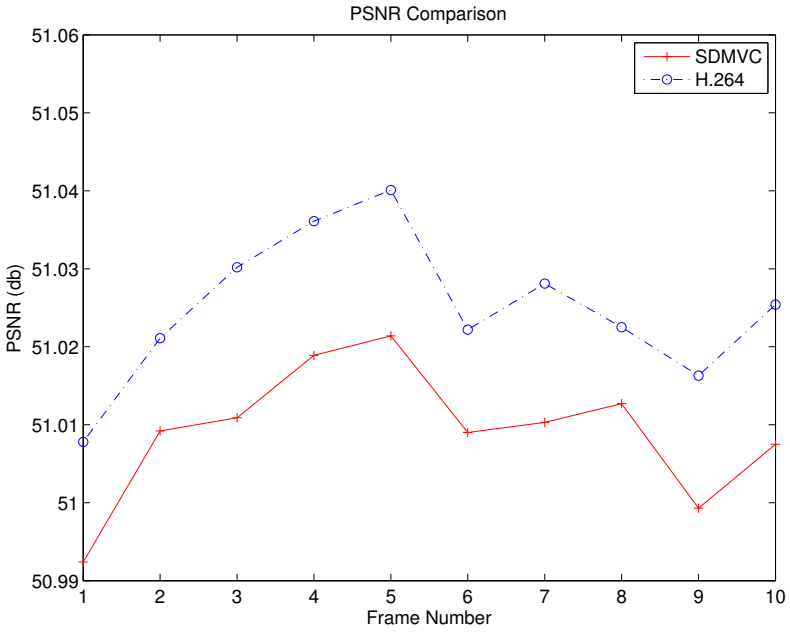
(b) IPP, $\gamma = 0.5$

Figure 7.17: Rate-Distortion Comparison for Ballet Video, IPP

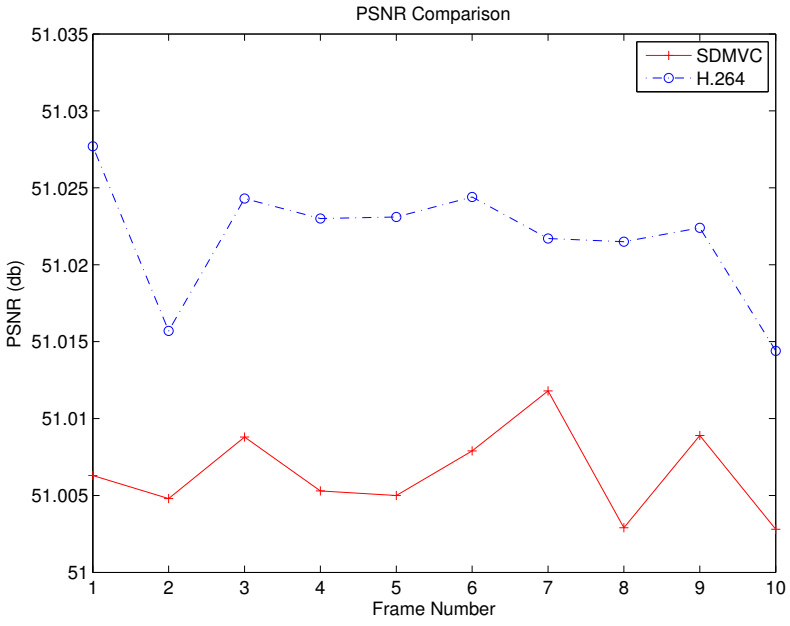
bit rate for both rate allocation cases. Similarly, SDMVC-SNS with III structure for the Ballet video can save around 1.2%, 3.3% and 2.8% of total bit rate for both rate allocation cases. SDMVC-SNS with IPP structure for the Ballet video can save around 1.7%, 3.7%, and 3.1% of total bit rate for both rate allocation cases. The performance of SDMVC-SNS is consistently inferier to the ASWC. ASWC can save about 0.41% total bit rate than SDMVC-SNS in average. This indicates that there is still significant room to improve the LDPC code design.

Fig. 7.18 and Fig. 7.19 give a per frame PSNR and rate comparison between SDMVC-SNS and H.264 for Breakdance video when III structure is used and $\gamma = 0.5$, $QP_l = 10$, $QP_h = 22$. PSNR in SDMVC-SNS has an around 0.01db gap with H.264. The reason is that reconstructed pixels based on QP_h, which is used for prediction of other blocks in the same frame, at 4×4 blocks in the unknown region in SDMVC-SNS has an inferior quality than those based on QP_l at the same blocks in H.264. Fig. 7.19 shows that both camera 4 and camera 5 can save bits. Fig. 7.20 and Fig. 7.21 show a PSNR and rate comparison between SDMVC-SNS and H.264 frame by frame for Ballet video when IPP structure is used and $\gamma = 0.4$, $QP_l = 10$, $QP_h = 22$. PSNR in SDMVC-SNS is also around 0.01db lower than H.264. Both cameras can save bits. However, camera 4 can save more bits than camera 5 since camera 4 has 20% less source bits sent by the encoder and thus more bits can be saved through SNS-SWC than camera 5. This also indicates that rate allocation between two encoders can be realized.

We next study the performance of SNS-SWC on each bit plane and compare it with the performance of arithmetic coding on each bit plane. Fig. 7.22 and Fig. 7.23 give the results for Breakdance video using III structure when $\gamma = 0.4$, $QP_l = 4$, and $QP_h = 16$. Fig. 7.24 and Fig. 7.25 show the result for Ballet video using IPP structure when $\gamma = 0.4$, $QP_l = 4$, and $QP_h = 16$. Bit planes are numbered from least significant bit plane to most significant plane. Namely, first bit plane is the least significant bit plane and second bit plane is the second least significant bit plane. The second bit plane is first decoded and the sign bit plane is last decoded. The figures indicate that the second bit plane has least bit saving and often there is no saving. The reason might be that it is first decoded bit plane and there is not enough side information to successfully decode the original source bits with fewer coded bits. Namely, its conditional entropy is very small or nearly zero and LDPC code is not powerful enough to help save bits. Most of the bit saving comes from the first bit plane. It is decoded with the help of the decoded second bit plane and thus more bits can be saved. The sign bit plane is last decoded and can use both decoded bit planes as side information.

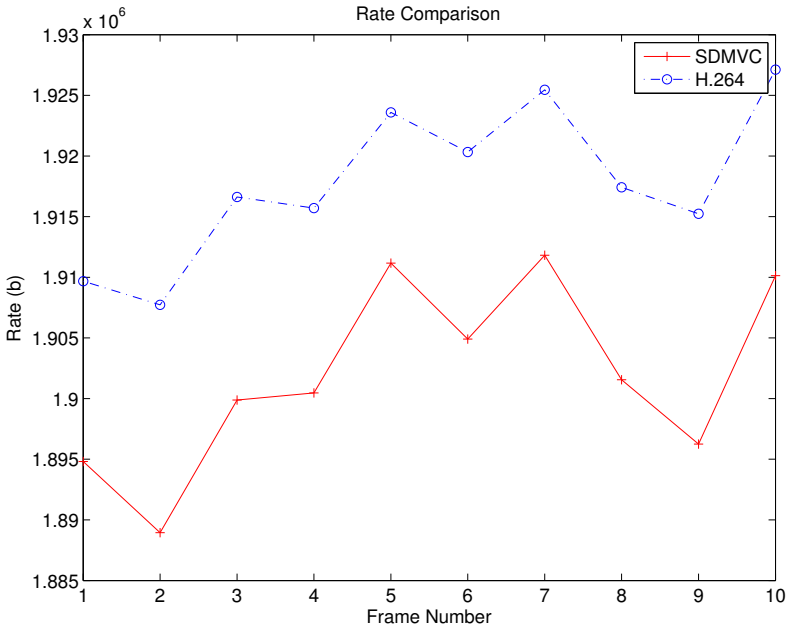


(a) Camera 4

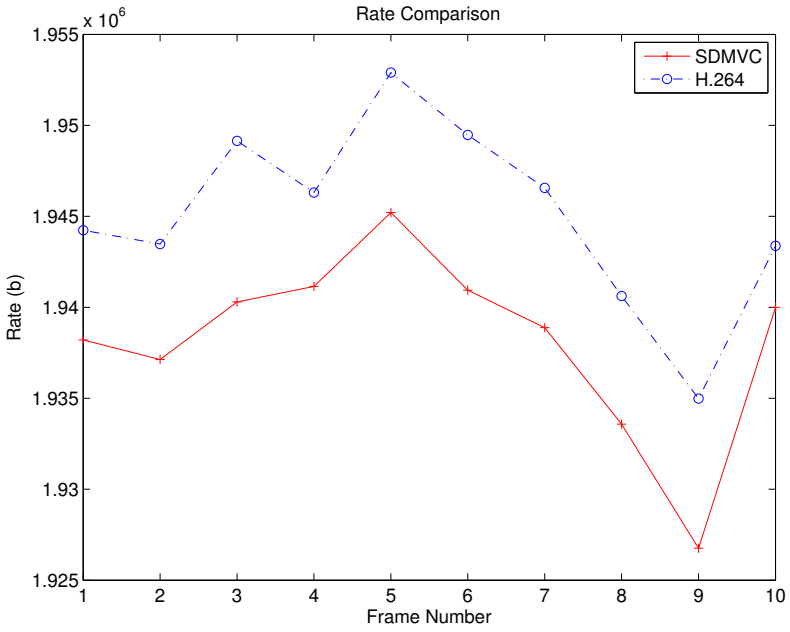


(b) Camera 5

Figure 7.18: Per Frame PSNR Comparison for Breakdance Video, III, $\gamma = 0.5$, $QP_l = 10$, $QP_h = 22$

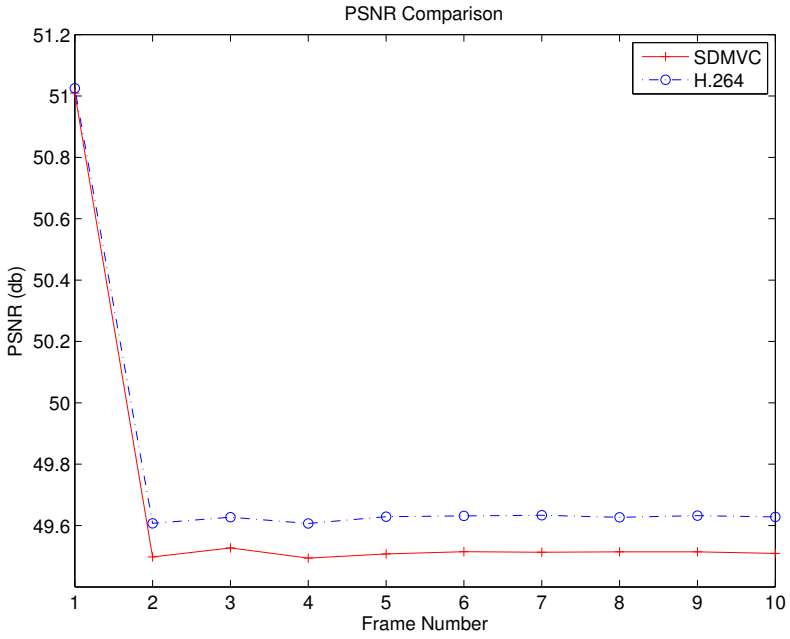


(a) Camera 4

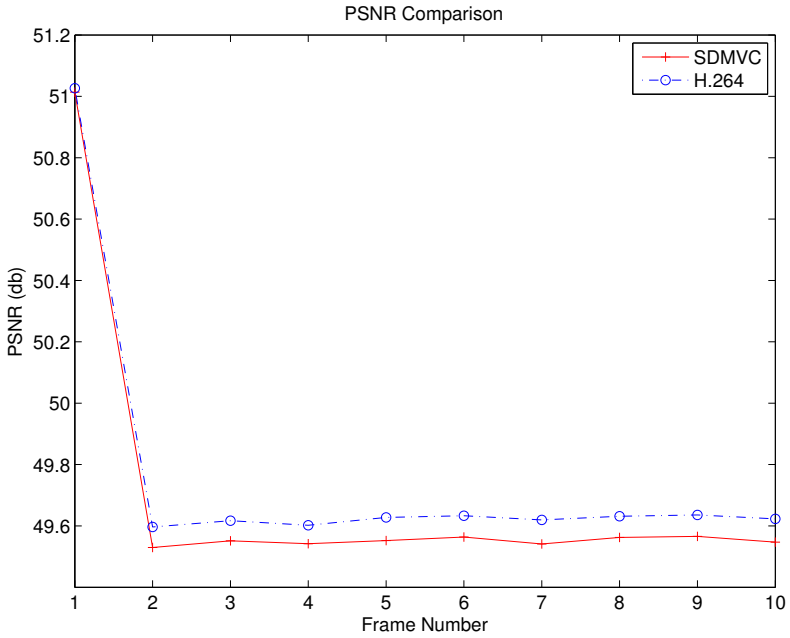


(b) Camera 5

Figure 7.19: Per Frame Rate Comparison for Breakdance Video, III, $\gamma = 0.5$, $QP_l = 10$, $QP_h = 22$

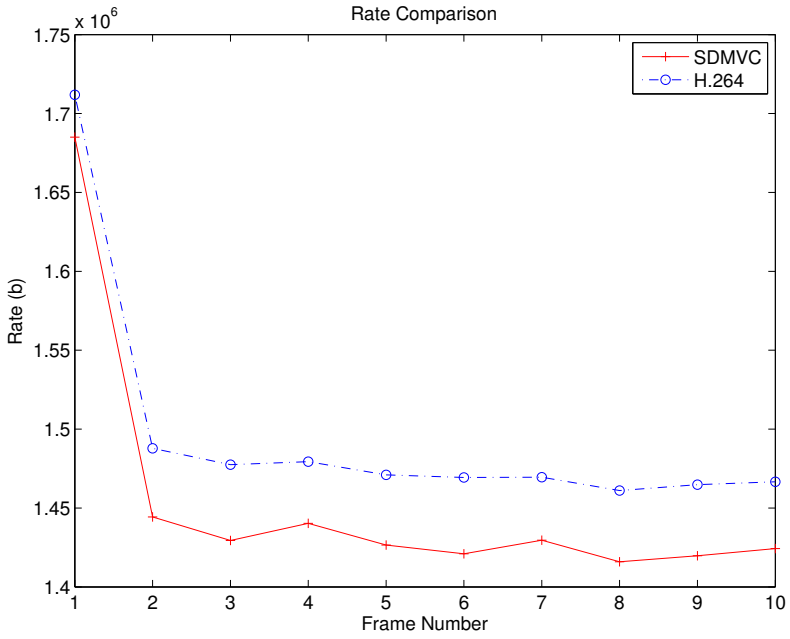


(a) Camera 4

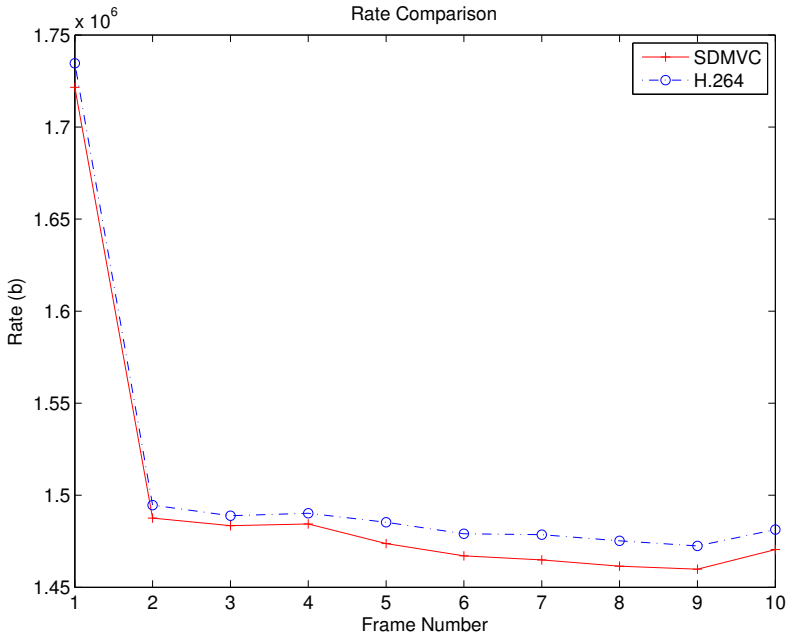


(b) Camera 5

Figure 7.20: Per Frame PSNR Comparison for Ballet Video, IPP, $\gamma = 0.4$, $QP_l = 10$, $QP_h = 22$

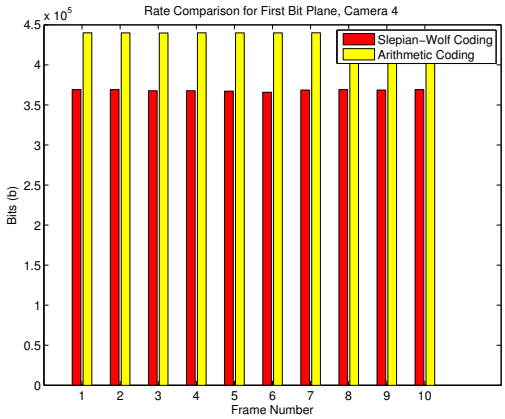


(a) Camera 4

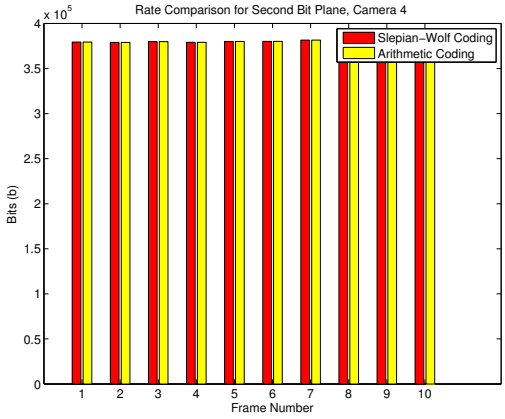


(b) Camera 5

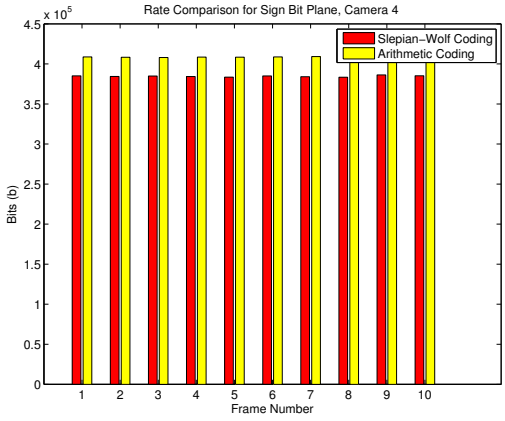
Figure 7.21: Per Frame Rate Comparison for Ballet Video, IPP, $\gamma = 0.4$, $QP_l = 10$, $QP_h = 22$



(a) First Bit Plane

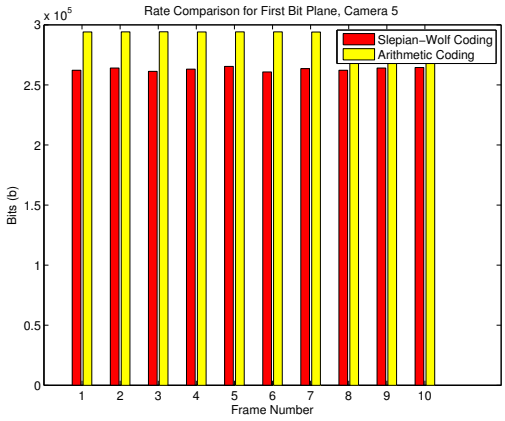


(b) Second Bit Plane

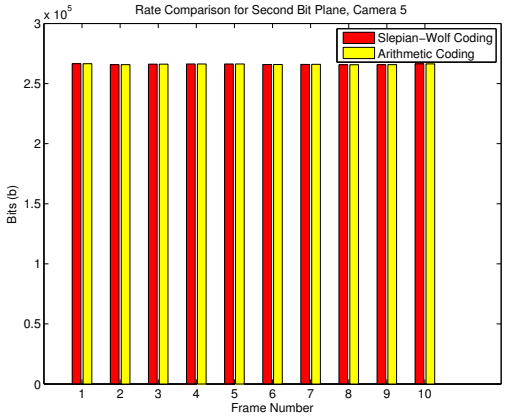


(c) Sign Bit Plane

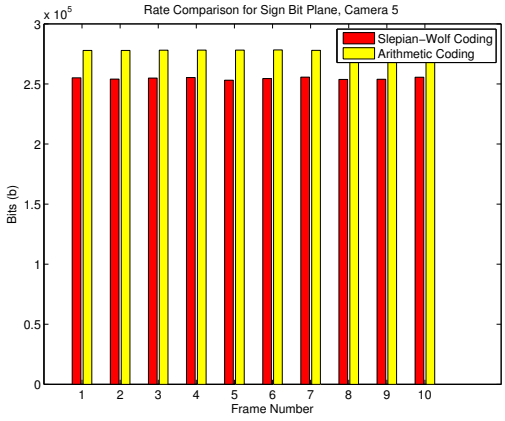
Figure 7.22: Rate Comparison between Slepian-Wolf Coding and Arithmetic Coding for Each Bit Plane of Breakdance Video, Camera 4, III, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$



(a) First Bit Plane

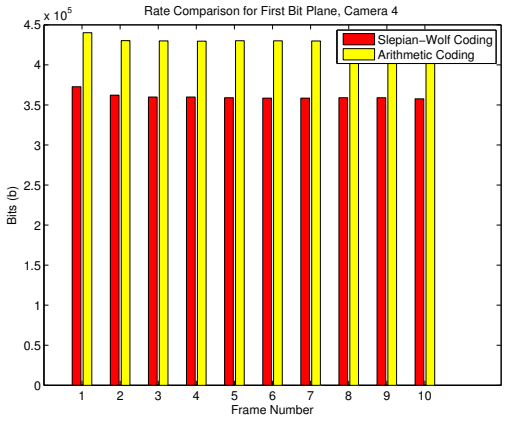


(b) Second Bit Plane

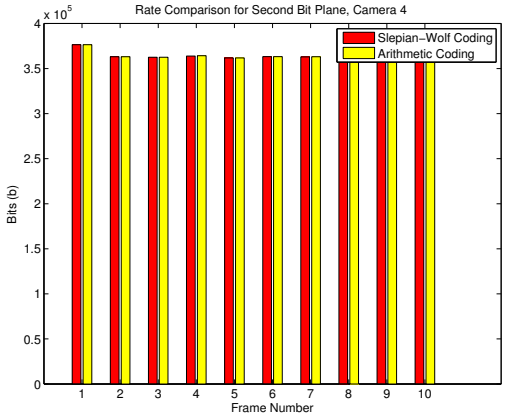


(c) Sign Bit Plane

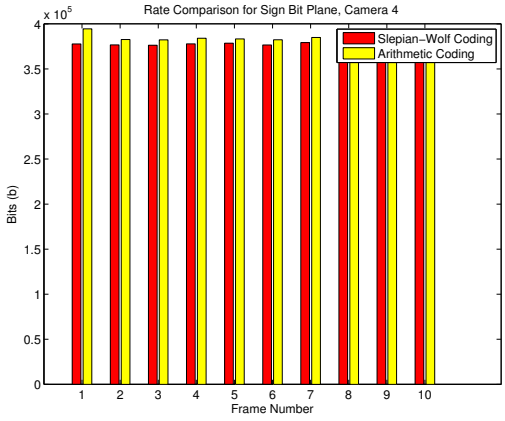
Figure 7.23: Rate Comparison between Slepian-Wolf Coding and Arithmetic Coding for Each Bit Plane of Breakdance Video, Camera 5, III, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$



(a) First Bit Plane

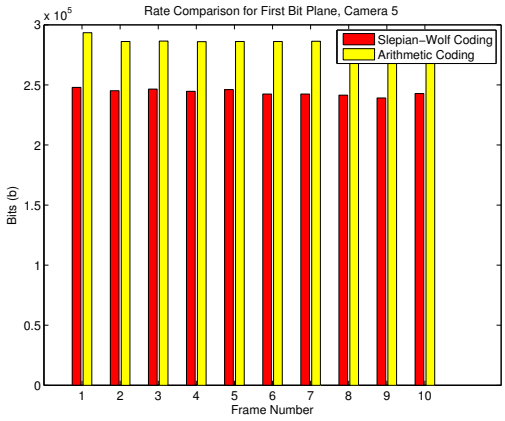


(b) Second Bit Plane

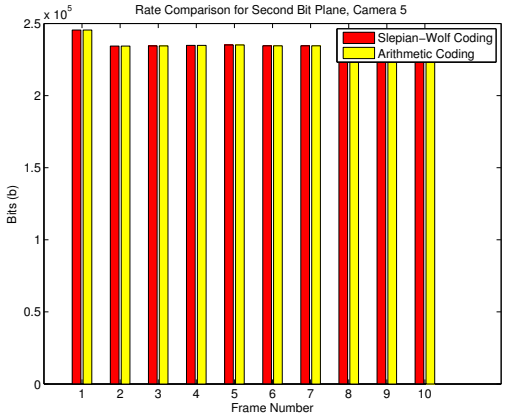


(c) Sign Bit Plane

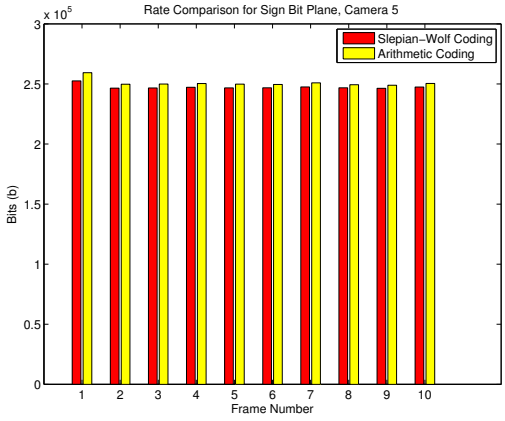
Figure 7.24: Rate Comparison between Slepian-Wolf Coding and Arithmetic Coding for Each Bit Plane of Ballet Video, Camera 4, IPP, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$



(a) First Bit Plane

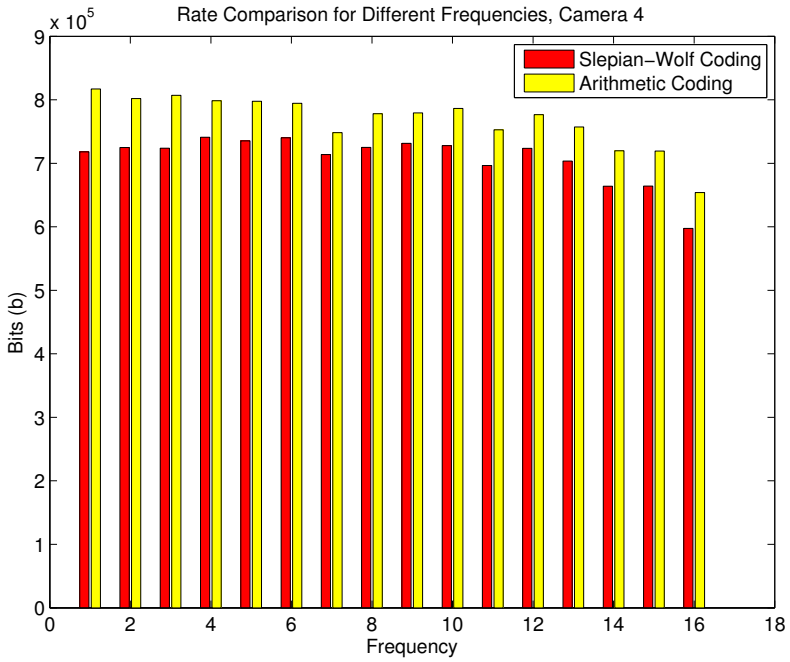


(b) Second Bit Plane

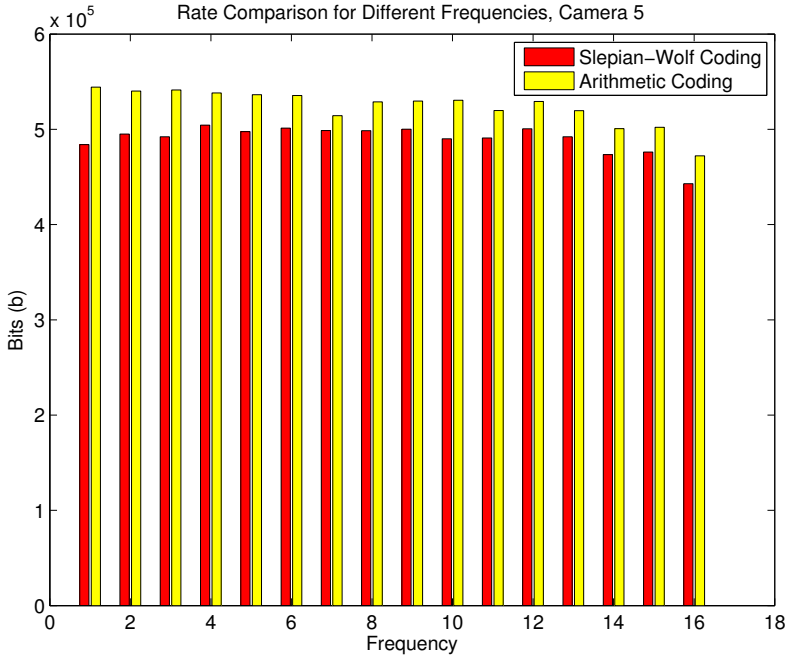


(c) Sign Bit Plane

Figure 7.25: Rate Comparison between Slepian-Wolf Coding and Arithmetic Coding for Each Bit Plane of Ballet Video, Camera 5, IPP, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$

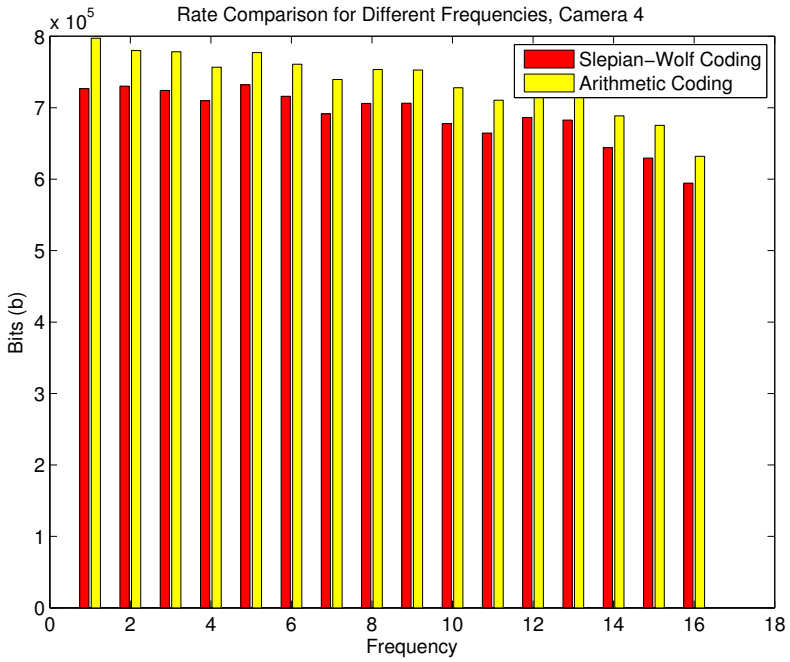


(a) Camera 4

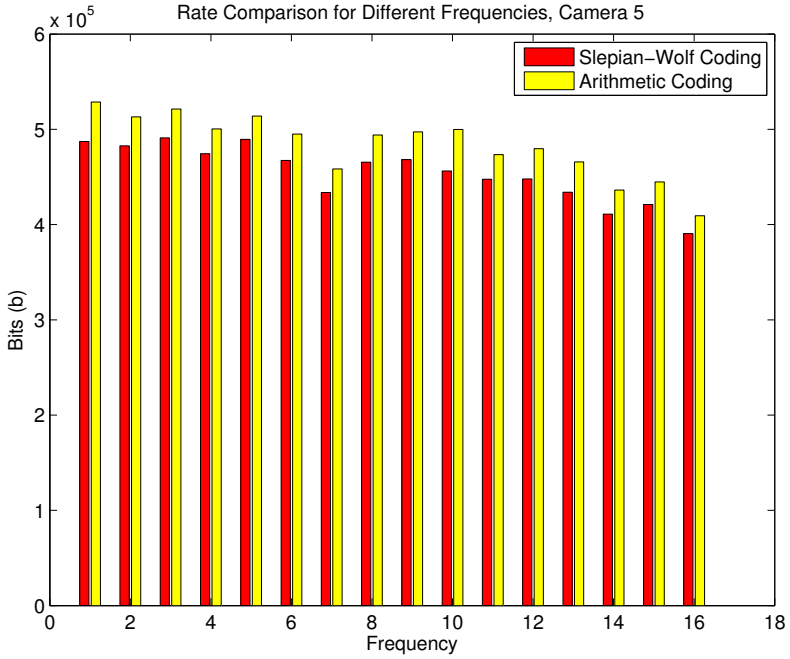


(b) Camera 5

Figure 7.26: Rate Comparison between Slepian-Wolf Coding and Arithmetic Coding for Each Frequency of Breakdance Video, III, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$

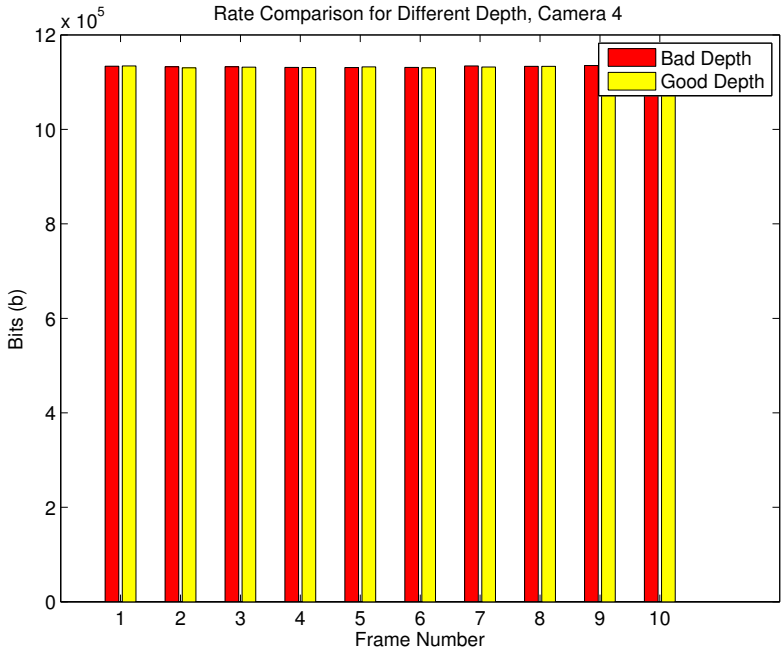


(a) Camera 4

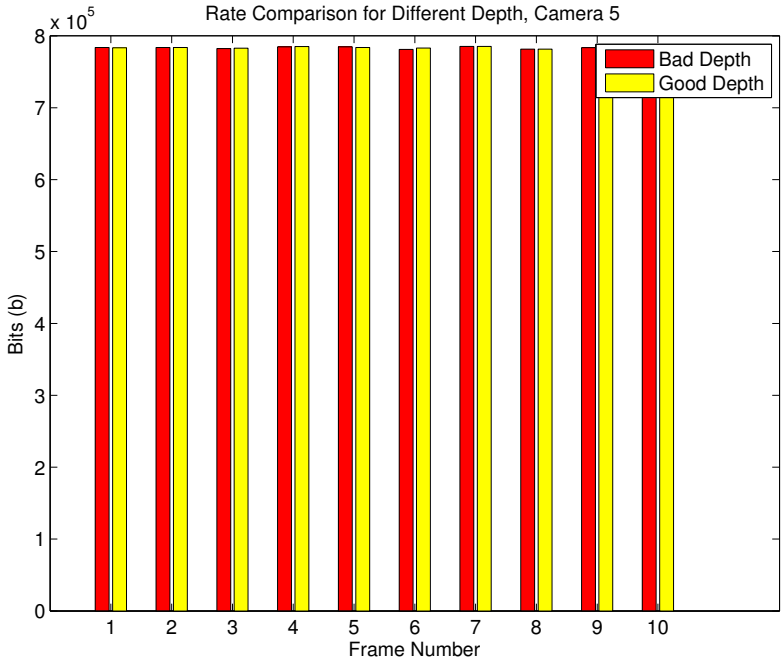


(b) Camera 5

Figure 7.27: Rate Comparison between Slepian-Wolf Coding and Arithmetic Coding for Each Frequency of Ballet Video, IPP, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$

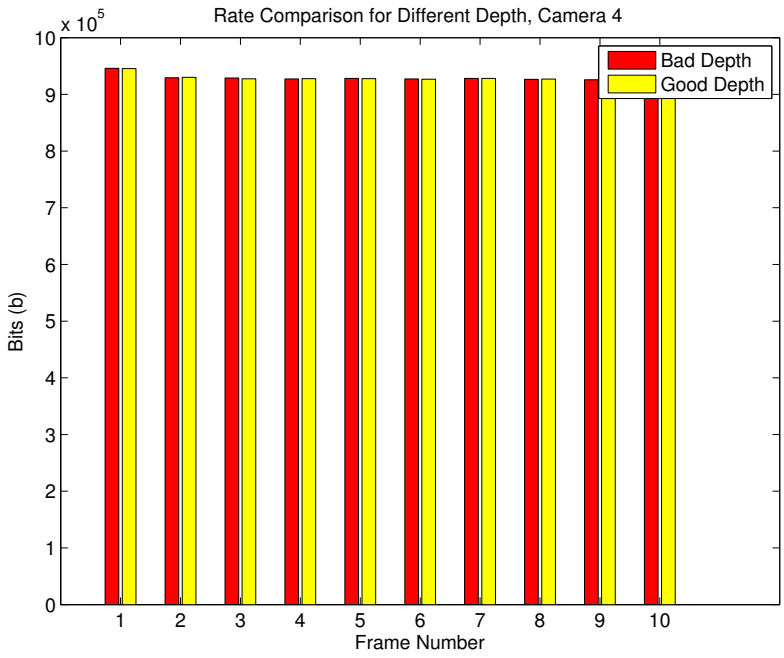


(a) Camera 4

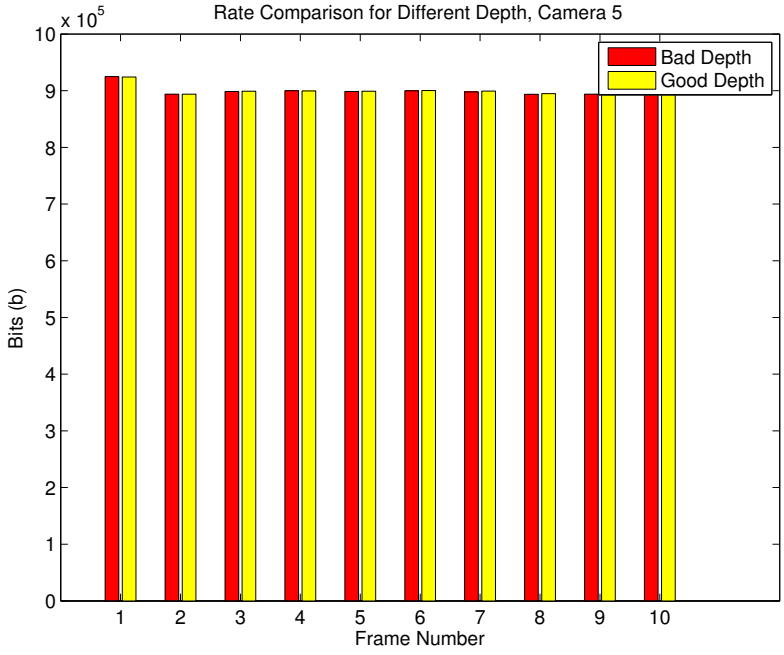


(b) Camera 5

Figure 7.28: Rate Comparison between Good Depth Map and Bad Depth Map of Breakdance Video, III, $\gamma = 0.4$, $QP_l = 4$, $QP_h = 16$



(a) Camera 4



(b) Camera 5

Figure 7.29: Rate Comparison between Good Depth Map and Bad Depth Map of Ballet Video, IPP, $\gamma = 0.5$, $QP_l = 4$, $QP_h = 16$

Therefore, we also can achieve some bit saving on the sign bit plane.

Fig. 7.26 and Fig. 7.27 give a view on the performance comparison between SNS-SWC and arithmetic coding for each frequency in a 4×4 block for Breakdance video with III structure and Ballet video with IPP structure when $\gamma = 0.4$, $QP_l = 4$ and $QP_h = 16$. They show that bits can be saved in every frequency.

We also evaluate the effect of depth quality on the performance of SMDVC-SNS. Fig. 7.28 gives the per-frame rate comparison for Breakdance video with III structure when $\gamma = 0.4$, $QP_l = 4$, and $QP_h = 16$. The good depth map is the original depth map that is estimated using original uncompressed images. The bad depth map is the rough depth map based on reconstructed low quality image at the decoder. It shows that in general SDMVC-SNS with good depth maps can save more bits, albeit very small, than SDMVC-SNS with bad depth maps. For the scenario shown in Fig 7.28, SDMVC-SNS with good depth map can save 6676 or 0.02% more bits than SDMVC-SNS with bad depth map. Fig. 7.29 shows the per-frame rate comparison for Ballet video with IPP structure when $\gamma = 0.5$, $QP_l = 4$, and $QP_h = 16$. In this case, SDMVC-SNS with good depth map can save 1439 or 0.003% more bits than SDMVC-SNS with bad depth map.

7.4 Conclusion

In this chapter, We propose a generic framework for the symmetric distributed multiview video coding. SDMVC-SNS can achieve the flexible rate allocation between two encoders. To the best of our knowledge, this is the first work to realize the simultaneous Slepian-Wolf coding of stereo video with the help of a symmetric distributed source code that achieves the whole Slepian-Wolf rate region. Our results show that the proposed SDMVC-SNS demonstrates a very promising result and moderate rate saving from H.264 frames can be achieved in the high-rate case. In the low-rate case, SDMVC-SNS cannot achieve better performance than separate H.264 coding scheme. SDMVC-SNS has a more complicated encoder and decoder.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

In the future, multiview video applications will likely emerge as one of prime multimedia applications. Multiview video compression algorithms are a crucial part of a multiview video system. In this thesis, we focus on developing novel and efficient multiview video coding algorithms. We make contributions on both joint multiview video coding and distributed multiview video coding. In summary, we have made the following contributions.

1. We propose a novel neighbor-based multiview coding scheme and design a new algorithm to automatically decide the stream encoding order and select the best neighbors for each stream to improve the compression performance [111].

2. A novel multiview video transcoder is put forward. The multiview video transcoder can be used to compress multiple compressed video streams and reduce the computational complexity of the encoder [114].

3. A learning-based multiview video scheme is designed and implemented. The multiview video coding problem is modeled as a semi-supervised learning problem. The scheme is a joint multiview coding scheme and can be used to compress multiview videos with a MVD format [82].

4. We propose two syndrome-based symmetric distributed source coding schemes: SNS-SWC and EETG. SNS-SWC has two independent sub-decoders and can simplify the decoding process of symmetric distributed source coding schemes. EETG simplifies the code construction of distributed source coding schemes using extended Tanner graph and is able to handle bit mismatch problem at the encoder [162].

5. A novel symmetric distributed multiview video coding scheme based on SNS-SWC is proposed. The scheme can achieve the flexible rate allocation between two distributed multiview video encoders. The scheme can have a better compression performance than the separate H.264 coding schemes at high rate [106].

The proposed algorithms seem very different, however, they have a common goal to achieve the joint entropy of multiview videos and reduce the computational complexity of the encoder and decoder. The neighbor-based multiview video coding scheme, multiview video transcoder and learning-based multiview video coding scheme are joint multiview video coding schemes. The symmetric distributed multiview video coding scheme belongs to the distributed multiview video coding category. SNS-SWC and EETG are generic distributed source coding schemes and can be used to help compress any data sources. These multiview coding schemes can be integrated into various subsystems of a multiview video system to achieve the design objective. For example, the multiview video transcoder and the symmetric distributed multiview video coding scheme can be used in video acquisition subsystem to reduce the video acquisition cost. The neighbor-based multiview video coding scheme and the learning-based multiview video coding scheme can be used with video transmission subsystem to reduce the network bandwidth.

8.2 Future Research

There are several avenues of research worthy to pursue in the future.

8.2.1 Extension to H.264 Schemes

The neighbor-based multiview coding scheme and multiview video transcoder discussed in Chapter 3 and 4 are general algorithms and can be used in any block-based multiview coding schemes. In this thesis, we evaluate the performance in MPEG2-based schemes. With the advent of H.264-based multiview coding schemes, it would be worthwhile and important to integrate them into H.264-based schemes.

8.2.2 Incorporate Compressive Sensing Theory into LMVC

LMVC is a novel joint multiview video coding scheme. However, its current rate-distortion performance is still inferior than block-based JMVM schemes. The essence of LMVC is to find a sparse representation of an image and use it to recover the original image. Compressive sensing theory [177] dictates that a sparse signal such as image can be compressed by

a sparse representation in one domain and recovered losslessly in another domain based on incoherence principle. Currently our implementation represents RP and recovers the original image both in the pixel domain. It might be helpful to incorporate compressive sensing theory into the schemes and recover the original image in the frequency domain.

8.2.3 Design Better LDPC Codes for EETG and SNS-SWC

Good LDPC codes are crucial to performance of EETG and SNS-SWC. EETG is based on extended Tanner graph. It is still an open problem to design good LDPC codes for extended Tanner graph. Our research suggests that LDPC codes for extended Tanner graph can come from one code ensemble. The challenge is how to design a density evolution algorithm for the extended Tanner graph. LDPC codes for SNS-SWC used in the symmetric distributed multiview video coding scheme is designed using the density evolution algorithm based on differential evolution [53]. There is still a performance gap with the asymmetric distributed multiview video coding. It is worthwhile to use density evolution algorithm for nonuniform channels [163] to design code and try to improve the performance.

8.2.4 Integrate Multiview Video Compression Subsystem with Other Subsystems of MVSN

In this thesis, we focus on addressing the challenges of multiview video compression. We treat the multiview video compression subsystem as a standalone system and study its performance independently. Since our ultimate goal is to build a multiview video system over Internet, it is worthy to integrate the multiview video compression with other subsystems and do a systematic study on its performance and find optimal configurations for different applications.

Bibliography

[1] C. Fehn, E. Cookie, O. Schreer, and P. Kauff, "3d analysis and image-based rendering for immersive tv applications," *Signal Processing: Image Communications*, vol. 17, pp. 705–715, 2002.

[2] W. Matusik and H. Pfister, "3d tv: A scalable system for real-time acquisition, transmission and autostereoscopic display for dynamic scenes," in *Proc. of ACM SIGGRAPH, 2004*, 2004.

[3] C. L. Zitnick, S. B. King, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *Proc. of ACM SIGGRAPH, 2004*, 2004.

[4] M. Gross, S. Wurmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. V. Gool, S. Lang, K. Strehlke, A. V. Moere, and O. Staadt, "blue-c: A spatially immersive display and 3d video portal for telepresence," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 819–828, 2003.

[5] J. Carranza, C. Theobalt, M. A. Magnor, and H. P. Seidel, "Free-viewpoint video of human actors," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 569–577, 2003.

[6] A. Smolic and P. Kauff, "Interactive 3d video representation and coding technologies," *Proceedings of the IEEE, Special Issue on Advances in Video Coding and Delivery*, 2004.

[7] M. Magnor, "3d-tv - the future of visual entertainment," in *Proc. of Multimedia Databases and Image Communications (MDIC'04)*, 2004.

[8] A. Smolic, C. Fehn, and K. Mueller, "Mpeg 3dav - video-based rendering for interactive tv applications," in *Proc. 10. Dortmunder Fernsehseminar, ITG/FKTG-Fachtagung*, 2003.

[9] A. Smolic, K. Mueller, P. Merkle, T. Rein, P. Eisert, and T. Wiegand, "Free viewpoint video extraction, representation, coding, and rendering," in *Proc. of IEEE International Conference on Image Processing (ICIP 2004)*, 2004.

[10] C. Fehn, R. D. Barre, and S. Pastoor, "Interactive 3d tv - concepts and key technologies," *Proceedings of IEEE*, vol. 94, no. 3, pp. 524–538, March 2006.

[11] S.-C. Chan, K.-T. Ng, Z.-F. Gan, K.-L. Chan, and H.-Y. Shum, "The plenoptic video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 12, pp. 1650–1659, December 2005.

[12] C. Zhang and T. Chen, "A survey on image-based rendering–representation, sampling and compression," *Signal Processing: Image Communications*, vol. 19, no. 1, pp. 1–28, 2004.

[13] H.-Y. Shum, S. B. Kang, and S.-C. Chan, "Survey of image-based representations and compression technique," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 13, no. 11, pp. 1020–1037, 2003.

[14] E. H. Adelson and J. R. Bergen, "The plenoptic function and the elements of early vision," in *in: M. Landy, J. Anthony Movshon (Eds.), Computational Models of Visual Processing, The MIT Press, Cambridge, MA*, 1991, pp. 3–20.

[15] L. McMillan and U. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proc. of ACM Annual Computer Graphics Conference (SIGGRAPH 1995)*, 1995, pp. 39–46.

[16] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. of ACM SIGGRAPH 1996*, 1996.

[17] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proc. of ACM SIGGRAPH 1996*, 1996.

[18] H.-Y. Shum, K.-T. Ng, and S.-C. Chan, "Virtual reality using the concentric mosaic: Construction, rendering and data compression," in *Proc. of IEEE International Conference on Image Processing (ICIP 2000)*, 2000.

[19] S. E. Chen, "Quicktime vr–an image-based approach to virtual environment navigation," in *Proc. of ACM SIGGRAPH 1995*, Los Angels, CA, 1995, pp. 29–38.

[20] S. B. Kang, Y. Li, X. Tong, and H.-Y. Shum, "Image-based rendering," *Foundation and Trends in Computer Graphics and Vision*, vol. 2, no. 3, pp. 173–258, 2006.

[21] J. C. Yang, M. Everett, C. Buehler, and L. McMillan, "A real-time distributed light field camera," in *Proc. of the 13th Eurograhics Workshop on Rendering*, 2002.

[22] ——, "The light field video camera," in *Proc. of Media Processors 2002, SPIE Electronic Imaging 2002*, 2002.

[23] I. Ihm, S. Park, and R. Lee, "Rendering of spherical light fields," in *Proc. of Pacific Graphics*, Seoul, Korea, 1997, pp. 59–68.

[24] E. Camahort, A. Lerios, and D. Fussell, "Uniformly sampled light fields," in *Proc. of Nineth Eurographics Workshop on Rendering*, Vienna, Austria, 1998, pp. 117–130.

[25] J. X. Chai, X. Tong, S. C. Chan, and H. Y. Shum, "Plenoptic sampling," in *Proc. of ACM Annual Computer Graphics Conference (SIGGRAPH 2000)*, 2000, pp. 307–318.

[26] C. Zhang and T. Chen, "Spectral analysis for sampling image-based rendering data," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 11, pp. 1038–1050, 2003.

[27] A. Smolic, K. Mueller, N. Stefanoski, J. Ostermann, A. Gotchev, G. B. Akar, G. Triatafyllidis, and A. Koz, "Coding algorithms for 3dtv —- a survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1606–1621, November 2007.

[28] M. Flierl and B. Girod, "Multiview video compression," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 67–76, November 2007.

[29] R. Braden, D. Clark, and S. Shenker, "Integrated services in internet architecture: An overview," in *RFC 1633*, 1994.

[30] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reservation protocol (rsvp)," in *RFC 2205*, 1997.

[31] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," in *RFC 2475*, 1998.

[32] S. Chen and L. Willams, "View interpolation for image synthesis," in *Proc. of ACM Aunnual Computer Graphics Conference (SIGGRAPH 1993)*, 1993, pp. 279–288.

[33] S. M. Seitz and C. M. Dyer, "View morphing," in *Proc. of ACM Annual Computer Graphics Conference (SIGGRAPH 1996)*, 1996, pp. 21–30.

[34] T. M. Cover and J. A. Thomas, *Elements of Information Theory, Second Edition*. Hoboken, New Jersy: John Wiley & Sons, 2006.

[35] R. G. Gallager, *Information Theory and Reliable Communication*. Hoboken, New Jersy: John Wiley and Sons, 1968.

[36] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[37] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang, "Network coding theory," *Foundation and Trends in Communications and Information Theory*, vol. 2, no. 4 and 5, pp. 241–381, 2005.

[38] C. Fragouli and E. Soljanin, "Network coding fundamentals," *Foundation and Trends in Communications and Information Theory*, vol. 2, no. 1, pp. 1–133, 2007.

[39] C. Guillemot, F. Pereira, L. Torres, T. Ebrahimi, R. Leonardi, and J. Ostermann, "Distributed monoview and multiview video coding," *IEEE Signal Processing Magazine*, vol. 24, no. 5, pp. 67–76, September 2007.

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

[41] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transaction on Information Theory*, no. 1, pp. 1–10, 1976.

[42] S. S. Pradhan, J. Chou, and K. Ramchandran, "Duality between source coding and channel coding and its extension to the side information case," *IEEE Transactions on Information Theory*, vol. 49, no. 5, pp. 1181–1203, May 2003.

[43] T. Berger, "Multiterminal source coding," in *in The Information Theory Approach to Communications, G. Longo, Ed. New York: Springer-Verlag*, 1977.

[44] S. Tung, *Multiterminal Rate-Distortion Theory*. Ithaca, NY: Ph. D Dissertation, School of Electrical Engineering, Cornell University, 1978.

[45] Y. Oohama, "The rate-distortion theory for the gaussian multiterminal source coding systems with several side information at the decoder," *IEEE Transaction on Information Theory*, no. 7, pp. 2577–2593, 2005.

[46] A. Wagner, S. Tavildar, and P. Viswanath, "Rate region of the quadratic gaussian two-encoder source coding problem," *IEEE Transaction on Information Theory*, no. 5, pp. 1938–1961, May 2008.

[47] Y. Yang, V. Stankovic, Z. Xiong, and W. Zhao, "On multiterminal source code design," *IEEE Transactions on Information Theory*, vol. 54, no. 5, pp. 2278–2302, May 2008.

[48] M. Marcellin and T. Fischer, "Trellis coded quantization of memoryless and gaussian-markov sources," *IEEE Transactions on Communications*, vol. 38, no. 1, pp. 82–93, 1990.

[49] R. G. Gallager, "Low density parity check codes," Ph.D. dissertation, Massachusetts Institute of Technology, 1963.

[50] D. J. C. MacKay and R. M. Neal, "Near shannon limit performance of low density parity check codes," *Eletronic Letters*, vol. 33, pp. 457–458, 1996.

[51] M. R. Tanner, "A recursive approach to low complexity codes," *IEEE Transaction on Information Theory*, no. 5, pp. 533–547, 1981.

[52] T. J. Richardson and R. L. Urbanke, "The capacity of low density parity check codes under message passing decoding," *IEEE Transaction on Information Theory*, no. 2, pp. 599–618, Feb. 2001.

[53] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low density parity check codes," *IEEE Transaction on Information Theory*, no. 2, pp. 619–637, Feb. 2001.

[54] F. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transaction on Information Theory*, no. 2, pp. 488–519, Feb. 2001.

[55] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA: Morgan Kaufmann Publisher Inc., 1988.

[56] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 7, pp. 1204–1216, July 2000.

[57] R. W. Yeung and Z. Zhang, "Distributed source coding for satellite communication," *IEEE Transactions on Information Theory*, vol. 45, pp. 1111–1120, 1999.

[58] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, February 2003.

[59] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE Transactions on Networking*, vol. 11, no. 5, pp. 782–795, October 2003.

[60] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1973–1982, June 2005.

[61] T. Ho, R. Koetter, M. Medard, M. Effros, J. Shi, and D. Karger, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, October 2006.

[62] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. of Allerton Conference on Communication, Control and Computing*, 2003.

[63] R. W. Yeung and N. Cai, "Network error correction, part i: Basic concepts and upper bounds," *Communications in Information and Systems*, vol. 6, no. 1, pp. 19–36, 2006.

[64] N. Cai and R. W. Yeung, "Network error correction, part ii: Lower bounds," *Communications in Information and Systems*, vol. 6, no. 1, pp. 37–54, 2006.

[65] C. Zhang and J. Li, "Compression of lumigraph with multiple reference frame (mrf) prediction and just-in-time rendering," in *Proc. of IEEE Data Compression Conference (DCC 2000)*, 2000.

[66] L. Luo, Y. Wu, J. Li, and Y.-Q. Zhang, "3-d wavelet compression and progressive inverse wavelet synthesis rendering of concentric mosaic," *IEEE Transactions on Image Processing*, pp. 802–816, 2002.

[67] C.-L. Chang, X. Zhu, P. Ramanathan, and B. Girod, "Inter-view wavelet compression of light fields with disparity-compensated lifting," in *Proc. of SPIE Visual Communications and Image Processing (VCIP-03)*, 2003.

[68] ——, "Shape adaptation for light field compression," in *Proc. of IEEE International Conference on Image Processing (ICIP-2003)*, 2003.

[69] J. Li, H. Shum, and Y.-Q. Zhang, "On the compression of image based rendering scene," in *Proc. of IEEE International Conference on Image Processing (ICIP 2000)*, 2000.

[70] M. Magnor, P. Ramanathan, and B. Girod, "Multi-view coding for image-based rendering using 3-d scene geometry," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 13, no. 11, 2003.

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

[72] I. .-. A. 3, "Mpeg-2 multiview profile," *ISO/IEC JTC1/SC29/WG11 document no. N1366*, September 1996.

[73] S.-C. Chan, K.-T. Ng, Z.-F. Gan, K.-L. Chan, and H.-Y. Shum, "The compression of simplified dynamic light field," in *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2003)*, April 2003.

[74] J. Lim, K. N. Ngan, W. Yang, and K. Sohn, "A multiview sequence codec with view scalability," *Signal Processing: Image Communication*, vol. 19, no. 3, pp. 239–256, 2004.

[75] A. Vetro, W. Matusik, H. Pfister, and J. Xin, "Coding approaches for end-to-end 3d tv systems," in *Proc. of Picture Coding Symposium (PCS 2004)*, 2004.

[76] E. Martinian, A. Behrens, J. Xin, A. Vetro, and H. Sun, "Extensions of h.264/avc for multiview video compression," in *Proc. of IEEE international Conference on Image Processing (ICIP 2006)*, Atlanta, GT, USA, 2006.

[77] E. Martinian, A. Behrens, J. Xin, and A. Vetro, "View synthesis for multiview video compression," in *Proc. of Picture Coding Symposium (PCS 2006)*, 2006.

[78] S. Ince, E. Martinian, S. Yea, and A. Vetro, "Depth estimation for view synthesis for multiview video coding," in *Proc. of 3DTV Conference (3DTV-CON 2007)*, May 2007.

[79] A. Vetro, Y. Su, H. Kimata, and A. Smolic, "Joint multiview video model jmvm 2.0," in *ITU-T and ISO/IEC Joint Video Team, Document JVT-U207*, HangZhou, China, 2006.

[80] S.-T. Na, K.-J. Oh, and Y.-S. Ho, "Joint coding of multiview video and corresponding depth map," in *Proc. of IEEE International Conference on Image Processing (ICIP 2008)*, San Diego, CA, USA, 2008.

[81] P. Merkle, A. Smolic, K. Mller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *Proc. of IEEE International Conference on Image Processing (ICIP 2007)*, San Antonio, TX, USA, 2007.

[82] B. Bai, L. Cheng, C. Lei, P. Boulanger, and J. Harms, "Learning-based multiview video coding," in *Proc. of the 27th Picture Coding Symposium (PCS 2009)*, Chicago, Illinois, USA, May 2009.

[83] B. L. Tseng and D. Anastassiou, "Multiviewpoint video coding with mpeg-2 compatibility," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 4, pp. 414–419, 1996.

[84] D. Tzovaras, N. Grammalidis, and M. G. Strintzis, "Object-based coding of stereo image sequences using joint 3-d motion/disparity compensation," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 312–327, 1997.

[85] S. Malassiotis and M. G. Strintzis, "Object-based coding of stereo image sequences using three-dimensional models," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 7, no. 6, pp. 892–905, 1997.

[86] R.-S. Wang and Y. Wang, "Multiview video sequence analysis, compression, and virtual viewpoint synthesis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 3, pp. 397–410, 2000.

[87] S.-Y. Chien, S.-H. Yu, L.-F. Ding, Y.-N. Huang, and L.-G. Chen, "Fast disparity estimation algorithm for mesh-based stereo image/video compression with two-stage hybrid approach," in *Proc. of SPIE Visual Communications and Image Processing (VCIP) 2003*, vol. 5150, July 2003, pp. 1521–1530.

[88] J. H. Park and H. W. Park, "A mesh-based disparity representation method for view interpolation and stereo image compression," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1751–1762, 2006.

[89] C. Fehn, P. Kauff, M. O. D. Beeck, F. Ernst, W. Ijssel-Steijn, M. Pollefeys, L. V. Gool, E. Ofek, and I. Sexton, "An evolutionary and optimized approach on 3d-tv," in *Proc. of International Broadcast Conference*, 2002, pp. 357–365.

[90] E. Lamboray, S. Wurmlin, and M. GrosS, "Real-time streaming of point-based 3d video," in *Proc. of IEEE Virtual Reality, 2004*, 2004.

[91] G. Ziegler, H. Lensch, M. Magnor, and H.-P. Seidel, "Multi-video compression in texture space using 4d spihtl," in *Proc. of IEEE International Workshop on Multimedia Signal Processing (MMSP'04)*, 2004.

[92] G. Ziegler, H. Lensch, N. Ahmed, M. Magnor, and H.-P. Seidel, "Multi-video compression in texture space," in *Proc. of IEEE International Conference on Image Processing (ICIP'04)*, 2004.

[93] S. Kum, K. Mayer-Patel, and H. Fuchs, "Real-time compression for dynamic 3d environments," in *Proc. of ACM Multimedia, 2003*, 2003.

[94] E. Lamboray, S. Wurmlin, M. Waschbusch, M. GrosS, and H. Pfister, "Unconstrained free-viewpoint video coding," in *Proc. of IEEE International Conference on Image Processing (ICIP 2004)*, 2004.

[95] S. K. Nath and E. Dubois, "An improved, wavelet-based, stereoscopic image sequence codec with snr and spatial scalability," *Signal Processing: Image Communications*, vol. 21, no. 3, pp. 181–199, 2006.

[96] H. Witsenhausen and A. Wyner, "Interframe coder for video signals," in *United States Patent 4191970*, 1980.

[97] A. Aaron, R. Zhang, and B. Girod, "Wyner-ziv coding for motion video," in *Proc. of Asilomar Conference on Signals and Systems*, Pacific Grove, CA, 2002.

[98] R. Puri and K. Ramchandran, "Prism: A new robust video coding architecture based on distributed compression principles," in *Proc. of Allerton Conference on Communication, Control, and Computing*, Allerton, IL, USA, 2002.

[99] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proceedings of the IEEE, Special Issue on Advances in Video Coding and Delivery*, 2005.

[100] X. Guo, Y. Lu, F. Wu, and G. W, "Distributed multiview video coding," in *Proc. of Visual Communications and Image Processing (VCIP 2006)*, San Jose, CA, 2006.

[101] M. Ouaret, F. Dufaux, and T. Ebrahimi, "Fusion-based multiview distributed video coding," in *Proc. of he 4th ACM international workshop on Video surveillance and sensor networks (VSSN 2006)*, Santa Barbara, California, USA, 2006.

[102] B. Song, O. Bursalioglu, A. K. Roy-Chowdhury, and E. Tuncel, "Towards a multi-terminal video compression algorithm using epipolar geometry," in *Proc. of IEEE international Conference on Acoustic, Speech, and Signal Processing (ICASSP 2006)*, Toulouse, France, May 2006.

[103] B. Song, A. Roy-Chowdhury, and E. Tuncel, "A multi-terminal model-based video compression algorithm," in *Proc. of IEEE international Conference on Image Processing (ICIP 2006)*, Atlanta, GT, USA, 2006.

[104] M. Flierl and B. Girod, "Coding of multiview image sequences with video sensors," in *Proc. of IEEE international Conference on Image Processing (ICIP 2006)*, Atlanta, GT, USA, 2006.

[105] Y. Yang, V. Stankovic, Z. Xiong, and W. Zhao, "Two-terminal video coding," *IEEE Transactions on Image Processing*, vol. 18, no. 3, pp. 534–551, 2009.

[106] B. Bai, Y. Yang, C. Lei, P. Boulanger, and J. Harms, "Symmetric distributed multiview video coding," in *Proc. of the 34th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2009)*, Taipei, Taiwan, 2009.

[107] X. Zhu, A. Aaron, and B. Girod, "Distributed compression for large camera arrays," in *Proc. of IEEE Workshop on Statistical Signal Processing*, St Louis, Missouri, 2003.

[108] T. Fujii and M. Tanimoto, "Free-viewpoint tv system based on ray-space representation," in *Proc. of SPIE ITCom, Vol. 4864-22*, 2002.

[109] P. E. Devebec, C. J. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: A hybrid geometry and image-based approach," in *Proc. of ACM Annual Computer Graphics Conference (SIGGRAPH 1996)*, 1996, pp. 11–20.

[110] J. Shade, S. Gortler, L.-W. He, and R. Szeliski, "Layered depth images," in *Proc. of ACM Annual Computer Graphics Conference (SIGGRAPH 1998)*, Orlando, FL, 1998, pp. 231–242.

[111] B. Bai, P. Boulanger, and J. Harms, "An efficient multiview compression scheme," in *Proc. of IEEE International Conference on Multimedia and Expo (ICME) 2005*, 2005.

[112] "http://www.povray.org."

[113] "Report on 3dav exploration," in *Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC), ISO/IEC JTC1/SC29/WG11, Doc. N5878*, July 2003.

[114] B. Bai, P. Boulanger, and J. Harms, "A multiview video transcoder," in *Proc. of the 13th ACM International Conference on Multimedia (ACMMM 2005)*, Singapore, 2005.

[115] J. Xin, C.-W. Lin, and M.-T. Sun, "Digital video transcoding," *Proceedings of IEEE*, vol. 93, no. 1, January 2005.

[116] A. Vetro, C. Christopulos, and H. Sun, "Video transcoding architectures and techniques: an overview," *IEEE Signal Processing Magazine*, vol. 20, no. 2, pp. 18–29, 2003.

[117] A. Eleftheiadis and D. Anastassiou, "Constrained and general dynamic rate reshaping of compressed digital video," in *Proc. of IEEE International Conference on Image Processing*, vol. 3, October 1995, pp. 396–399.

[118] H. Sun, W. Kwok, and J. W. Zdepski, "Architectures for mpeg compressed bitstream scaling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 2, pp. 191–199, 1996.

[119] Y. Nakajima, H. Hori, and T. Kanoh, "Rate conversion of mpeg coded video by requantization process," in *Proc. of IEEE International Conference on Image Processing*, vol. 3, October 1995, pp. 408–411.

[120] J. Youn, M.-T. Sun, and J. Xin, "Video transcoder architectures for bitrate scaling of h.263 bit streams," in *Proc. of ACM Multimedia*, November 1999.

[121] G. Keesman, R. Hellinghuizen, F. Hoeksema, and G. Heideman, "Transcoding of mpeg bitstreams," *Signal Processing: Image Communication*, vol. 8, no. 6, pp. 481–500, 1996.

[122] P. A. A. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bitrate reduction of mpeg-2 bit streams," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 8, pp. 953–967, 1998.

[123] N. Bjork and C. Christopoulos, "Transcoder architecture for video coding," *IEEE Transactions on Consumer Electronics*, vol. 44, no. 1, pp. 88–98, Feburary 1998.

[124] B. Shen, I. K. Ishwar, and V. Bhaskaran, "Adaptive motion-vector re-sampling for compressed video downscaling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, pp. 929–936, September 1999.

[125] J. Xin, M.-T. Sun, B. S. Choi, and K. W. Chun, "An hdtv to sdtv spatial transcoder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 11, pp. 998–1008, November 2002.

[126] J. Youn, M.-T. Sun, and C.-W. Lin, "Motion vector estimation for high-performance transcoding," *IEEE Transactions on Multimedia*, vol. 1, no. 1, pp. 30–40, Mar 1999.

[127] M.-J. Chen, M.-C. Chu, and C.-W. Pan, "Efficient motion estimation algorithm for reduced frame-rate video transcoder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 4, pp. 269–275, April 2002.

[128] K.-D. Seo and J.-K. Kim, "Motion vector refinement for video downsampling in the dct domain," *IEEE Signal Processing Letter*, vol. 9, no. 11, pp. 356–359, November 2002.

[129] K.-T. Fung, Y.-L. Chan, and W.-C. Siu, "New architecture for dynamic frame-skipping transcoder," *IEEE Transactions on Image Processing*, vol. 11, no. 8, pp. 886–900, August 2002.

[130] F. Hartung and B. Girod, "Watermarking of uncompressed and compressed video," *Signal Processing*, vol. 66, no. 3, pp. 283–301, May 1998.

[131] J. Meng and S. F. Chang, "Embedding visible video watermarks in the compressed domain," in *Proc. of IEEE International Conference on Image Processing*, vol. 1, 1998, pp. 474–477.

[132] G. de los Reyes, A. R. Reibman, S.-F. Chang, and J. C.-I. Chuang, "Error-resilient transcoding for video over wireless channels," *IEEE Journal on Selected Areas of Communications*, vol. 18, no. 6, June 2000.

[133] S. Dogan, A. Cellatoglu, M. Uyguroglu, A. H. Sadka, and A. M. Kondoz, "Error-resilient video transcoding for robust internetwork communications using gprs," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 6, pp. 453–464, June 2002.

[134] R. Puri, K.-W. Lee, K. Ramchandran, and V. Bhargavan, "An integrated source transcoding and congestion control paradigm for video streaming in the internet," *IEEE Transactions on Multimedia*, vol. 3, no. 1, pp. 18–32, Mar 2001.

[135] J. Xin, "Improved standard-conforming video transcoding techniques," Ph.D. dissertation, University of Washington, Seattle, WA, December 2002.

[136] Y. Su, M.-T. Sun, and V. Hsu, "Global motion estimation from coarsely sampled motion vector field and the applications," vol. 2, 2003, pp. 628–631.

[137] J. Xin, M.-T. Sun, and K.-S. Kan, "Bit allocation for joint transcoding of multiple mpeg coded video streams," in *Proc. of IEEE International Conference on Multimedia and Expo (ICME 2001)*, 2001.

[138] I. Koo, P. Nasiopoulos, and R. Ward, "Joint mpeg-2 coding for multi-program broadcasting of pre-recorded video," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 1999)*, 1999.

[139] H. Sorial, W. E. Lynch, and A. Vincent, "Joint transcoding of multiple mpeg video bitstreams," in *Proc. of IEEE International Symposium on Circuits System*, 1999.

[140] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[141] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[142] L. Cheng and S. Vishwanathan, "Learning to compress images and videos," in *Proc. of the 24th Annual Conference on Machine Learning (ICML 2007)*, Corvallis, Oregon, USA, 2007.

[143] X. Zhu, "Semi-supervised learning literature survey," in *Technical Report 1530, Computer Sciences, University of Wisconsin-Madison*, 2005.

[144] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *Journal of Machine Learning Research*, no. 7, pp. 2399–2434, 2006.

[145] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structure for multiview video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1461–1473, 2007.

[146] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *Proc. of ACM Annual Computer Graphics Conference (SIGGRAPH 2004)*, Los Angels, CA, 2004, pp. 689–694.

[147] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, May 2002.

[148] A. Moffat, R. Neal, and I. Witten, "Arithmetic coding revisited," *ACM Transactions on Information Systems*, vol. 16, no. 3, pp. 256–294, 1998.

[149] Z. Xiong, A. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, vol. 21, pp. 80–94, 2004.

[150] R. Zamir, S. Shamai, and U. Erez, "Nested linear/lattice codes for structured multiterminal binning," *IEEE Transactions on Information Theory*, vol. 48, no. 6, pp. 1250–1276, 2002.

[151] S. S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (discus): Design and construction," in *Proc. of IEEE Data Compression Conference*, Snowbird, UT, USA, Mar. 1999, pp. 158–167.

[152] A. D. Wyner, "On source coding with side information at the decoder," *IEEE Transaction on Information Theory*, no. 5, pp. 294–300, May 1975.

[153] B. Rimoldi and R. Urbanke, "Asynchronous slepian-wolf coding via source-splitting," in *Proc. of IEEE International Symposium on Information Theory (ISIT 1997)*, Ulm, Germany, 1997.

[154] T. P. Coleman, A. H. Lee, M. Medard, and M. Effros, "On some new approaches to practical slepian-wolf compression inspired by channel coding," in *Proc. of IEEE Data Compression Conference (DCC 2004)*, Snowbird, UT, 2004.

[155] V. Stankovic, A. D. Liveris, Z. Xiong, and C. N. Georghiades, "On code design for the slepian-wolf problem and lossless multiterminal networks," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1495–1507, 2006.

[156] P. Tan and J. Li, "A general and optimal framework to achieve the entire rate region for slepian-wolf coding," *EURASIP Signal Processing Journal, Special Issue on Distributed Source Coding*, vol. 86, no. 11, 2006.

[157] N. Gehrig and P. L. Dragotti, "Symmetric and asymmetric slepian-wolf codes with systematic and non-systematic linear codes," *IEEE Communication Letters*, vol. 9, no. 1, 2005.

[158] M. Sartipi and F. Fekri, "Distributed source coding in wireless sensor networks using ldpc coding: The entire slepian-wolf rate region," in *Proc. of the Wireless Communication Networking Conference (WCNC 2005)*, New Orlean, USA, 2005.

[159] D. Schonberg, "Practical distributed source coding and its application to the compression of encrypted data," Ph.D. dissertation, University of California, Berkeley, 2007.

[160] J. Garcia-Frias and Y. Zhao, "Compression of correlated binary sources using turbo codes," *IEEE Communication Letters*, vol. 5, no. 10, pp. 417–419, 2001.

[161] S. S. Pradhan and K. Ramchandran, "Generalized coset codes for distributed binning," *IEEE Transactions on Information Theory*, vol. 51, no. 10, pp. 3457–3474, 2005.

[162] B. Bai, Y. Yang, P. Boulanger, and J. Harms, "Symmetric distributed source coding using ldpc," in *Proc. of the IEEE International Conference on Communication (ICC 2008)*, Beijing, China, 2008.

[163] H. Pishro-Nik, N. Rahnavard, and F. Fakri, "Nonuniform error correction using irregular low density parity check codes," *IEEE Transaction on Information Theory*, no. 7, pp. 2702–2714, 2005.

[164] C. Yeo and K. Ramchandran, "Robust distributed multiview video compression for wireless camera network," in *Proc. of SPIE Visual Communications and Image Processing (VCIP 2007)*, San Jose, CA, USA, 2007.

[165] Y. Yang, V. Stankovic, W. Zhao, and Z. Xiong, "Multiterminal video coding," in *Proc. of IEEE International Conference on Image Processing (ICIP 2007)*, San Antonio, TX, USA, 2007.

[166] K. Kang and C. Lu, "Multiview distributed video coding with low-complexity intersensor communication over wireless video sensor network," in *Proc. of IEEE International Conference on Image Processing (ICIP 2007)*, San Antonio, TX, USA, 2007.

[167] V. Thirumalai, I. Tosic, and P. Frossard, "Symmetric distributed coding of stereo omnidirectional image," *Signal Processing: Image Communication*, vol. 23, no. 5, pp. 379–390, June 2008.

[168] M. Tagliasacchi, G. Prandi, and S. Tubaro, "Symmetric distributed coding of stereo video sequences," in *Proc. of IEEE International Conference on Image Processing (ICIP 2007)*, San Antonio, TX, USA, 2007.

[169] D. Varodayan, A. Mavlankar, M. Flierl, and B. Girod, "Distributed grayscale stereo image coding with unsupervised learning of disparity," in *Proc. of IEEE Data Compression Conference (DCC 2007)*, Snowbird, Utah, 2007.

[170] D. Chen, D. Varodayan, M. Flierl, and B. Girod, "Wyner-ziv coding of multiview images with unsupervised learning of two disparity," in *Proc. of IEEE International Conference on Multimedia and Expo (ICME 2008)*, Hannover, Germany, 2008.

[171] Y. Oohama, "Gaussian multiterminal source coding," *IEEE Transaction on Information Theory*, no. 11, pp. 1912–1923, 1997.

[172] ——, "The rate-distortion function for the quadratic gaussian ceo problem," *IEEE Transaction on Information Theory*, no. 5, pp. 1057–1070, May 1998.

[173] H. Viswanathan and T. Berger, "The quadratic gaussian ceo problem," *IEEE Transaction on Information Theory*, no. 9, pp. 1549–1559, 1997.

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

[175] C. Lei, J. Selzer, and Y. H. Yang, "Region-tree based stereo using dynamic programming optimization," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, New York, NY, USA, 2006, pp. 2378–2385.

[176] JVT Reference Software JM73 [online], Available: http://iphome.hhi.de/suehring/tml/download/old_jm/jm73.zip.

[177] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, March 2008.