Side Information Exploitation, Quality Control and Low Complexity Implementation for Distributed Video Coding

Min Zheng

A Thesis Submitted for the Degree of Doctor of Philosophy

School of Engineering and Informatics

University of Sussex

September 2013

University of Sussex

Thesis Submitted in Fulfilment of the Requirements for the Degree of

Doctor of Philosophy

[Side Information Exploitation, Quality Control and Low Complexity Implementation for Distributed Video Coding]

By: Min Zheng

Summary

Distributed video coding (DVC) is a new video coding methodology that shifts the highly complex motion search components from the encoder to the decoder, such a video coder would have a great advantage in encoding speed and it is still able to achieve similar rate-distortion performance as the conventional coding solutions. Applications include wireless video sensor networks, mobile video cameras and wireless video surveillance, etc. Although many progresses have been made in DVC over the past ten years, there is still a gap in RD performance between conventional video coding solutions and DVC. The latest development of DVC is still far from standardization and practical use. The key problems remain in the areas such as accurate and efficient side information generation and refinement, quality control between Wyner-Ziv frames and key frames, correlation noise modelling and decoder complexity, etc.

Under this context, this thesis proposes solutions to improve the state-of-the-art side information refinement schemes, enable consistent quality control over decoded frames during coding process and implement highly efficient DVC codec.

This thesis investigates the impact of reference frames on side information generation and reveals that reference frames have the potential to be better side information than the extensively used interpolated frames. Based on this investigation, we also propose a motion range prediction (MRP) method to exploit reference frames

and precisely guide the statistical motion learning process. Extensive simulation results show that choosing reference frames as SI performs competitively, and sometimes even better than interpolated frames. Furthermore, the proposed MRP method is shown to significantly reduce the decoding complexity without degrading any RD performance.

To minimize the block artifacts and achieve consistent improvement in both subjective and objective quality of side information, we propose a novel side information synthesis framework working on pixel granularity. We synthesize the SI at pixel level to minimize the block artifacts and adaptively change the correlation noise model according to the new SI. Furthermore, we have fully implemented a state-of-the-art DVC decoder with the proposed framework using serial and parallel processing technologies to identify bottlenecks and areas to further reduce the decoding complexity, which is another major challenge for future practical DVC system deployments. The performance is evaluated based on the latest transform domain DVC codec and compared with different standard codecs. Extensive experimental results show substantial and consistent rate-distortion gains over standard video codecs and significant speedup over serial implementation.

In order to bring the state-of-the-art DVC one step closer to practical use, we address the problem of distortion variation introduced by typical rate control algorithms, especially in a variable bit rate environment. Simulation results show that the proposed quality control algorithm is capable to meet user defined target distortion and maintain a rather small variation for sequence with slow motion and performs similar to fixed quantization for fast motion sequence at the cost of some RD performance.

Finally, we propose the first implementation of a distributed video encoder on a Texas Instruments TMS320DM6437 digital signal processor. The WZ encoder is

efficiently implemented, using rate adaptive low-density-parity-check accumulative (LDPCA) codes, exploiting the hardware features and optimization techniques to improve the overall performance. Implementation results show that the WZ encoder is able to encode at 134M instruction cycles per QCIF frame on a TMS320DM6437 DSP running at 700MHz. This results in encoder speed 29 times faster than non-optimized encoder implementation. We also implemented a highly efficient DVC decoder using both serial and parallel technology based on a PC-HPC (high performance cluster) architecture, where the encoder is running in a general purpose PC and the decoder is running in a multicore HPC. The experimental results show that the parallelized decoder can achieve about 10 times speedup under various bit-rates and GOP sizes compared to the serial implementation and significant RD gains with regards to the state-of-the-art DISCOVER codec.

Acknowledgements

There are far too many to thank individually, but the efforts of the following are crucial. My deepest gratitude is to my supervisor Dr. Falah H. Ali, from whom I have learned the arts of making research compelling and engaging. Every facet of this work reflects his wisdom, thoughtful comments, and also his insights into the ideas. The contributions of this thesis are the results of numerous discussions with him. His rich research experiences and affability contributed to make this journey enriching and pleasant.

All my colleagues in the Communications Research Group were sources of inspiration and friendship. It was sheer enjoyment to work with Walid, Marwan, Bilal, Saif, Fei, Tom, Ibrahim and Murtala. We shared, not just ideas, but the daily ups and downs of research life. Special thanks to Marwan for his warm words of support and for his valuable comments and suggestions on this thesis.

Since arriving at Brighton, I have had the companionship and encouragement of friends, old and new. There were my long-time friends Bo Yao and Lei Zou from China, my previous landlord Paul and Marie, my ex-girlfriend Sahar Delikhan and her family. I befriended many wonderful people in Brighton, among them Eric, Audrey and Roger, Xiaolin Zhang, Frank, Xiaolei, Jie Li, Zuhal, Paula, Zahara, Maria, Jorge, Hussam, Jamila, Szevia and Nadia.

My deepest and forever gratitude is for my family. My parents raised me across continents to think across boundaries. My brother and I share a lifelong dialogue that influences me a lot. I thank them for all their love, understanding, efforts in support of my success and patience.

I would also like to thank to Zebra Geosciences Ltd. for offering me the job and financial support. And also thank to all the colleagues in this company for the relax and pleasant working environment and their kind assistance and advice on technical issues.

List of Publications

- M. Zheng, F. H. Ali, “DSP Implementation of On-Board Distributed Video Coding,” 4th European DSP Education and Research Conference, EDERC2010, 5 pages, Nice, France, Dec. 2010.
- M. Zheng, F. H. Ali, “Consistent Quality Control for Wireless Video Surveillance Using Distributed Video Coding,” The 4th International Conference on Imaging for Crime Detection and Prevention (ICDP-11), London, UK, Dec. 2011.
- M. Zheng, F. H. Ali, “Exploration and Exploitation of Reference Frames in Distributed Video Coding,” IEEE Signal Processing Letters, vol. 19, issue 7, pp. 411-414.
- M. Zheng, F. H. Ali, “Pixel Granularity Side Information Synthesis Framework and Parallel Implementation for Distributed Video Coding,” submitted to IEEE Transactions on Multimedia.

Table of Contents

List of Abbreviations

2D	Two dimensional
AVC	Advanced video coding
CNM	Correlation noise modelling
CRC	Cyclic redundancy check
CRG	Communications Research Group
CUDA	Compute Unified Device Architecture
dB	Decibel
DC	Direct current
DCT	Discrete cosine transform
DISCOVER	Distributed coding for video services
DISCUS	Distributed source coding using syndromes
DQ	Distortion-quantization
DSC	Distributed source coding
DSP	Digital signal processing
DVC	Distributed video coding
EM	Expectation maximization
FIR	Finite impulse response
GOP	Group of pictures
GPGPU	General-Purpose computing On Graphics Processing Units
GPU	Graphics processing unit
HPC	High performance cluster
IDCT	Inverse discrete cosine transform
i.i.d	Independent and identically distributed
ISO	International Organization for Standardization

ITU-T	International Telecommunications Union – Telecommunications Standardization Sector
JPEG	Joint Photographic Experts Group
LDPC	Low density parity check
LDPCA	Low density parity check accumulate
LSB	Least significant bit-plane
MAD	Mean absolute difference
MAE	Mean absolute error
MAP	Maximum a posteriori
MCFI	Motion compensated frame interpolation
MCI	Motion compensated interpolation
MMSE	Minimum mean square error
MPEG	Motion Picture Experts Group
MRP	Motion range prediction
MSB	Most significant bit-plane
MSE	Mean square error
MV	Motion vector
OpenMP	Open Multi Processing
PC	Personal computer
PDWZ	Pixel domain Wyner-Ziv
PGSIS	Pixel granularity side information synthesis
PRISM	Power-efficient, robust, high-compression, syndrome-based multimedia coding
PSNR	Peak signal to noise ratio
QCIF	Quarter common intermediate format for images (144 lines ×

	176 columns)
QP	Quantization parameter
RD	Rate distortion
SAD	Sum of absolute differences
SI	Side information
SISO	Soft-input, Soft-output
SMF	Statistical motion fields
SPA	Sum product algorithm
TDWZ	Transform domain Wyner-Ziv
UC	University of California
VLC	Variable length coding
WZ	Wyner-Ziv
XOR	Exclusive or

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Background and Motivations

Video applications are extensively used nowadays which has encouraged the deployment of multimedia products such as mobile phones, digital cameras, DVD systems and many other digital devices and software products. These applications require large amount of video data storage or transmission, and therefore efficient compression of video data is important. Since data transmission over network, especially wireless networks, is prone to errors, compression algorithms with good error resilience properties are desired. Furthermore, emerging applications such as wireless and handheld devices are tend to be small in size and restricted by their battery life and computational resources. Therefore, low complexity processing, low power consumption and simple implementation for such applications are necessary as they cannot afford to run complex routines. Conventional video coding schemes, such as MPEG-x and H.26x [1][2], use predictive coding techniques to exploit the correlation between adjacent video frames. This results in computationally intensive encoders due to high complexity of the encoder side motion search component. In contrast, the decoders are usually much simpler. This type of architecture succeeds in a wide range of down link model applications such as video broadcasting and video-on-demand, where the cost of the decoder is critical. However, the predictive coding strategies are not suitable for the aforementioned emerging applications that requiring simple but still efficient encoders, where the power consuming of the encoder is critical. Please note that unless specifically defined in this thesis, the term “complexity” herein refers to the number of computational operations.

A new coding paradigm, Distributed Video Coding (DVC), emerged under this circumstance. It shifts the major computation (i.e. the motion search component), partially or fully, from encoders to the decoder [3]. A DVC codec typically divides the video sequence into two kinds of frames: key frames and Wyner-Ziv (WZ) frames. Key frames are inserted periodically, depending on the group of picture sizes (GOP). They are typically intra-coded by a conventional coding solution, whereas the WZ frames are coded by the DVC principle. These two kinds of frames can be separately encoded without any reference to each other, but still achieve similar or even the same coding efficiency as the conventional coding approach. This novel feature enables simple but still efficient encoders. At the decoder side, one or more already decoded frames serve as side information, providing a noisy version of the WZ frame and their correlation are modelled and exploited. The decoder complexity can be reduced by properly increase the number of key frames, thus reducing the decoding of WZ frames and therefore it allows the encoder to share the overall complexity depending on the target platforms and applications. This feature enables flexible adjustment of complexity between encoders and decoders. And the hybrid video coding architecture is not only compatible with most conventional "down-link" applications, but also benefits "uplink" applications.

1.2 Aims and Objectives

The main objective of this thesis is to investigate, develop and evaluate new, more efficient and more practical solutions for DVC, thus bridge the gap between theory approach and realistic applications, and bring the state-of-the-art DVC codec one step closer to practical use, particularly through the proposal of the following methods.

- In-depth investigation and analysis of the impact of using reference frames as side information on the coding efficiency in terms of RD performance and decoding complexity;
- Efficient motion search technique to exploit the correlation between the reference frames and WZ frames;
- Finer granularity side information refinement framework for high quality side information generation;
- Adaptive correlation noise modelling for updated side information;
- Efficient rate distortion model to achieve consistent quality of decoded frames;
- Highly efficient serial and parallel DVC implementations for practical video coding systems.

All of above are based upon the investigation and evaluation of existing research works. The proposed methods are validated carefully before the implementation and a systematic test and measurement are carried out to show the correctness and the efficiency of our proposals.

1.3 Original Contributions

This thesis proposes solutions to improve the state-of-the-art side information refinement schemes, enable consistent quality control over decoded frames and implement highly efficient DVC codecs. The main contributions of this thesis are summarized below.

1. This thesis investigates the impact of reference frames on side information generation and reveals that reference frames have the potential to be better side information than the extensively used interpolated frames. Based on this investigation, we propose a motion range prediction (MRP) method to exploit

reference frames and precisely guide the statistical motion learning process. Extensive simulation results show that choosing reference frames as SI performs competitively, and sometimes even better than interpolated frames. Furthermore, the proposed MRP method is shown to significantly reduce the decoding complexity without degrading any RD performance.

2. To minimize the block artifacts and achieve consistent improvement in both subjective and objective quality of side information, we propose a novel side information synthesis framework working on pixel granularity. We synthesize the SI at pixel level to minimize the block artifacts and adaptively change the correlation noise model according to the new SI. Furthermore, we have fully implemented a state-of-the-art DVC decoder with the proposed framework using serial and parallel processing technologies to identify bottlenecks and areas to further reduce the decoding complexity, which is another major challenge for future practical DVC system deployments. The performance is evaluated based on the latest transform domain DVC codec and compared with different standard codecs. Extensive experimental results show substantial and consistent rate-distortion gains over conventional standard video codecs and significant speedup over serial implementation.
3. In order to bring the state-of-the-art DVC one step closer to practical use, we address the problem of distortion variation introduced by typical rate control algorithms, especially in a variable bit rate environment. Simulation results show that the proposed quality control algorithm is capable to meet user defined target distortion and maintain a rather small variation for sequence with slow motion and performs similar to offline fixed quantization settings for fast motion sequence at the cost of some RD performance.

4. Finally, we propose the first implementation of a distributed video encoder on a Texas Instruments TMS320DM6437 digital signal processor. The WZ encoder is efficiently implemented, using rate adaptive low-density-parity-check accumulative (LDPCA) codes, exploiting the hardware features and optimization techniques to improve the overall performance. Implementation results show that the WZ encoder is able to encode at 134M instruction cycles per QCIF frame on a TMS320DM6437 DSP running at 700MHz. This results in encoder speed 29 times faster than non-optimized encoder implementation. We also implemented a highly efficient DVC decoder using both serial and parallel technology based on a PC-HPC (high performance cluster) architecture, where the encoder is running in a general purpose PC and the decoder is running in a multicore HPC. The experimental results show that the parallelized decoder can achieve about 10 times speedup under various bit-rates and GOP sizes compared to the serial implementation and significant RD gains with regards to the state-of-the-art DISCOVER codec.

1.4 Outline of the Thesis

This thesis is organized as follows. This chapter presents the background and the motivations, along with the main objectives of our work, highlights of the original contributions and the structure of the thesis.

Chapter 2 reviews the theoretical background supporting the WZ video coding as well as a comprehensive literature review on the state-of-the-art DVC performance and research topics that are directly relevant to this thesis. These include the latest developments of side information generation and refinement, correlation noise modelling, consistent quality control for decoded video frames and practical DVC implementations.

Chapter 3 investigates the impact of reference frames in DVC on the RD performance and reveals that reference frames have the potential to be better side information than the extensively used interpolated frames. Based on this investigation, we propose a motion range prediction method to exploit reference frames and precisely guide the statistical motion learning process.

A novel SI synthesis framework based on pixel granularity is proposed in Chapter 4. We synthesize the SI at pixel level to minimize the block artifacts and adaptively change the correlation noise model according to the new SI. The decoding complexity is another major research challenge in practical DVC system deployments. We have fully implemented a state-of-the-art DVC decoder with the proposed framework using both serial and parallel processing technologies. The performance is evaluated based on the latest transform domain DVC codec and compared with different standard video codecs.

In Chapter 5, we propose a novel algorithm to facilitate quality controls for both key frames and WZ frames. The proposed algorithm adjusts the quantization parameters according to the visual content and the user defined target quality online without any external control. A distortion-quantization model derived from MPEG-2 distortion estimation model is employed. With the proposed algorithm, low complexity encoding is still guaranteed by performing the distortion estimation partly at the decoder side. The proposed algorithm addresses the problem of distortion variation introduced by typical rate control algorithms, especially in a various bit rate environment.

Chapter 6 proposes the first implementation of a distributed video encoder on a Texas Instruments TMS320DM6437 digital signal processor. The WZ encoder is efficiently implemented, using LDPCA codes, exploiting the hardware features and optimization techniques to improve the overall performance. This chapter also presents

a highly efficient DVC decoder using both serial and parallel technology based on a PC-HPC (high performance cluster) architecture, where the encoder is running in a general purpose PC and the decoder is running in a multicore HPC. Both the encoder and the decoder are carefully evaluated and compared with the state-of-the-art codecs.

Finally, Chapter 7 summarizes the main achievements of this thesis and identifies possible areas for our future works.

Chapter 2
Overview of Distributed Video Coding

2.1 Introduction

DVC has been evolving significantly since the first practical solutions. However, this coding paradigm is still relatively new and its latest development shows that it is still far from standardization and industrial deployments, although compared to the conventional coding solutions, the coding efficiency of DVC has achieved similar performance.

This chapter reviews the recent status and trends in distributed video coding. The foundation of DVC including the Slepian-Wolf and Wyner-Ziv theorem are presented in section 2.2. After the theorem introduction, section 2.5 is devoted to the early development of DVC architectures, mainly the Berkeley DVC architecture and the Stanford architecture. The current research challenges are summarized in section 2.4 and section 2.5, the latest developments of DVC in terms of overall performance and relevant research areas including SI generation, SI refinement, correlation noise modelling, consistent quality control and fast DVC implementations are reviewed. Finally in section 2.6, we conclude this chapter.

2.2 Theoretical Background

In classic video coding standards, such as MPEG-x or H.26x recommendations [1][2], predictive coding techniques exploiting the statistics of the video contents are adopted at encoder side. This brings intensive computational complexity and thus sets rather high requirements on the hardware performance at the encoder, whereas the decoder is very straightforward and simple. This architecture suits well for most

"downlink" or "storage" scenarios which only compress the data once but can be streamed to multiple terminals and decompressed whenever requested, as depicted in Figure 2.1. Live video can be captured and sent to a central storage server to be encoded. The encoded data is typically stored offline for future streaming requests. This scenario can be characterized as a one-to-many video coding paradigm with highly complex front-ends but allows multiple simple terminals. It emphasizes the reuse of encoded data resources as video compression is far less frequent than video decoding in this case. Typical applications include internet video streaming and broadcastings, video surveillance, etc.

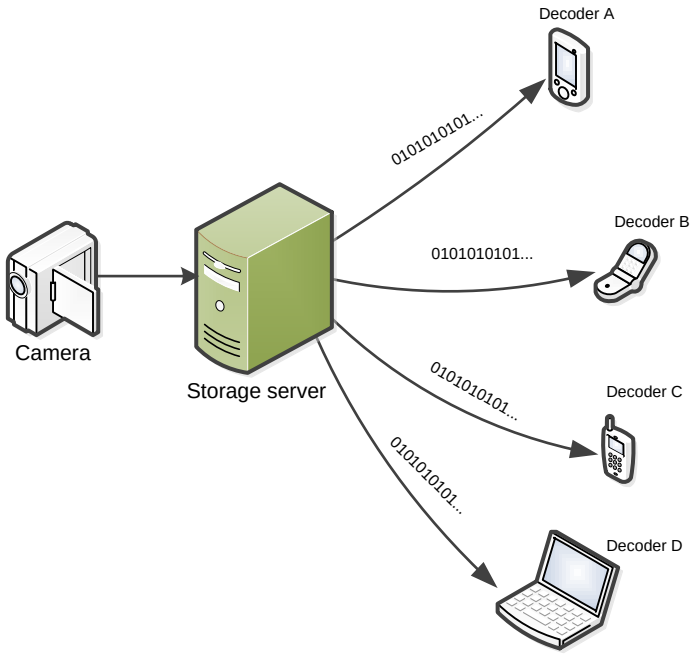


Figure 2.1 Conventional "down-link" model applications

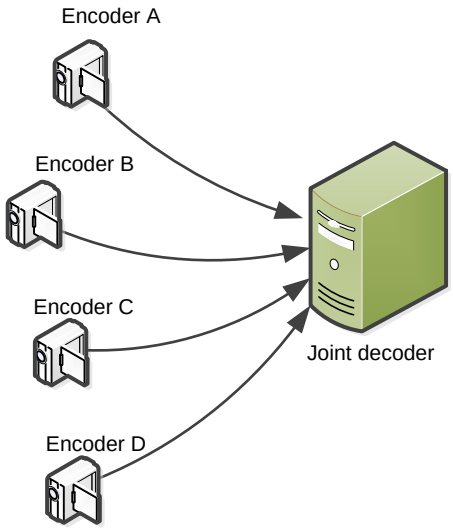


Figure 2.2 DVC provide "up-link" model applications

Distributed video coding aims at very low complexity encoding but still achieve the same or similar coding efficiency as the conventional solutions. DVC shifts the major computational component (i.e. the motion search module) from encoder to decoder side. This novel insight enables video compression ability in resource critical devices which is currently limited or even impossible for conventional coding solutions. Figure 2.2 shows a typical "up-link" application scenario where the power restricted devices are now able to upload captured video data efficiently.

However, for real-time applications, decoders also have complexity restraint and thus a transcoder is required to guarantee that both encoder and decoder are of low complexity. Figure 2.3 shows the use of a transcoder to convert the decoded video data from a DVC decoder into a conventional video codec such as H.264/AVC encoder. Real-time decoding can therefore be achieved at the decoder side as well.

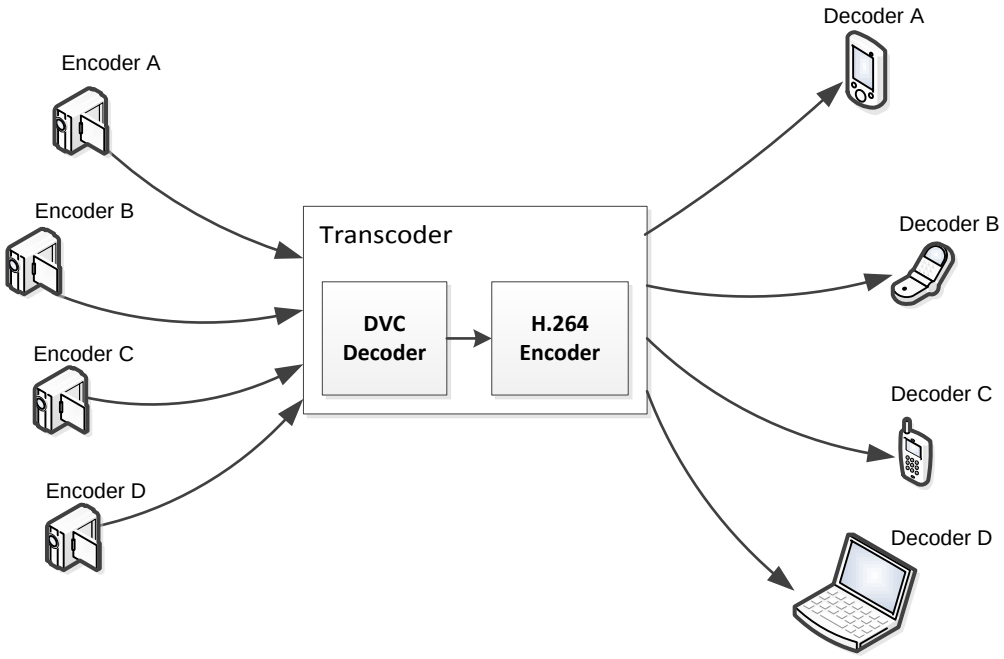


Figure 2.3 Use of a transcoder to achieve real-time video coding

The theoretical foundations of distributed video coding are based on Slepian-Wolf theorem [6] in which the entropies of correlated information are proposed and it also shows that two isolated sources can be compressed as efficiently as if they were communicating with each other. Shortly after this finding, Aaron D. Wyner and Jacob Ziv extended this theorem to lossy compression with decoder side information [7].

2.2.1 Slepian-Wolf Coding

The Slepian-Wolf theorem gives the rate bound to reconstruct the correlated data with arbitrarily small error probability. Consider two independent and identically distributed (i.i.d.) sequences X and Y. Shannon's source coding theory [8] indicates that a rate of joint entropy $H(X|Y)$ is sufficient to compress X and Y losslessly based on the complete knowledge of X and Y at a single encoder, whereas Slepian and Wolf showed that this rate can still be achieved even X and Y are compressed separately by independent encoders. The Slepian-Wolf theorem shows that to recover separately encoded X and Y losslessly, a rate of $R_X + R_Y = H(X,Y)$ is sufficient if $R_X \geq H(X|Y)$

and $R_Y \geq H(Y|X)$. These inequalities form the achievable rate region [6], given by Figure 2.4.

The top right region in dark grey is the rate region for conventional coding solutions that encode X and Y separately without exploiting their correlation. Special cases can be seen from the corner points of the rate region which is commonly referred to as compression with decoder side information, where one data source is available at the decoder side but not accessible at the encoder side, e.g. trying to achieve a rate of $H(X|Y)$ when encode X, while a rate of $H(Y)$ has been used to encode Y.

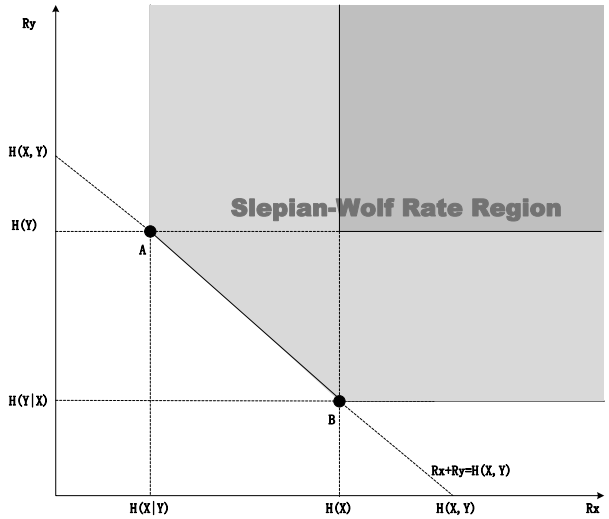


Figure 2.4 The Slepian-Wolf rate region

Since one of the two correlated data can be seen as a noisy version of the other obtained through a virtual correlation noise channel, Slepian-Wolf coding can therefore relate to channel coding. A Slepian-Wolf codec can be implemented using efficient channel codes such as Turbo codes [13] and LDPC codes [14] given particular correlation models.

2.2.2 Wyner-Ziv Coding

In 1975, Wyner and Ziv extended Slepian and Wolf's work to lossy coding with decoder side information scenario [7]. The theorem gives the lower rate bound for encoding Gaussian memoryless source [15] under the constraint of mean squared error (MSE) distortion. And this rate bound will not change even if the side information is not available at the encoder side, i.e. there is no coding efficiency loss when the side information is only available at the decoder.

Lossy compression is usually obtained by introducing quantizers. Therefore, a practical Wyner-Ziv codec can be seen as a Slepian-Wolf codec with a quantizer and a de-quantizer.

2.3 Early DVC Architectures

Although theories state that DVC solutions can perform as efficient as joint coding solutions, the practical DVC architectures only came out a decade ago. Among them, the early DVC architectures developed by UC Berkeley and Stanford research groups remain the most popular architectures nowadays.

2.3.1 The Berkeley DVC Architecture

The first attempt to design a practical DVC started in 2002, i.e. PRISM codec (Power-efficient, Robust, hIgh compression Syndrome-based Multimedia coding) [9][10]. Its architecture is shown in Figure 2.5. Input frames are divided into 8×8 blocks and DCT transformed. At the same time, zero-motion block differences are used to evaluate the correlation level between neighbouring frames, which result in 16 different encoding classes. For instance, blocks with very low correlation are encoded using conventional Intra-coding method, whereas blocks with very high correlation are simply

skipped without coding. The remaining blocks are encoded based on DVC principles. The estimated correlation levels are also utilized to determine the number of least significant bits (LSB) of the transform coefficients, and syndrome bits are generated from them. The lowest bit planes in the LSB are encoded using standard entropy coding principles with a (run, depth, path, last) 4-tuple alphabet. The higher bit planes in the LSB are coded using channel codes. And BCH block codes are chosen for their good performance on small block-lengths. With regards to the most significant bits (MSB), they can be derived from the block predictor or SI. In order to check successful decoding, a 16-bit Cyclic Redundancy Check (CRC) [16] is calculated for quantized code words at encoder. At the decoder side, the syndrome bits are then used to correct predictors, which are generated from the motion search module.

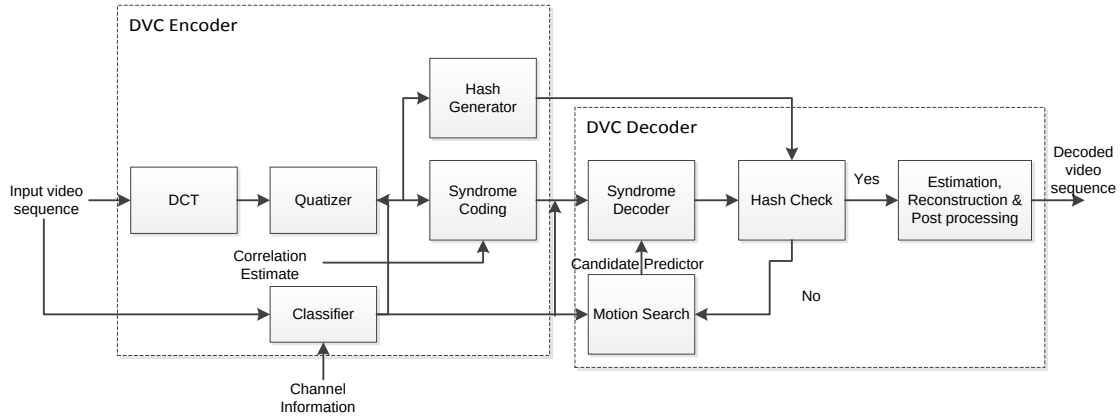


Figure 2.5 PRISM DVC Architecture

2.3.2 The Stanford DVC Architecture

Almost at the same time, Rane, Aaron and Girod proposed another DVC architecture [11]. The functional blocks of this architecture is shown in Figure 2.6. The input video sequence is split into key frames and WZ frames. Key frames are encoded using a conventional video coding solution such as H.264/AVC Intra [12]. The WZ frames are encoded using distributed video coding principles. WZ frames are quantized without DCT transform, which is usually referred to as pixel-domain DVC architecture

(whereas transform-domain DVC architecture refers to DVC with coding of pixels in a transformed form). Bit-planes are then extracted from the quantized symbols which will feed to a Turbo encoder. The Turbo encoder generates parity bits and they are stored in a buffer for the decoder requests.

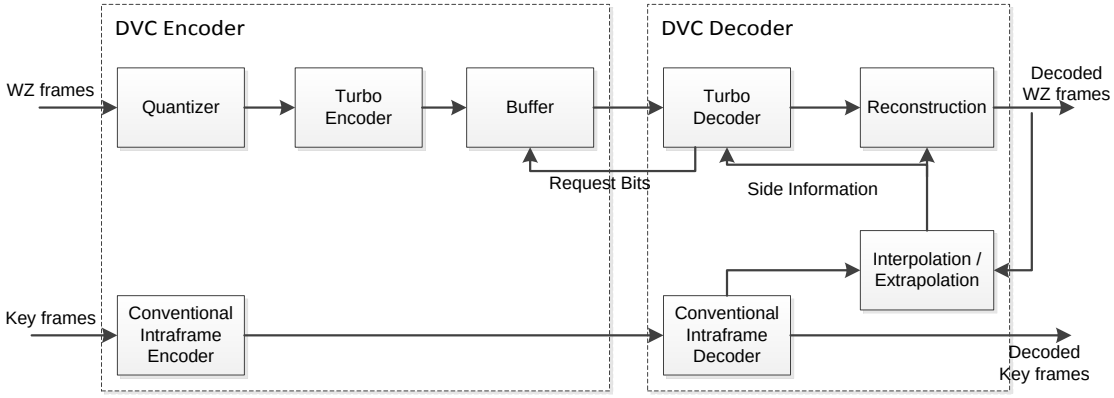


Figure 2.6 Stanford DVC Architecture

At the decoder side, motion-compensated frame interpolation or extrapolation using previously decoded frames is performed to generate SI. The turbo decoder then correct the errors in the SI using the parity bits requested from encoder buffer via a feedback channel. Finally, bit-planes of WZ frames are reconstructed, and decoded WZ frames and key frames are re-ordered to form the decoded video sequence.

The above two architectures are still the main structural designs for modern DVC codecs implementations. However, the features in one architecture can sometimes be used in the other. Most modern DVC codecs nowadays actually combine the modules from the two. The fundamental differences between them are highlighted in Table 2.1.

The most obvious difference is that PRISM uses different coding mode according to the block correlation which allows for a better adaptation of various local textures of video content. In this way, the WZ coding mode is only used when the correlation is sufficient since WZ coding performs poorly for intense motion or scene changes. Although block classification by simple inter-frame prediction does not dramatically

increase encoder complexity, block partitioning results in short block-length which impairs efficient channel coding. BCH codes [17][18] are therefore used for this reason. On the contrary, Stanford solution encodes the entire WZ frame without block partitioning and classification. However, more efficient channel codes, such as Turbo or LDPC codes, were used to improve the coding efficiency.

Table 2.1 Stanford Architecture vs. Berkeley Architecture

Techniques \ Architecture	Stanford Architecture	Berkeley Architecture
Coding unit	Frame	Block
Block classification	No	Yes
Rate control	Decoder side	Encoder side
Channel codes	LDPC codes	Turbo codes
Auxiliary data	None	Hash codes
Use of motion information	Initial SI generation	Candidate SI decoding

Rate control mechanisms used in these two frameworks form another important fundamental distinction. PRISM removed the feedback channel by estimating a minimum rate at encoder side. This is key to reducing the decoder complexity, although false estimation may cause some coding performance loss. On the contrary, the Stanford approach relies on the feedback channel to achieve better coding efficiency. Although a feedback channel can allow virtual noise adaptation and achieve optimal coding rate, it is usually considered to be not practical in real-time applications, and the complexity it brings to the decoder side is tremendous.

Furthermore, motion estimation is also performed in different ways. The Stanford architecture estimates motion when generating the SI. Motion estimation between the reference frames is performed at the decoder side which can provide a good estimate of

the motion vectors between SI and WZ frame for slow motion video content but the accuracy may decrease for intense motion scenarios. On the contrast, PRISM searches over the space of candidate motion vectors and each candidate associates to a motion compensated SI. The SI that successfully decodes the syndrome bits and passes the CRC check is believed to have associated with the best-matched motion vector. A more detailed comparison can also be found in [5].

2.4 Research Challenges

Although numerous work has been done to improve the RD performance and speed up the practical use of DVC, the main challenges remain in the areas of the following.

2.4.1 Side information generation

Side information can be seen as a noisy version of the WZ frames, therefore the aim of side information generation is to create an estimate of the WZ frame that is as similar as possible. The quality of SI has a direct impact on the final RD performance as well as the decoding complexity since the better the SI is, the less error it contains and thus fewer parity bits are required for decoding. SI is typically generated by frame interpolation or extrapolation of reference frames, taking into account the motion activities. However, this estimation can be very challenging since the motion information is not necessary consistent and smooth over time, and scene changes or intense motion can seriously affect the accuracy of interpolation/extrapolation based methods. Furthermore, coding with long GOP sizes can also lead to poor SI quality.

2.4.2 Side information refinement

Transform domain DVC usually converts video frames into bands, and decoding is carried out band by band. As one band is successfully decoded, it provides information

not initially available to the decoder. With the help of this information, SI can be refined gradually and thus improve the coding efficiency for decoding the rest of the frame. The refinement can be significant, especially when the motion is intense or scene changes occur since interpolation/extrapolation for the initial SI generation performs poorly under these situations. However, the selection of areas in SI is essential since not the entire SI requires update, i.e. some regions would not change over time and any updates on these regions may bring even poorer SI and increase decoding complexity at the same time. Furthermore, any changes in SI will affect accuracy of the conditional bit probability model and the correlation noise model established earlier for the initial SI. The improvements in SI quality during the decoding process may not necessarily transfer to the final RD performance gains if the other related modules are not coordinated well.

2.4.3 Correlation noise modelling

Since SI can be created at the decoder side, with the knowledge of the correlation between SI and WZ frames, WZ frames can be decoded. This is very similar to the channel coding scenarios where WZ frames are “transmitted” through a virtual channel and the SI can be seen as the received version with transmission noise. This virtual noise is actually a form of the correlation information and is typically following a Laplacian distribution [4][5][19][26]. Correlation noise model is used to estimate the noise distribution and it also has a direct impact on the final RD performance and decoding complexity. The main challenge here is that WZ frames are unknown at the decoder side. Therefore, the correlation between WZ frames and SI has to be estimated from reference frames, which can be very unreliable since the differences between reference frames do not directly reflect the difference between WZ frames and SI.

Furthermore, SI might be updated during decoding process, and the correlation noise will be changing accordingly, which makes the estimation even harder.

2.4.4 Consistent Quality Control

Most of the existing DVC solutions use pre-defined quantization parameters for coding both key frames and WZ frames [3][4][5][11]. And these parameters are typically obtained from extensive offline experiments trying to achieve a consistent image quality over time. However, this is not practical for real-time applications. Therefore, it is necessary to design a rate control algorithm to coordinate the quantization settings for both key frames and the WZ frames. The major challenges are that the key frames and the WZ frames are coded independently by different codecs. Therefore the quantization settings have to be estimated separately for both of them. Furthermore, the major advantage of DVC is the low encoding complexity and hence the controlling algorithm should not add major complexity burden to the encoder.

2.4.5 DVC implementation

Although enormous solutions have been proposed to tackle various DVC problems, the same solution can be interpreted differently in implementations. The efficiency of different implementations, in terms of coding and complexity performances can vary dramatically. In addition, the transference of complexity from encoder to decoder plus various refinement algorithms added on the top can make the already slow decoder overburdened. This is usually overlooked by the research community but it is a critical problem for practical DVC applications, especially for real-time scenarios. Therefore, an efficient implementation of DVC codec is essential and deserves more attentions. However, since the encoder and the decoder are usually targeted at different hardware platforms, it requires the design and the implementations to consider the restriction on

both of the software and hardware, available resources, portability of the codes, communications between the encoder and decoder platforms, etc. Furthermore, different interpretations of the algorithms will results in different implementations, and the correctness and the efficiency of the implementations may need to be tested by extensive experiments.

2.5 Relevant Recent Advances on DVC

2.5.1 State-of –the-art Performance

Developed in 2005, the European project DISCOVER [19], is one of the best performing DVC codecs reported in the literature to date. It is based on the Stanford architecture [11] but a lot of improved modules have been integrated. [20] presents a comprehensive evaluation of this codec. This transform domain Wyner-Ziv codec introduced a hybrid bit-rate control mechanism which operates at both the encoder and the decoder sides. The encoder estimates a minimum rate budget and if it is not sufficient, the decoder can request more parity bits from the encoder buffer through a feedback channel. This allows a great reduction on decoder complexity and also enables rate adaptability. Furthermore, a motion vectors smoothing algorithm was applied to motion compensated interpolation in order to generate SI with enhanced quality. Notably, the correlation noise distribution was online estimated and more advanced channel codes, i.e. LDPCA codes [21][22] are used at the decoder. The decoded symbols are reconstructed in a mean squared error-optimal way [23]. The reported RD performance shows that the DISCOVER codec consistently outperforms H.264/AVC Intra, except high motion video content such as the “Soccer” sequence. For low motion video content such as “Hall Monitor”, up to 3 dB gains can be observed. When compared to H.263+ Intra codec [24], a remarkable 8 dB gains can be observed for the

“Hall Monitor” sequence. However, there is still some gaps between the RD performance of the DISCOVER codec and H.264/AVC No Motion [12].

More advanced DVC codec was later developed by the VISNET II project [25].This codec is also based on the early architecture in [11] and integrates numerous advanced tools. The major improvements over DISCOVER codec are mainly brought by the iterative SI refinement method and the deblocking filter. After each DCT band is decoded, the partially decoded WZ frame is exploited to refine the SI and also provides better reconstructed WZ frame. After frame reconstruction, an adaptive deblocking filter is used to improve the subjective and objective quality of WZ frame. In terms of RD performance, the VISNET II codec consistently outperforms the DISCOVER codec for all the test sequences and various bit-rates. Gains of up to 5 dB can be achieved over H.264/AVC Intra for low motion content such as Hall Monitor. However, although for video sequences with regular global motion, such as Coastguard sequence, VISNET II still can achieve better RD performance over H.264/AVC No Motion, for most other video sequences, the performance of the VISNET II codec is still significantly lower. Regarding to complexity, [26] shows that the DVC encoding complexity in terms of software execution time is about 1/6 of the average encoding time of H.264/AVC Intra and H.264/AVC No Motion.

The best performing DVC codecs, as far as the author can check, are presented in [27] and [28]. [27] estimates the parameters of the global motion at the encoder using scale invariant feature transform and combines the global and local motion compensation at the decoder side. Those encoder estimated parameters are sent to the decoder in order to generate a globally motion compensated side information. Based on motion-compensated temporal interpolation of neighbouring reference frames, it also generates a locally motion compensated side information. And finally, an improved

fusion of global and local side information can be obtained during the decoding process using the partially decoded Wyner–Ziv frame and decoded reference frames. The presented experimental results show that when compared to DISCOVER codec, this method can achieve a RD performance gain of up to 1.92 dB for GOP size 2 and a remarkable 4.65 dB for longer GOP sizes. More impressing and encouraging is that, DVC now outperformed H.264/AVC Intra or H.264/AVC No motion in all reported test conditions. And the performance gap between the proposed DVC scheme and H.264/AVC Inter prediction with motion is considerably reduced. However, the astonishing RD performance gains mainly come from the motion information provided by the encoder. Therefore, strictly speaking, this codec is not a pure DVC codec since the correlated information, WZ frames and Key frames are not "distributed". There is information exchange between them. With that being said, however, for practical system design and implementation, it is still highly recommended to partially rely on the encoder to analyse motion. In contrast, [28] does not perform any motion search at the encoder side. It used optical flow to compensate the weaknesses of using block-based methods to improve side information generation, and it also introduced clustering algorithm to capture cross band correlation and increase local adaptivity in the noise modelling. In addition, multiple techniques are combined to calculate several candidates of soft side information for channel decoding. This method can achieve 1.53 dB improvement in average on RD performance over the DISCOVER codec for the most difficult test sequence (Soccer) using GOP size 2. The RD performance gains are mainly achieved by multi-hypothesis based decoding method. However, the decoding complexity can increase significantly along with the increase of the number of side information.

2.5.2 Side Information Generation

It is well known that the performance of DVC highly depends on the quality of side information. Early attempts to generate SI are rather simple and intuitive. In 2002, the Stanford research group proposed to use the average of the reference frames to generate the SI [11]. This method exploited limited correlation between the reference frames since motion information was not taken into account, and therefore it has much poorer coding efficiency. Later in 2004, they extended the previous work and evaluated different SI generation schemes [29]. Two simple SI generation schemes, average interpolation and previous frame extrapolation, are evaluated and compared with the motion compensation based interpolation methods. The simulation results show that the RD performance is about 1 to 2 dB lower for Foreman sequence when motion compensation is not performed. More recently, in 2005, the IST research group proposed a frame interpolation with motion smoothing algorithm [30]. The two reference frames are low pass filtered to reduce the noise for motion estimation. Forward motion vectors are then obtained by block based motion estimation. To avoid holes effects, it selects the motion vector that has the intercepting point closest to the centre of the non-overlapped block. Since interpolated frames are not available, bi-directional motion estimation is then performed between the two reference frames with the constraint of a linear trajectory of forward and backward motion vectors. After bi-directional motion estimation, weighted vector median filter is applied to improve the spatial coherence, and finally, bi-directional motion compensation is performed to generate the interpolated frame. This algorithm was first proposed for a pixel domain DVC codec but was later widely adopted by transform domain DVC codecs.

Although frame interpolation has been proved to perform better than extrapolation techniques, it is more suitable for real-time applications to generate SI by extrapolation

since frame interpolation breaks the original frame order and requires some future frames to be available beforehand. However, this is impossible for real-time scenario since video frames have to be decoded in sequence in this case. In 2005, [31] proposed frame extrapolation module to generate SI based on motion fields smoothening method. Firstly, forward motion vectors are estimated by 8×8 overlapped blocks using two previously decoded reference frames. Secondly, motion vectors are smoothened by calculating the average of all the neighbouring motion vectors. And finally, motion compensation is performed using the obtained motion vectors. For any overlapping pixels, average values are used. And for uncovered areas, local spatial interpolation is performed taking into account 3 neighbouring pixels. Simulation results show about 7 dB loss in RD performance for the Foreman sequence when comparing with frame interpolation method. In 2007, a more advanced frame extrapolation based SI generation method is proposed [32]. It exploited 3 previously decoded frames to generate initial motion vectors. It then generates another set of motion vectors only from the nearest two frames. Final motion vectors are chosen from these two set motion vectors by taking into account the consistency restraint, i.e. motion vectors with the lowest difference between subsequent neighbouring frames are selected. Although this solution provides better extrapolated frames, similar to other extrapolation based methods, it performs poorly for intense and inconsistent motion scenarios since motion vectors are always estimated by the information obtained from previously decoded frames.

2.5.3 Side Information Refinement

Various techniques have been proposed for SIS in the past. The recent advances on this issue can be categorized as below.

Multiple hypotheses: SIS using this method typically selects the SI candidate that first converge the decoding iterations [28][33]. The idea is to evaluate different SI in each iteration of belief propagation process [34] and choose the one that stops the iterative decoding successfully. It always guarantees the best SI to be chosen for decoding. However, the major drawback is the seriously increased complexity. The complexity of iterative decoding process increases in proportion to the number of candidate soft inputs. For the simplest case of using only 2 soft inputs, the complexity of belief propagation almost doubles. Furthermore, appropriate correlation noise models will be required to fit different noise distribution brought by different SI generation schemes.

Statistical motion learning: Motion learning is seen as an indirect method of SI update. Due to the high complexity, it usually limits the motion search in block level by estimating the probability of every possible displacement of each block. The resulting probabilities for each possible motion vectors are contributing to the new correlation noise distribution and optionally for SI refinement [35][36][37]. However, motion learning suffers from high complexity in motion search process as each possible motion field for the over-complete SI has to be evaluated to produce an accurate estimation. And each time a band has been successfully decoded, the statistical motion fields will need to be re-calculated to further update correlation noise model and SI. Additionally, this results in a highly restricted decoder design. For a typical transform domain motion learning algorithm, the motion probabilities are computed in transform domain, which requires everything relevant to be converted into transform domain as well, e.g. the generation of over-complete SI has to be converted into transform domain. All the efforts made on performing motion learning in transform domain actually results in similar or even worse motion fields as those achieved from straightforward spatial

domain approaches. In addition, if the motion learning model relies on other processing modules such as the correlation noise model, poor correlation noise estimation will results in bad motion probability distribution, whereas spatial domain motion search is generally independent, so the accuracy of the motion fields only relies on the quality of reference frames.

Spatial domain SIS: This is a straightforward technique that directly exploits the correlation from the reference frames. Spatial domain motion estimation can be further categorized into block level and pixel level motion estimation. The former saves more computational complexity whereas the later gives better precision, especially for high motion video content. Most work in the literature use block level motion estimation [38][39].

DCT domain SIS: SI refinement can also be performed in transform domain as the partially decoded WZ frame is initially obtained in transform domain before any inverse transform is carried out. In [40] and [41], motion estimation is performed between the decoded and oversampled DC frame and transformed key frames. The refined SI is synthesized considering the forward, backward, and bi-directional prediction together with the motion vectors obtained from the initial SI generation. Since band-by-band decoding model only gives a subset of all the Discrete Cosine Transform (DCT) bands after a certain band is successfully decoded each time, the motion estimation using only a small subset of the DCT bands of a single block has been reported to be not so accurate [39][42]. Therefore, [40] and [41] used block of DCT coefficients to improve the accuracy for transform domain motion search, which requires bigger block size that may not favor complex motion in local area. Complexity is also an important issue for this approach as it requires DCT transform for each over-complete SI candidate.

Bit-plane by bit-plane update [43][44]: SI can be updated each time a bit-plane is successfully decoded. However, the information provided by a small portion of decoded bit-planes is not sufficient for transform domain DVC, and is therefore more suitable for pixel domain DVC.

2.5.4 Correlation Noise Modelling

One of the most important aspects influencing the coding performance of DVC is the virtual channel noise model which is used to estimate the noise distribution between the side information and the WZ frame. There are mainly two kinds of correlation model in literatures, i.e. offline correlation noise modelling [3][29][45][46] where the noise is estimated with the original WZ frame provided; online correlation noise modelling estimates the noise using reference frames. Since offline modelling either requires the encoder to perform the complex motion estimation task, or requires the original frame to be available at the decoder which is unrealistic. Therefore, the following reviews of the recent literatures are only dedicated to the online correlation modelling.

The authors of [47] proposed an algorithm to online estimate the noise at frame level for pixel domain DVC codec. It used a weighted mean square error between motion compensated backward and forward reference frames to approximate the variance between SI and WZ frame. And the noise distribution is assumed to be Laplacian distribution. The parameter for the probability dense function is computed from the estimated variance. Simulation results show that there is only a very slight RD performance loss regarding to the off-line approach. Later in the same year, they extended the algorithm to model the Laplacian parameter at different granularity for both offline and online models [48]. Three granularity levels, i.e. frame level, block

level and pixel level are investigated in the modelling algorithm. Experimental results show that the model can achieve better performance at finer granularity and the performance gap between offline models and online models has been reduced. A more comprehensive study of the correlation noise modelling for both pixel domain and transform domain DVC codecs is presented in [4]. This method has been widely adopted in literatures and the best performing DVC codecs.

More recently, in 2009 [49] investigated the online CNM techniques and found that quantization noise also has an impact on the accuracy of noise distribution. Therefore, they estimated the quantization noise for intra frames at the encoder and sent this information to the decoder. The experimental results show significant bit rate reduction for coarse quantization. [50] proposed to use a category map based on previously decoded DCT bands. The map divides transformed coefficients of the current band into two categories, where different parameter estimators are applied to locally compute the Laplacian parameters. Finally, each transformed coefficient is assigned a Laplacian parameter based on its corresponding category and reliability. Compared with the coefficient level noise model in [4], the proposed noise model can only improve the RD performance for high bit-rates up to 0.5 dB. Since the cross-band correlation and the successfully decoded information can significantly influence the reliability of block classification and the accuracy of noise parameter estimation of subsequent bands, later in 2011, they proposed another algorithm [51] to adaptively estimate the Laplacian parameter by using clustering method to exploit correlation across all frequency bands. It was also proposed to combine their algorithm with the noise model in [50] to adaptively optimize the soft side information for LDPCA decoding. The proposed model achieved average improvement in PSNR up to 1.24 dB over the DISCOVER codec. In the same year, [52] proposed a progressive refinement approach for CNM

which used the previously decoded bit-planes and quantization errors to refine estimated correlation noise. Although the proposed CNM refinement consistently performs better than DISCOVER codec, the maximum PSNR gains are only about 0.2dB. [53] estimated the Laplacian parameters for each group in each band, where the groups are derived from classification on the residual energy. The calculation of the Laplacian parameters for each group still follows the method proposed in [4]. However, the Laplacian parameter assigned to each group is derived from a look-up table which is obtained offline. This approach can slightly reduce the CNM complexity compared to the coefficient level model, but the offline lookup table may not well suitable for real-time applications and may not adapt to the various video content.

In 2012, [54] proposed to refine the residual frame by exploiting the correlation of neighbouring coefficients. Residuals of already decoded frames are used to influence the noise distribution of the current frame and thus further exploit the temporal correlation. It then grouped the coefficients in each band into clusters and generated candidate noise parameters for each cluster. Adaptive optimization of the noise parameters are achieved by multiple convergence tests in LDPCA decoding process. Actually, this approach can be seen as a multiple side information approach and therefore it may introduce significant decoding complexity when the number of candidate noise parameters increase, although a good overall RD performance gain can be achieved with regard to the DISCOVER codec.

2.5.5 Consistent Quality Control

The rate control discussed here refers to the decoded frame quality control in distributed video coding. A smooth decoding quality over time is usually desired. Recent advances on this topic are studied below.

Different solutions have been proposed to solve the problem. In [55], a hybrid coding framework using zero vector motion compensation was proposed. The residual of adjacent frames are intra coded and their low frequency coefficients are sent to the decoder. The SI is then generated by taking into account the previously decoded frame and the received residual coefficients. The decoded image quality can be controlled by the quantization step size of the residual frame and the amount of transmitted coefficients according to the quality requirements. The percentage of the to-be-sent low-frequency coefficients is proportional to the SAD of two adjacent frames at the encoder side. However, more efficient residual coder is needed to reduce the encoder complexity as well as the bit rate cost at residual frames. In [56], a distortion model pixel-domain Wyner-Ziv video codecs using the distribution of correlation noise was proposed. The model sees the coding distortion as a function of the quantization step size and the correlation noise parameter. Thus, once the noise parameter is estimated, the encoder can choose the quantization step size that minimizes the difference between the estimated distortion of WZ frames and the target distortion. However, this model requires the estimation of the correlation noise parameters at the encoder side which can further increase the encoding complexity. Furthermore, the distortion model used for key frame coding is very inefficient since the selection of key frame quantization parameters are based on iterative trials of encoding and decoding at encoder side, which again can significantly increase the encoder complexity. More recently, authors in [57] proposed another quality control mechanism by establishing distortion-quantization (DQ) models for both key frames and WZ frames. The correlation noise between the SI and WZ frames is modelled by recreating the rough side information (SI) for each WZ frame at the encoder. With the calculated distribution of the correlation noise, the distortion of WZ frames is online estimated. The quantization parameter which gives

distortion best matches the target distortion is selected by an exhaustive search performing at frequency band level. The algorithm provides a rather smooth image quality over time by an increase of 10% encoder complexity for coarse quantizer.

In [58], Hong Bin et al. proposed to use greedy search algorithm using estimated RD curve to control the quality of WZ frames. Given a target distortion, SI is first estimated at the encoder side to form a range of RD points by using a RD estimator. The resulting RD points are then connected to generate a RD curve which can be used for the greedy search algorithm. It decreased the quantization level from the largest level and measured the distortion using the curve values until it found the best one. The results show smaller quality variance over the decoding time and better overall RD performance than [57]. However, this method lacks quality control for key frames and the complexity of the algorithm has not been measured.

2.5.6 DVC Implementation

Over the past few years, multi-core processors have been widely used across many application domains including general-purpose, embedded, network, digital signal processing, and graphics. The improvement in performance gained by the use of a multi-core processor depends very much on the software algorithms used and their implementation. In the particular case of DVC, parallel implementation could help to reduce the huge complexity of the decoder. The development of efficient DVC implementation is discussed in this section.

Since DVC is a relatively new video coding paradigm, its practical implementation only comes up in recent years. In 2010, [59] proposed a parallel DVC implementation using General Purpose Graphics Processing Unit (GPGPU) [60] and since LDPCA decoding contributes the primary computational complexity, the LDPCA decoding

algorithm was implemented to run in parallel. In 2011, [61] split WZ frames into spatial partitions and each partition is then assigned a processing core such that the decoding can be run in parallel. More recently, a parallel message-passing decoding algorithm [62] for computing LDPCA syndromes is applied through the Compute Unified Device Architecture (CUDA) [63] based on GPGPU. It divides the message passing algorithm into horizontal processing and vertical processing parts, which corresponds to the calculations for the variable nodes messages and the check nodes messages. The calculation of the messages can therefore be parallelized. Furthermore, they also proposed a rate control algorithm to reduce the number of requests in the decoding process. It assigns small step sizes for bit-planes with smaller number of requests while large step sizes for bit-planes with larger number of requests. This algorithm can significantly reduce the decoding complexity but still maintain the adaptivity to the virtual noise, especially when the noise in the SI is high which can lead to numerous parity bits requests. In [64], the authors proposed a parallel implementation for DVC encoder. It divides each frame into multiple tiles. Since there are no computational dependencies among tiles, they can be encoded in parallel using OpenMP [65]. Bit-plane packing technique is also applied to the LDPCA encoding for each tile to speed up the encoding process.

A more comprehensive implementation of DVC decoder in different parallel levels is presented in [66]. This work investigated four parallel Wyner-Ziv decoding algorithms, i.e. parallelism in decoding each bit-plane, parallelism in decoding each spatial partition to avoid dependences between bit-planes, parallelism in decoding each GOP and parallelism in both GOP level and frame partition level. As expected, the last approach achieved the most reduction in decoding complexity.

2.6 Conclusion

This chapter reviews relevant advances on distributed video coding, starting with the theoretical background and possible applications. We then introduce the early architectures of DVC. We have also identified the current research challenges. The current performance status of the state-of-the-art DVC codecs in terms of RD performance and complexity is reviewed. Details of research progress on five areas, namely side information generation, side information refinement, correlation noise modelling, quality control and efficient DVC implementation are described. These areas have fundamental impacts on practical DVC performance and our contributions in Chapters 3 to 6 of this thesis build upon the techniques of this literature.

Chapter 3
Exploration and Exploitation of Reference Frames

3.1 Introduction

Side information generation is an essential function in the DVC decoder, and plays a key-role in determining the coding performance. Frame interpolation is one of the most popular methods used for SI generation, since it takes advantage of both forward and backward reference frames, especially when motion is considered [30][39][67]. The reference frames refer to previously decoded key frames or WZ frames. It is widely believed that interpolated frames give better performance compared with reference frames. However, we found that using reference frames without interpolation performs very close to, or sometimes even better than complicated frame interpolation methods.

Motion learning is a typical approach used to exploit the correlation between SI and Wyner-Ziv frames. In 2008, Varodayan et al. [35] proposed an unsupervised motion learning mechanism to model the forward statistical motion fields at the decoder. It employs an Expectation Maximization method [68] to progressively update the motion. Later, Martins et al. in [37] also proposed a motion learning method that makes use of the previously decoded Discrete Cosine Transform bands to reduce the total bit-rate. However, the above methods suffer from high decoding complexity and do not efficiently exploit the motion information readily available in reference frames.

The main novelty and contributions of this chapter include the following,

We investigate the impact of reference frames on the RD performance and find that the common belief that taking interpolated frames as SI is better than reference frames is not always true.

Based on the above investigation, a new motion learning algorithm exploiting reference frames directly is proposed, leading to significant decoding complexity reduction without incurring any penalty in coding efficiency.

This chapter is organized as follows: Section 3.2 describes the architecture of the proposed DVC codec. The correlation model used in this chapter is explained in detail in Section 3.3. Section 3.4 explores the information provided by reference frames and analyses the advantages of taking reference frame as SI with no interpolation or extrapolation. Next, in Section 3.5, we propose a novel motion learning algorithm and simulation results are shown in Section 3.6. Finally, Section 3.7 concludes this chapter.

3.2 Novel DVC Codec Architecture

The transform domain DVC encoder and decoder proposed in this chapter are illustrated in Figure 3.1 and Figure 3.2, respectively. The system is based on the Stanford architecture [11], which is briefly described as follows.

The input video sequence is split into key frames and WZ frames. The key frames are encoded by a conventional video coding solution, such as H.264 Intra codec. The WZ frames are divided into 4-by-4 blocks. DCT is applied over each block and the resulting coefficients are uniformly quantized (Q) to ensure a low complexity encoder. Quantized coefficients are then converted into a bit stream and encoded by Low Density Parity Check Accumulated codes [22]. The resulting parity bits are stored in the buffer for decoder requests.

At the decoder side, previously decoded frames serve as reference frames. The correlation noise between SI and WZ frames is estimated by the online correlation noise model followed by soft SI generation, as shown in Figure 3.2, and is progressively refined during the iterative EM process, until the stopping criteria is met or the maximum number of iterations is reached. It can be noted that there is also a quantizer at decoder side, which means the soft SI is calculated based on quantized DCT coefficients. This will lead to certain loss in coding efficiency but the decoding complexity can be reduced significantly. Therefore, the use of decoder side quantizer in practical DVC codec design should depend on the restrictions on hardware computing power and also the target RD performance. It is a trade-off between decoding complexity and coding efficiency and this is important as the idea is based on DVC with quantizer installed at decoder side. However, for most DVC codecs in the literature, in order to avoid any performance loss non-quantized DCT coefficients are used to compute the bit probabilities. Finally, the decoded symbols are optimally reconstructed [23]. More details will be discussed in section 3.5.

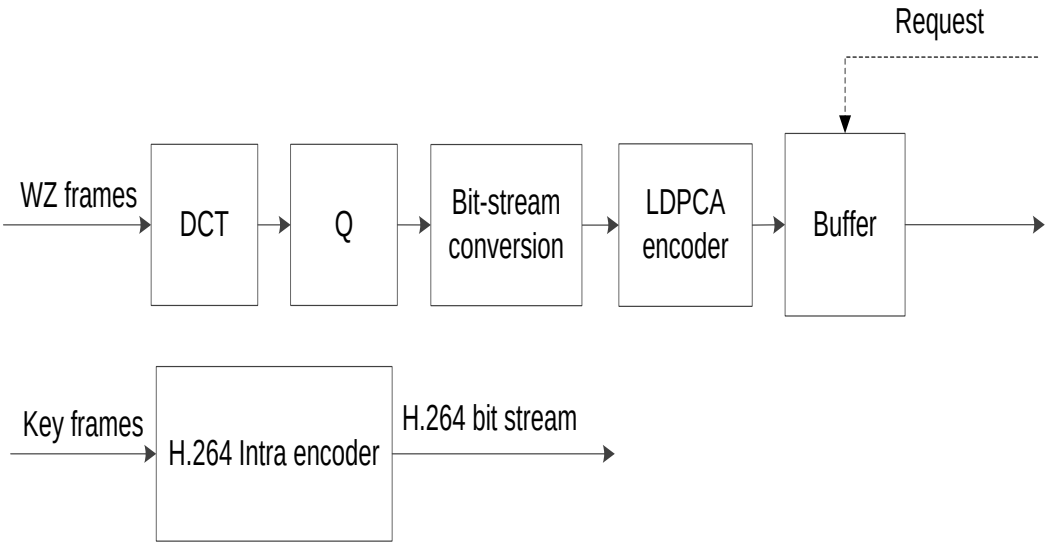


Figure 3.1 DVC encoder architecture

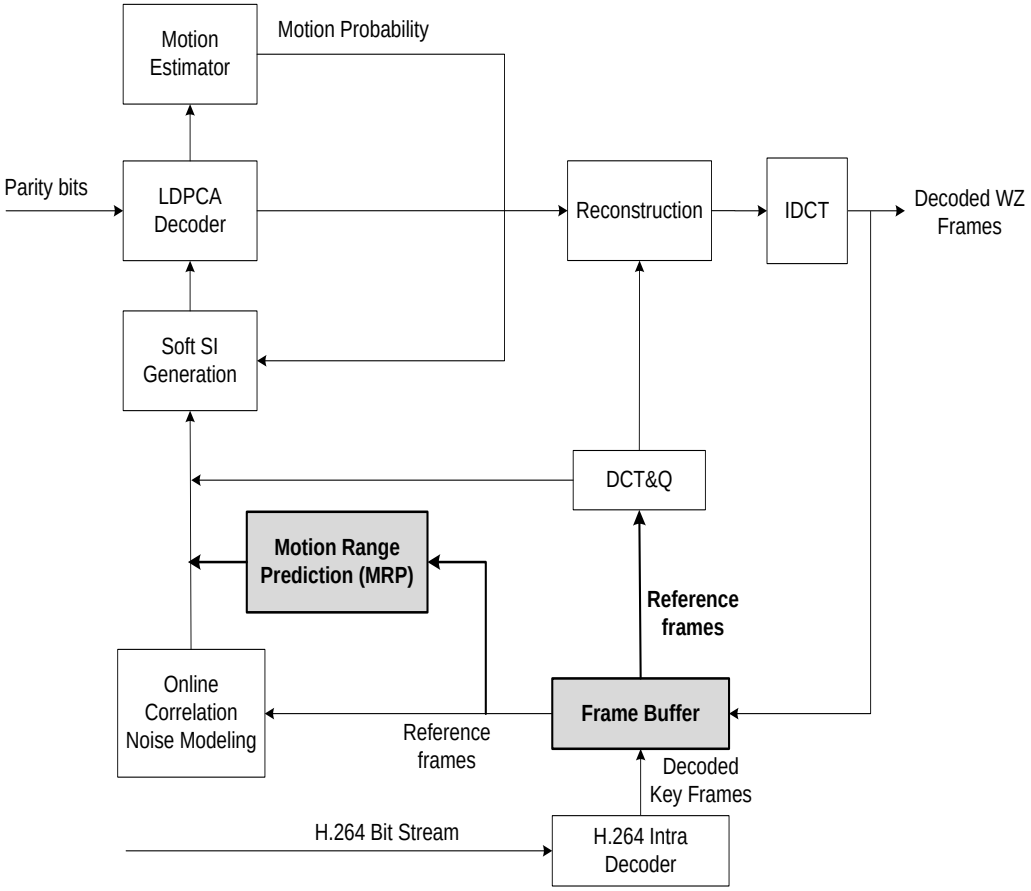


Figure 3.2 Proposed DVC decoder architecture

In the proposed decoder architecture, instead of frame interpolation, reference frames are directly used as SI. Furthermore, the motion fields between the reference frames are also exploited in the Motion Range Prediction module to estimate the position and the size of the searching window of motion fields. This results in a smaller but more precise search region and thus high coding efficiency can be achieved with significantly reduced computational complexity.

3.3 Correlation Noise Modelling

The distribution of correlation noise between WZ frame and SI frame is assumed to be Laplacian distribution. The probability density function of Laplacian distribution for random variable x is $f(x|\alpha, b) = \frac{\alpha}{2} e^{-\alpha|x-b|}$, where b is a location parameter and $\alpha \geq 0$

is usually called as Laplacian parameter or the scale parameter which scales the distribution up and down. The smaller α is, the wider the tail of the curve is. In distributed video coding, b is usually set to 0 so that the distribution is symmetrically centred at 0. Sometimes, in order to restrict the range of probabilities to [0,1] without going through a normalization process, a form of $f(x|\alpha) = e^{-\alpha|x|}$ can be used. After all, the Laplacian distribution is merely an analogous form of the actual distribution. A graph of this function with α=3.5 is shown in Figure 3.3. It can be seen that variables close to the centre of the curve have much higher probabilities than other region. The residual of WZ frame and SI frame follows this distribution since they are very similar to each other and therefore most of the values of the residual frame are on the brink of zeros.

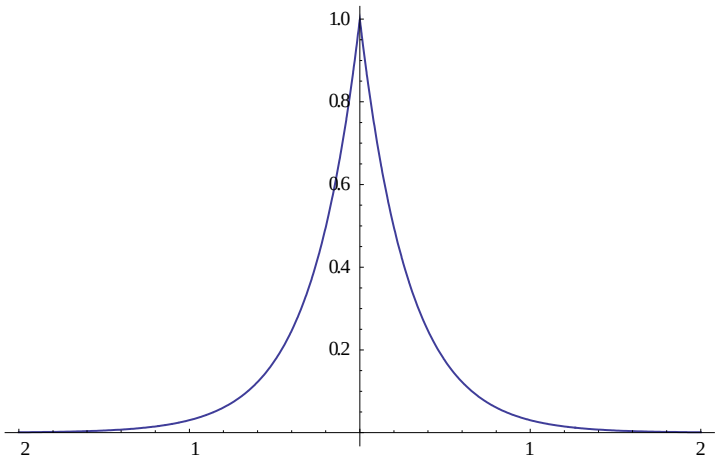


Figure 3.3 Laplacian distribution $\alpha = 3.5$

3.4 Exploration of Reference Frames

Side information is seen as a "noisy" version of the WZ frame. It is obvious that the fewer "errors" SI carries, the fewer parity bits are needed, and thus the better the overall coding efficiency. Frame interpolation based methods assume the motion fields between adjacent frames to be smooth and WZ frames are seen as a transition of the reference

frames. It is expected that interpolated frames contain fewer errors than reference frames when collocated pixels are compared to WZ frames, since interpolation exploits more information than using direct reference frames, especially when motion aided interpolation is applied.

However, if motion estimation and compensation are applied in reference frames, the quality of reference frames is similar, if not better than interpolated frames. For this reason, we infer that choosing reference frames as SI will not have significant loss in RD performance when compared with interpolated frames.

The above assumption is verified by the following simulation and the experimental results presented in Section 3.6. Recall the decoding process that SI is shifted to all the possible positions and compared with WZ frames in transform domain. The probability of each motion field yield by its corresponding shift operation will then be calculated during the EM process. We simulate the decoder motion estimation process by an analogous experiment.

Three different SI are compared in the simulation, namely motion compensated interpolated frames (MCI), backward (B), and forward (F) reference frames. The simulation assumes that DVC codec works in transform domain and the decoder is able to find the best match for each quantized coefficient by searching its surrounding samples within a predefined window. A window size of ±5 is enough to provide good searching results. The three SI are divided into 4-by-4 blocks and then motion compensated. The resulting frames are DCT transformed and uniformly quantized, which corresponds to the soft comparison process between SI and WZ frames at the decoder. The 4-by-4 quantization step matrix depicted in Table 3.1 is divided by the scale factors $S_i \in \{$ 64.0, 32.0, 16.0, 8.0, 4.0, 2.0, 1.0, 0.5 $\}$, according to the

quantization index Q_i with $i \in [1,8]$. Table 3.1 is derived from the intra-coding initializing quantization matrix of H.264 reference software [69]. Each scale factor results in different quantization levels, which will be used to evaluate the coding distortion under different bit rate. The first quantization index Q_1 therefore represents the lowest bit-rate and the last index Q_8 represents the highest bit-rate. The key frames are coded with constant quantization parameters as defined in Table 3.2, obtained from extensive experiments aiming to achieve the best RD performance. These settings are different than the one used in DISCOVER codec since the coding algorithms are different and we only focus on the DVC codecs with a decoder quantizer.

Table 3.1 Quantization step matrix

7	16	22	24
6	22	24	28
18	22	27	33
22	24	32	47

Table 3.2 Key frames quantization parameters

	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8
Hall Monitor	47	45	43	40	37	33	29	24
Foreman	45	44	44	43	37	33	29	24
Coastguard	45	44	42	40	37	33	29	25

Table 3.3 Quality comparison of motion compensated reference frames and interpolated frames for Hall Monitor

Average PSNR	Hall Monitor		
	B	F	MCI
Q1	77.33	77.37	77.80
Q2	71.55	71.56	71.64
Q3	69.34	69.34	69.58
Q4	66.16	66.17	66.49
Q5	63.19	63.22	63.65
Q6	60.41	60.48	60.88
Q7	56.91	56.99	57.35
Q8	53.17	53.30	53.65

Table 3.4 Quality comparison of motion compensated reference frames and interpolated frames for Foreman

Average PSNR	Foreman		
	B	F	MCI
Q_1	76.81	76.72	76.33
Q_2	71.80	71.90	71.71
Q_3	68.99	69.05	68.85
Q_4	64.38	64.43	64.18
Q_5	61.04	61.09	60.66
Q_6	56.91	56.97	56.23
Q_7	52.23	52.28	51.31
Q_8	47.80	47.87	46.71

Table 3.5 Quality comparison of motion compensated reference frames and interpolated frames for Coastguard

Average PSNR	Coastguard		
	B	F	MCI
Q_1	72.43	72.47	72.50
Q_2	73.64	73.58	73.67
Q_3	68.86	68.84	68.73
Q_4	65.15	65.11	64.92
Q_5	61.60	61.50	60.92
Q_6	57.76	57.57	56.44
Q_7	53.36	53.04	51.37
Q_8	48.93	48.54	46.43

We try to find the best match for each block of WZ frame from the shifted blocks in the SI. The performance of each SI is measured by the average peak signal to noise ratio (PSNR) between WZ frames and the motion compensated SI after DCT transform and uniform quantization. Only luminance components of Hall Monitor, Foreman and Coastguard representing videos of different types of motion are used. All the test video sequences are of size QCIF (176×144) at frame rate of 15 Hz. All frames in the test sequences are used, which means 165 frames for Hall Monitor and 150 frames for Foreman and Coastguard. Simulation results presented in Table 3.3-Table 3.5 show that after motion compensation, reference frames have very similar quality to MCI frames. In the case of Foreman sequence, the quality of reference frames is even consistently better than MCI frames. And as of Coastguard sequence, it is observed that reference frames also have better quality in most quantization settings.

3.5 Exploitation of Reference Frames for Motion Range Prediction

The statistical motion fields (SMF) for a 4-by-4 WZ block is given by a probability matrix containing the probabilities of a SI block moving to all the possible positions

using the search window size of $\pm L$. The higher the probability is, the more likely the displaced block is the best match to the WZ block. The decoding algorithm based on the Stanford architecture [11] using the proposed MRP method directly utilizing reference frames as SI is outlined below. It approximates the target motion vectors by an iterative EM refinement process.

A. The statistical motion fields $P\{M_{x,y}\}$ for each 4-by-4 block with top left pixel located at (x,y) are initialized by experimentally chosen distributions which gives good overall performance:

$$P\{M_{x,y}\} = \begin{cases} 0.0004, & if\ x \neq 0\ and\ y \neq 0 \\ 0.016, & if\ only\ x\ or\ y = 0 \\ 0.64, & if\ x = 0\ and\ y = 0 \end{cases} \tag{3.1}$$

B. The Expectation-Step (E-Step) updates the statistical motion fields of each block by the corresponding soft estimate $\mu_{x,y}$ of the block, which before normalization is written as

$$\begin{aligned} P^{(t)}\{M_{x,y}\} &= P^{(t-1)}\{M_{x,y}\} P\left\{SI_{(x,y)+M_{x,y}} \middle| M_{x,y}; \mu_{x,y}^{(t-1)}\right\} \\ &= P^{(t-1)}\{M_{x,y}\} \sum_{l=0}^{2^d-1} \mu_{x,y}^{(t-1)} PMF(l - SI_{(x,y)+M_{x,y}}) \end{aligned} \tag{3.2}$$

where $SI_{(x,y)+M_{x,y}}$ is the SI block located at $(x, y) + M_{x,y}$ and PMF is the probability mass function of the residual between WZ frame and SI in transform domain after quantization. The computation of the sum product in (3.2) for each statistical motion field is over each possible quantization level $l \in \{0, \dots, 2^d - 1\}$, where d denotes bit depth. Therefore, for a single 4-by-4 block, a full search with window size $\pm L$ requires $(4 \times 4) \times (2 \times L + 1)^2$ times computation of (3.2). The computational complexity increases dramatically with the increase of the size of searching window.

Furthermore, this iterative learning algorithm is possible to converge to a coarse motion vectors combination since EM algorithm does not guarantee that the convergence will be to a global maximum. However, motion vectors between forward and backward reference frames can be used to guide the learning process if they are similar to the motion between SI and WZ frame. This requires SI to be the same as a reference frame so that the number of candidate motion vectors for each block can be reduced to the surrounding motion vectors of the guidance motion vector. This results in more precise motion learning and lower computational complexity.

It is proposed here a MRP method that the motion fields between two adjacent reference frames are used as location and size indicator for the searching window, assuming the motion between neighbouring frames is smooth. We search in a small window of size $\pm R$ centered at half of the reference motion vector $M_{x,y}^{Ref}$ as depicted in Figure 3.4, where $W1$ is the proposed searching window and $W2$ is the initial searching window. Experiments show that window size of 2 is sufficient to provide good overall results. The motion fields (M_x, M_y) between SI block and WZ block is confined by the region defined in (3.3), where it guarantees that the search region will not exceed the initial region defined by L.

$$\begin{aligned} M_x &\in \left[max\left(\frac{M_x^{Ref}}{2} - R, -L\right), min\left(\frac{M_x^{Ref}}{2} + R, L\right)\right] \\ M_y &\in \left[max\left(\frac{M_y^{Ref}}{2} - R, -L\right), min\left(\frac{M_y^{Ref}}{2} + R, L\right)\right] \end{aligned} \tag{3.3}$$

Considering the search region defined by (3.3), the complexity reduction in terms of number of motion vector candidates after applying the proposed search region in each EM iteration is,

$$(2 \times L + 1)^2 - (2 \times R + 1)^2$$

For example, for the search window used in [35] with *L=10*, and taking $R = 2$ as defined previously can reduce computational complexity in each EM iteration by 94.3%. This is considerable reduction that will have major impact on implementation aspects of DVC.

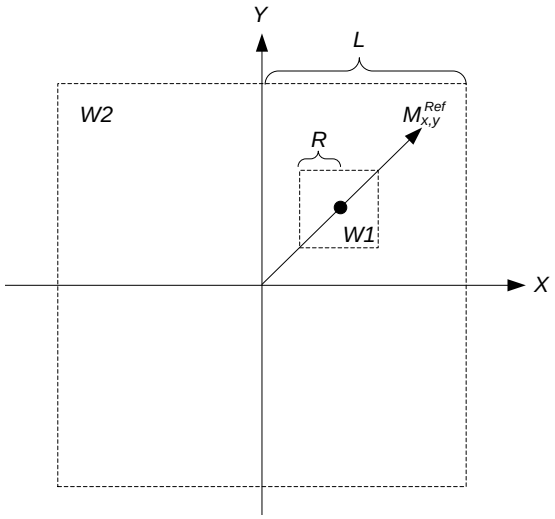


Figure 3.4 Proposed search window for a reference motion vector

C. The Maximization-Step aims to update $\mu_{x,y}$ by generating soft SI followed by an iterative joint pixel LDPCA decoding,

$$\mu_{x,y}^{(t)} = arg \max_{\mu_{x,y}} \sum_{m_{x,y}} P^{(t)}\{M_{x,y} = m_{x,y}\} P\{SI, S | M_{x,y} = m_{x,y}; \mu_{x,y}\} \quad (3.4)$$

where the summation is over each motion field $m_{x,y}$ and S represents syndrome checks.

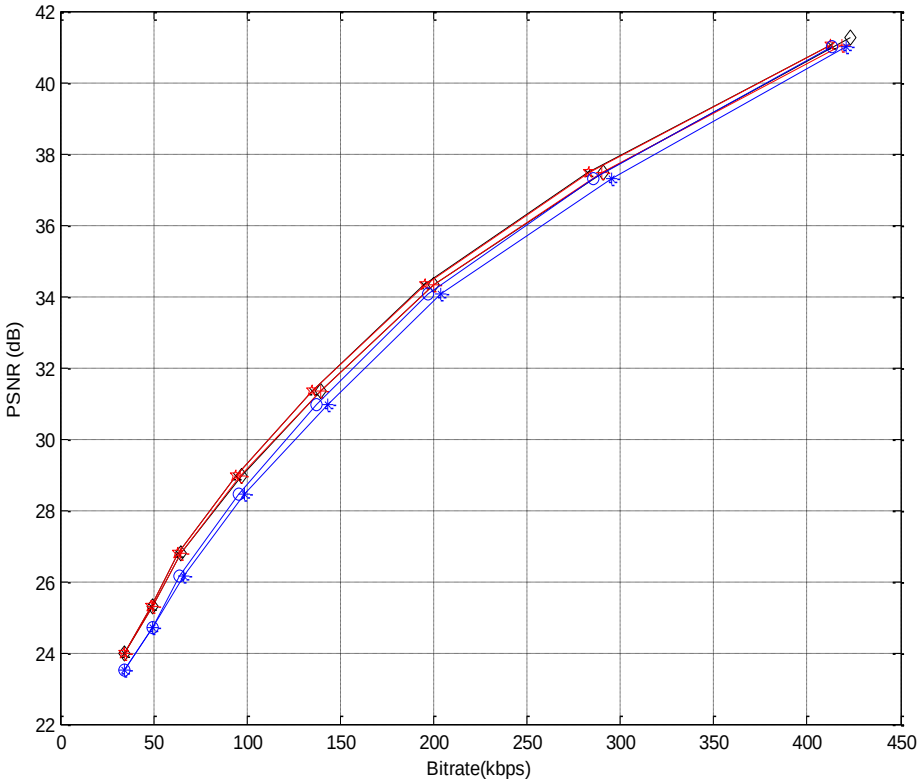
D. The EM algorithm terminates when the syndrome check is satisfied.

3.6 Simulation Results

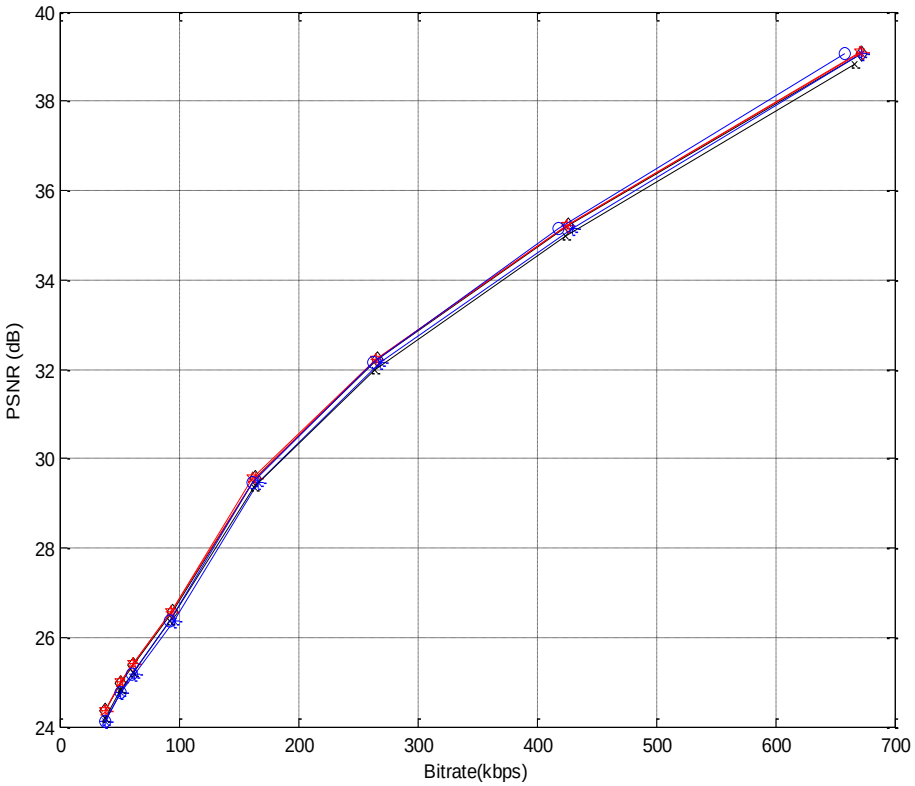
The proposed MRP algorithm is evaluated by our DVC codec presented in Figure 3.1 and Figure 3.2 in Section 3.2. We compare the RD performance of the basic DVC codec with and without the proposed MRP algorithm, using different side information.

The same coding configurations as in section 3.4 are also applied in the following experiments. A constant GOP size of 2 is used for all test sequences, i.e. odd frames are key frames whereas even frames are WZ frames.

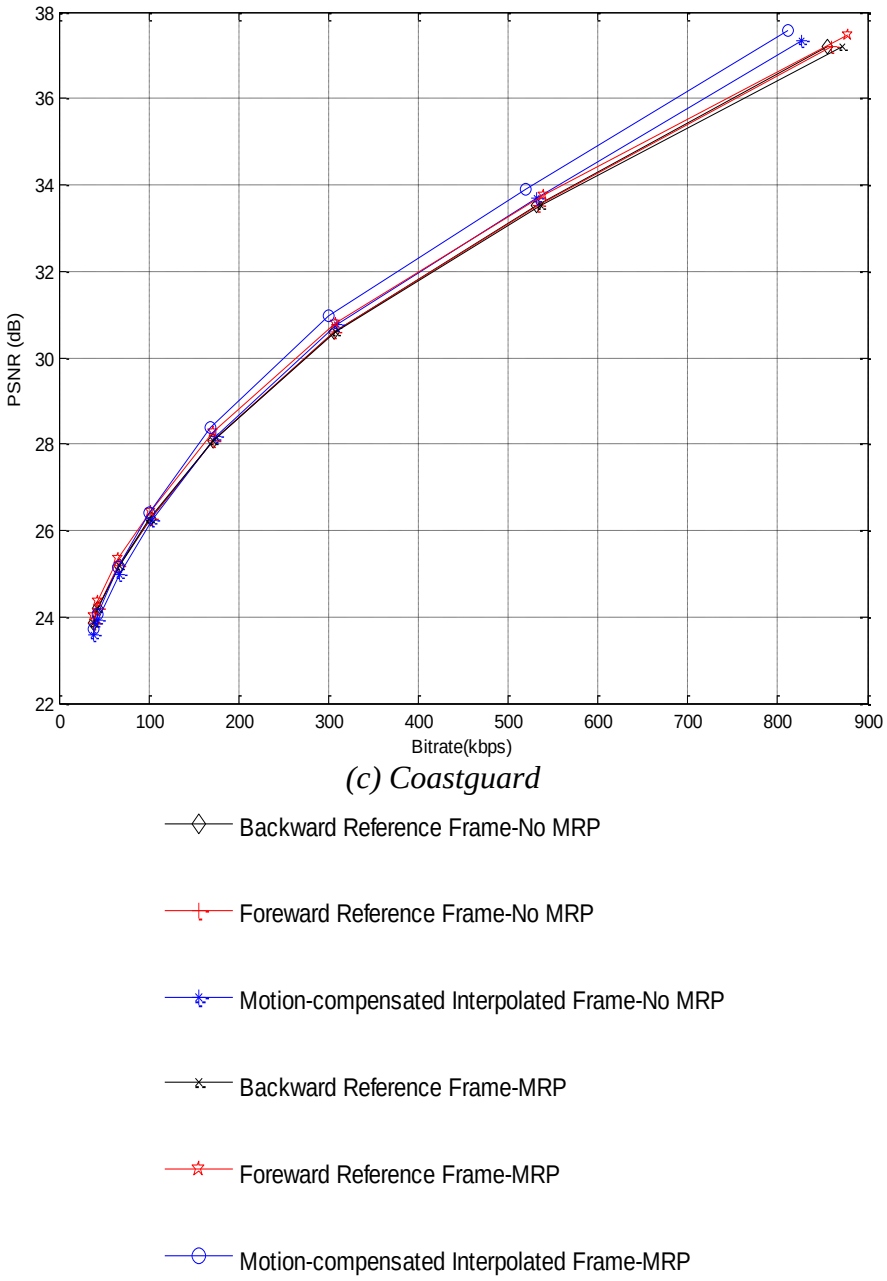
Figure 3.5 shows the RD performance of the decoded frames for all the sequences. It is observed in Hall Monitor sequence that backward and forward reference frames perform consistently better than MCI frames, with and without MRP algorithm. In Foreman and Coastguard sequences, reference frames have better performance in low bit-rate, both with and without MRP algorithm. However, the MRP boosts the performance of MCI frames in high bit-rate, as the increase of bit-rate brings more details in the frames. When the motion or scene change is high, these details show significant increase in bit consumption. When the search region is restricted by MRP algorithm, if EM converges with a coarse motion field, the strength of MCI frames start to show up. The proposed MRP method gives slightly better RD performance in most of the time over all the test sequences.



(a) Hall Monitor



(b) Foreman



(c) Coastguard

Figure 3.5 RD performance for different video sequences

The above results prove that reference frames can be a good SI candidate that gives very similar or sometimes better RD performance when compared with interpolated frames. Therefore, it is fair to say reference frames offer potential for a better SI candidate.

3.7 Conclusion

This chapter reveals for the first time that using reference frames as SI is capable to achieve similar or sometimes even better coding efficiency than the widely used MCI frames when the bit probabilities are computed in quantization domain. In order to maximize their potential, we also presented a new technique to exploit the motion information between reference frames. Simulation results show that the proposed MRP method can significantly reduce complexity in each Expectation-Maximization iteration with no loss in RD performance. This work brings new insight and strength to the use of reference frames. It opens attractive perspectives that allow us to better understand the role of reference frames in distributed video coding.

Chapter 4
Pixel Granularity Side Information Synthesis Framework and Parallel Implementation

4.1 Introduction

For transform domain DVC, the transformed coefficients are usually grouped into bands and the Wyner-Ziv frames are decoded band by band. Each decoded band provides partial information of the WZ frame, which is not available previously and thus, can be utilized to improve the SI. Currently, this refinement process is mainly carried out using block level motion search algorithms in the state-of-the-art literature due to complexity issues. For high motion video content and long group of picture sizes, this can bring significant block artifacts to the decoded frames. Furthermore, each time when SI is improved, the correlation noise between SI and WZ frame changes accordingly. Therefore, the initially estimated noise distribution may not be accurate anymore and thus require the correlation noise model to adapt itself to the changing noise.

Since iterative algorithms are widely used for DVC, the decoder is naturally slow in computation. Introduction of SI refinement and correlation noise re-modeling will therefore add more computational complexity to the decoder.

To tackle the SI refinement problem which is a major challenge in DVC advancement, we propose a flexible pixel granularity side information synthesis (PGSIS) framework and investigate its performance compared with block based classical systems. To provide in depth study of DVC decoding complexity and future improvements we

have fully implemented a state-of-the art DVC decoder using both conventional serial approach and parallel processing technology. The main contributions of this chapter are summarized below.

1) A finer granularity side information synthesis (SIS) framework is proposed. It works efficiently at pixel level and provides superior synthesized SI in both subjective and objective image quality. The proposed architecture is flexible and modular based which can be integrated into most modern DVC architectures.
2) To further save the required parity bits and hence improve the rate-distortion performance, we propose an adaptive virtual noise model alongside the SIS algorithm. It learns the new noise distribution during the SI refinement and gives more accurate knowledge of the correlation of the WZ frames and SI.
3) Full implementation of serial and parallel DVC decoders with block based and PGSIS SI refinement techniques. A highly parallelized software implementation is recommended to speed up the decoding time and bring DVC one step closer to practical use. We have also identified potential areas for further complexity reduction to be made proportional to the number of CPU employed for faster practical systems applications. Since our implementation is platform independent, it is scalable for any multicore hardware architecture.

The rest of the chapter is organized as follows. Section 4.2 describes the transform domain DVC architecture with the proposed framework. It also provides details of initial SI generation and virtual channel modelling techniques without the proposed framework, which will be used for performance comparison in Section 4.5. Section 4.3 introduces the novel SIS framework. The parallel implementation is described in section 4.4. Section 4.5 is dedicated to the experimental results, performance evaluation and analysis of the proposed framework. Finally, Section 4.5.3 concludes this chapter.

4.2 System Architecture

The framework proposed in this chapter is based on the popular Stanford architecture [11] and will be described in detail next.

The input video is divided into key frames and WZ frames, as shown in Figure 4.1. Key frames are inserted periodically determined by GOP size and encoded by conventional intra codec, such as H.264/AVC Intra codec [1]. The WZ frames are divided into 4-by-4 blocks. In each block, DCT and a uniform quantization are performed. The quantized DCT coefficients are grouped into frequency bands and converted into bit-planes. Each bit-plane is separately encoded using low-density-parity-check accumulated (LDPCA) codes [22] and stored in a buffer for the decoder requests. An 8-bit cyclic redundancy check (CRC) code is also generated for each bit-plane to confirm decoding is successful.

At the decoder side, two reference frames obtained from the decoded key frames and WZ frames are interpolated using optical flows to generate the initial SI. The intermediate motion compensated version of the two reference frames are then DCT transformed and their residue is used for virtual channel modelling.

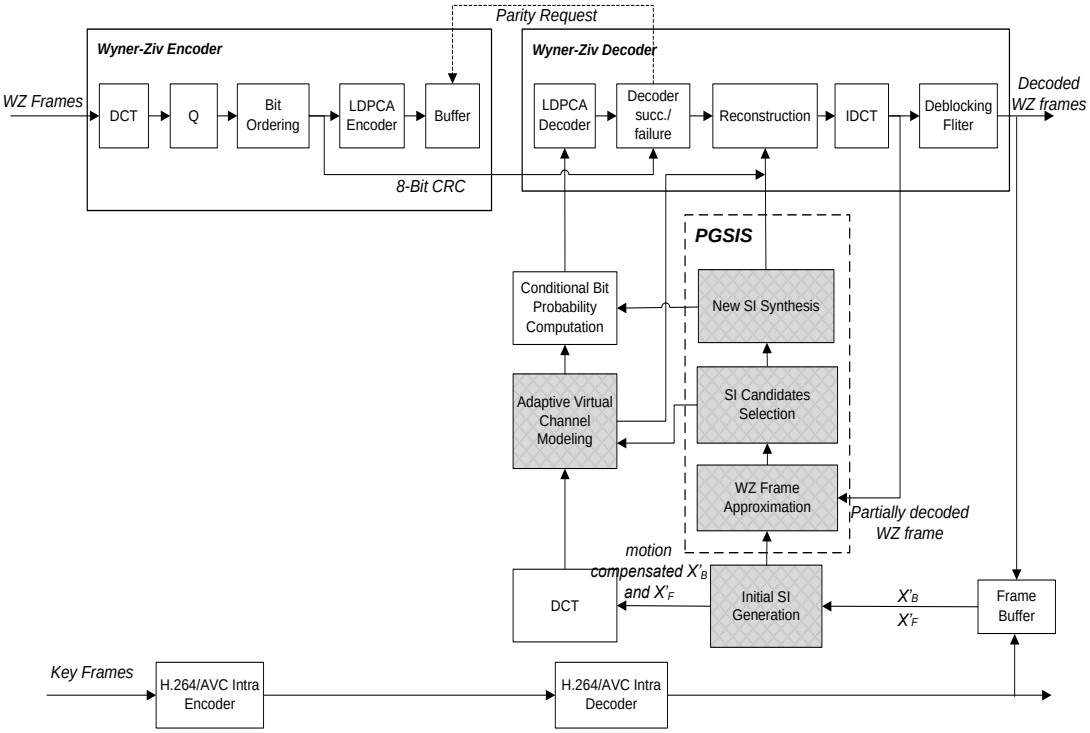


Figure 4.1 Proposed PGSIS-DVC system architecture

The estimated correlation between SI and WZ frame is exploited to compute the conditional bit probability. Using this bit probability, the LDPCA decoder performs an iterative message propagation algorithm (MPA) to decode each bit-plane, starting from an estimated code rate and will request more parity bits from encoder if the available parity bits are not sufficient. The decoding procedure follows the zigzag scan order. After successfully decoding all the bit-planes for each DCT band, these bit-planes are grouped together to form the quantized symbols and optimally reconstructed [23]. A de-blocking filter is then applied to the final decoded frames to give better image quality.

The proposed PGSIS framework and its associated components are highlighted in Figure 4.1. The initial SI is generated by an optical flow algorithm and it will be updated in PGSIS framework each time a DCT band being successfully decoded. PGSIS framework consists of three key components. WZ frame approximation gives a rough estimate of the actual WZ frame according to the partially decoded information. The approximated WZ frame provides a reference for the next component to select the

right SI candidates. Each selected SI candidate is then assigned a weight factor to mark its importance with regards to its quality. Finally the new SI can be synthesized considering all the selected candidates and their weight factors. The produced weight factors are exploited again in the virtual channel modelling to adapt the updated correlation noise. More details of the proposed framework will be explained in section 4.3.

The initial SI generation and the virtual noise modelling for our transform domain DVC codec without using the PGSIS framework are briefly explained below.

4.2.1 Initial SI Generation

1) Bi-directional motion estimation

Optical flows are used to determine the motion between two neighbouring reference frames. To generate precise motion vectors, we used a highly improved Horn–Schunck method optical flow estimation [70][71][72]. The algorithm is based on a coarse-to-fine warping strategy using a variational model to minimize a rotationally invariant energy function for optical flow computations based on two terms: a robust data term with brightness constancy and a gradient constancy assumption, combined with a discontinuity preserving spatio-temporal smoothness constraint. The algorithm is robust under considerable amount of noise and allows for large motion displacements, which is favourable for high motion video content and long GOP sizes. The bi-directional motion estimation is performed between the two adjacent reference frames. Optical flows are extracted from one of them by taking the other one as a reference.

2) Motion compensated frame interpolation

The motion vectors (dx, dy) obtained from previous step can be used for frame interpolation. We employ a straightforward interpolation scheme that the initially

interpolated SI pixel $Y^{(0)}(x,y)$ at position (x,y) is derived from the mean of the forward motion compensated pixels $X'_F\left(x+\frac{dx_F}{2}, y+\frac{dy_F}{2}\right)$ and backward motion compensated pixels $X'_B\left(x+\frac{dx_B}{2}, y+\frac{dy_B}{2}\right)$ through half of the forward motion vector (dx_F, dy_F) and the backward motion vector(dx_B, dy_B), respectively. For any motion vectors that go out of image boundaries, the co-located pixel of corresponding reference frame will be used instead, as depicted in the following formula,

$$Y^{(0)}(x,y) =$$

$$\begin{cases} X'_B(x,y) & if\left(x+\frac{dx_B}{2}\right) \notin [0,W)\ or\ \left(y+\frac{dy_B}{2}\right) \notin [0,H) \\ X'_F(x,y) & if\left(x+\frac{dx_F}{2}\right) \notin [0,W)\ or\ \left(y+\frac{dy_F}{2}\right) \notin [0,H) \\ \frac{X'_B\left(x+\frac{dx_B}{2}, y+\frac{dy_B}{2}\right)+X'_F\left(x+\frac{dx_F}{2}, y+\frac{dy_F}{2}\right)}{2} & else \end{cases} \quad (4.1)$$

where W and H are the width and height of each frame, respectively. The motion compensated version of backward and forward reference frames can be further utilized in virtual channel modeling process.

It should be noted that since the optical flow and frame interpolation techniques are independent modules, it is possible to replace them with more advanced solutions such as the top ranked optical flow in [73] and interpolation method in [74].

4.2.2 Virtual Channel Modelling

The correlation noise between SI and WZ frames are assumed to be Laplacian distributed and estimated by the virtual channel model in the following steps.

1) Residual frame generation. Residual frame provides an estimate of the actual noise between the SI and WZ frame. The motion compensated version of backward and

forward reference frames obtained previously for initial SI generation are used again to compute the residue frame N given by

$$N(x,y) = X'_B\left(x + \frac{dx_B}{2}, y + \frac{dy_B}{2}\right) - X'_F\left(x + \frac{dx_F}{2}, y + \frac{dy_F}{2}\right) \tag{4.2}$$

2) DCT transform on the residual frame. Since the proposed DVC codec works in transform domain, the residual frame R has to be DCT transformed,

$$R = |DCT(N)| \tag{4.3}$$

where $|x|$ is the absolute value of x.

3) The Laplacian parameters are estimated online at coefficient level based on the algorithm proposed in [4].

4.3 Pixel Granularity Side Information Synthesis

4.3.1 Typical Approach

A typical approach for SIS consists of three steps, WZ frame approximation, candidate SI generation and new SIS. The basic idea is to find the best match from candidate SI to the actual WZ frame and use the best match to replace corresponding SI. WZ frame is not available at the decoder side. However, during the band-by-band decoding process, partial knowledge of WZ frame will become gradually available and hence can be utilized to generate an approximation of the original WZ frame. Candidate SI can be chosen from any frame that is similar to the WZ frame. Once the SI candidates are created, the current SI can be updated to a better quality for decoding the subsequent bands. Due to the changes of SI, the initially estimated correlation noise model will also need to be updated to give more accurate noise distribution. The details of these steps are presented below.

4.3.2 WZ Frame Approximation

In order to measure the quality of the candidate SI, some knowledge of the WZ frame is required at the decoder side. During the decoding process, some bands of the WZ frame will gradually become available. We can reconstruct the currently decoded bands and apply inverse DCT transform to generate the so called partially decoded WZ frame. This frame can be seen as an approximation of the genuine WZ frame. The approximation can also be in the transform domain, i.e. without inverse DCT transform after reconstruction. However, most DVC frameworks only allow sequential band-by-band decoding, i.e. there is only a small subset of all the DCT bands available at one time. These limited bands information is not sufficient for block based motion estimation. A technical report from HP lab [42] shows that the number of DCT bands must be sufficient for block based motion estimation to provide acceptable block matching results. The increased complexity is also an obvious drawback. Although the reconstructed symbols are already in transform domain and directly available for further computation, SI will have to be shifted to all possible directions and DCT transformed for motion estimation. Therefore, motion analysis in pixel domain is clearly more desirable than in transform domain.

Extensive experimental results have shown that there are certain areas in a frame cannot be refined by the above approach since the best match to the partially decoded WZ frame may not necessarily be the best match for the actual WZ frame. This also confirms the results in [39]. Including these areas in the SI update not only increases the computational complexity but may also lead to poorer SI quality.

Therefore, it is important to exclude such regions before proceeding to the next step. A reasonable approach is to filter the blocks that are similar to the co-located blocks in

the partially decoded WZ frame since these blocks are believed to be well updated already, no need for further refinement.

We propose here not to refine block n if it has a sum of squared difference (SSD) smaller than the average SSD of all the blocks in SI frame. The SSD of the nth 4×4 block is defined by

$$SSD_n = \sum_{y=0}^{3}\sum_{x=0}^{3}[Y^{(t-1)}(x,y) - \hat{X}^{(t-1)}(x,y)]^2 \tag{4.4}$$

where $Y^{(t-1)}$ and $\hat{X}^{(t-1)}$ are the previously updated SI block and the partially decoded WZ block at time $t-1$, respectively. A block that has been discarded for updating the current SI may be picked up for refinement in the future as long as it contains fewer "errors" than the average "errors" in the SI frame. The average SSD is given by

$$E[SSD] = \frac{1}{L}\sum_{n=0}^{L-1} SSD_n \tag{4.5}$$

where L is the number of blocks in a frame.

4.3.3 Candidate SI Selection

The blocks selected in the previous step have to be updated with the newly synthesized information. This information comes from multiple SI candidates and they need to be carefully selected from a range of frames. Reasonable candidates for SIS can be decoded key frames, initial SI frame, currently updated SI frame and partially decoded WZ frames, but some may provide more information than the others. Both backward and forward decoded reference frames are used for initial SI generation and we further exploit them in the SIS process. The previously updated SI may have some new information derived from the SI candidates and therefore it is also selected to prepare the information for further SIS. As explained in the previous step, the partially

decoded WZ frame is used to measure the quality of SI candidates and the better candidates will contribute more on the final synthesized SI.

In classical DVC, each selected block is typically replaced entirely by the candidate block. This approach ignores the texture context and may produce obvious edge effects. We avoid this common practice and use a finer granularity, pixel level SI candidate selection to address the block artifacts problem introduced by block level synthesis. It is observed in our experiments that these block artifacts can propagate during the SI refinement process, further impairing the objective and subjective SI quality.

We propose to use three SI candidates $Y_c, c = 1,2,3$ for the new SI synthesis, i.e. the pixels from the motion compensated backward reference frame Y_1, forward reference frame Y_2 and the previously updated SI frame Y_3,

$$\begin{aligned} Y_1(x,y) &= X'_B(x + dx_B, y + dy_B) \\ Y_2(x,y) &= X'_F(x + dx_F, y + dy_F) \\ Y_3(x,y) &= Y^{(t-1)}(x + dx_{Y^{(t-1)}}, y + dy_{Y^{(t-1)}}) \end{aligned} \tag{4.6}$$

where $(dx_{Y^{(t-1)}}, dy_{Y^{(t-1)}})$ is the motion vector between the previously updated SI $Y^{(t-1)}$ and the partially decoded WZ frame.

The previously updated SI is considered as a more reliable candidate than backward and forward reference frames since it contains information from both of them. Therefore, it deserves more exploration for corresponding candidate generation. We take two steps to prepare Y_3.

1) Block based bi-directional motion compensation: A 3-by-3 low pass filter is applied on both reference frames to facilitate the subsequent motion estimation. Then forward and backward motion vectors are estimated by block matching algorithm between both reference frames and partially decoded WZ frame using mean absolute difference as the cost function.

These motion vectors are further refined by sub-pixel motion estimation and current SI is updated by bi-directional motion compensation.

2) Pixel level optical flow motion compensation: The SI obtained in the above step provides a good estimate of SI candidate but can be improved by pixel level optical flow motion compensation.

However, for low bit rate scenarios, step 1 above is not recommended as critical bit rate condition can introduce serious block artifacts using block based motion compensation, which further affect the performance of step 2.

4.3.4 New SI Synthesis

In the proposed SIS framework, all the selected candidates are exploited in pixel domain to give more precise motion estimation and lower computational complexity. Three candidate SI frames computed in (4.6) are employed for new SI synthesis. The error of each pixel is measured in square error considering all of its surrounding pixels and the final synthesis is performed by weighted mean of all the selected candidates where smaller weights are assigned to the pixels containing more errors.

Since multiple SI is used for synthesis, it is expected to have the candidate pixel that contains fewer errors bigger weight than the others. However, the partially decoded WZ frame is only an approximation of the actual WZ frame, considering the difference of a single pixel is not appropriate. Therefore, it is proposed to take all the neighbouring pixels into account when calculating the weight. We still use SSD to measure the error ε_c of the cth candidate pixel at position (x_0, y_0), considering all its surrounded pixels,

$$\varepsilon_c = \sum_{y=y_0-1}^{y_0+1} \sum_{x=x_0-1}^{x_0+1} [Y_c(x,y) - \hat{X}(x,y)]^2 \tag{4.7}$$

where the candidate pixel Y_c is taken from (4.6). Since the weight for a candidate pixel should monotonically decrease with its "errors", we simply take the reciprocal of ε_c as the weight,

$$\omega_c = \frac{1}{\varepsilon_c} \tag{4.8}$$

This weight must be always followed by a normalization step as in (4.9) to achieve a normalized distribution of the weights.

$$\overline{\omega_c} = \frac{\omega_c}{\sum \omega_c} \tag{4.9}$$

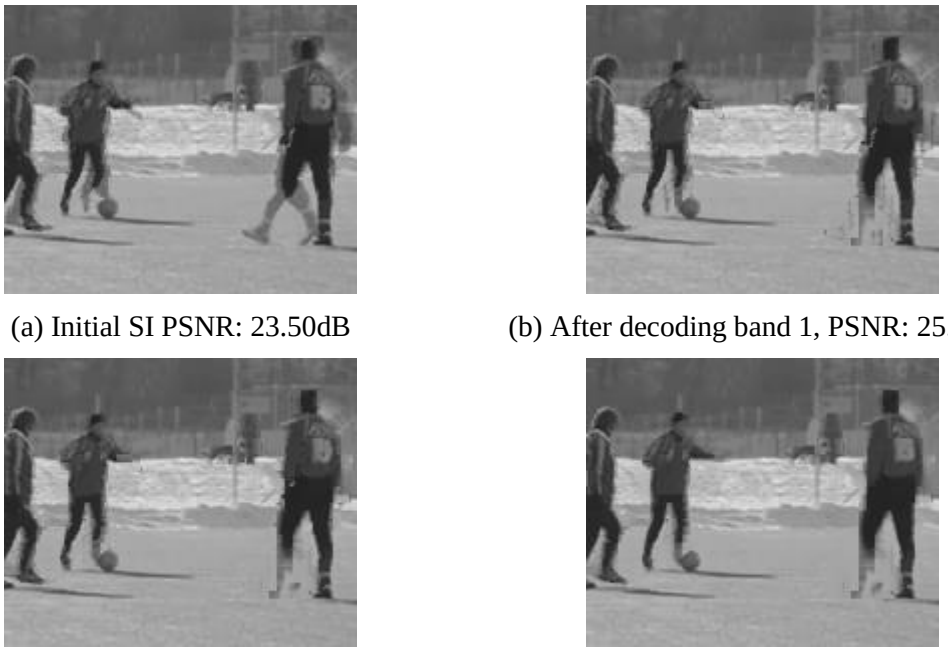
where $\sum \omega_c$ is the sum of the weights of all three SI candidates.

It should also be noticed that for a slow motion video sequence such as Hall Monitor, there may be no difference between some candidate pixels and the partially decoded pixel. Therefore, there is no need to update these pixels and they can be skipped before any synthesis. Finally, the synthesis of the new SI is obtained by the sum of all the weighted candidates,

$$Y^{(t)}(x,y) = \sum_{c=1}^{3} Y_c(x,y) \times \overline{\omega_c} \tag{4.10}$$

The block artifacts can be severe using block based SI refinement solutions, especially for high motion contents. Figure 4.2 shows one experimental result of the updated SI for the Soccer sequence using the 8th quantization matrix Q8 [19][37][39] during the decoding process using a classical block wise SI refinement approach [37]. It can be observed that the block artifacts are propagating throughout the refinement process, degrading the refinement achieved during the decoding process. Figure 4.3 shows the updated SI frames under the same test conditions using the proposed method.

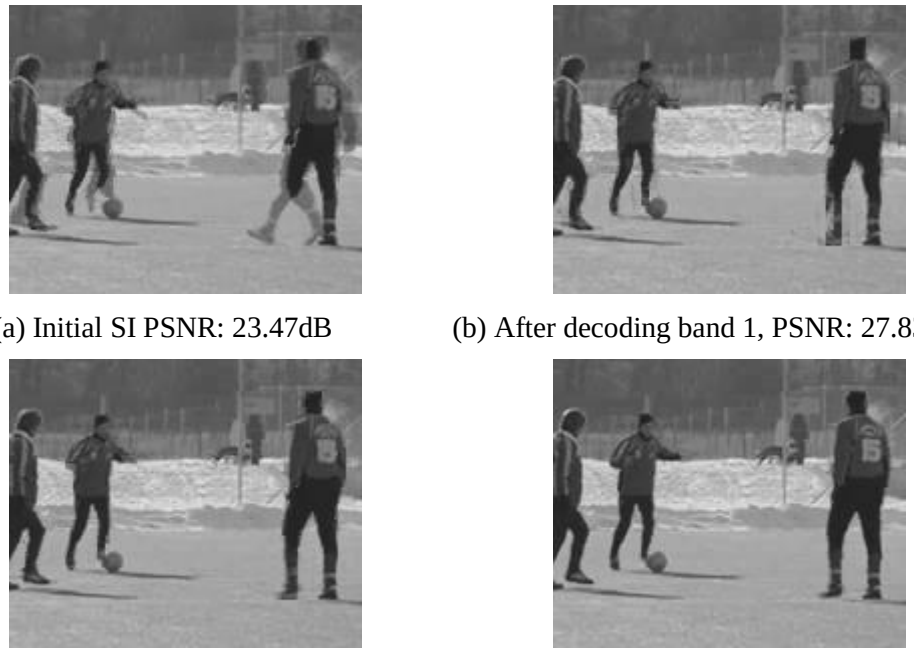
There is no severe block artifact throughout the decoding process and both the subjective and objective SI quality is better than the block based approach.



(a) Initial SI PSNR: 23.50dB (b) After decoding band 1, PSNR: 25.79dB

(c) After decoding band 2, PSNR: 26.83dB (d) After decoding band 13, PSNR: 28.05dB

Figure 4.2 Block granularity SIS for Soccer Q8



(a) Initial SI PSNR: 23.47dB (b) After decoding band 1, PSNR: 27.83dB

(c) After decoding band 2, PSNR: 30.01dB (d) After decoding band 13, PSNR: 34.37dB

Figure 4.3 Pixel granularity SIS for Soccer Q8

Taking a closer look at the block based approach, it can be observed that this approach can hardly represent arbitrary edges and the basic motion search can easily fail for intense motion scenario or fast scene changes. It can be seen from Figure 4.2 (b) that the duplicated leg shading has almost been removed from the person stand on the right after decoding the first band. However, the edges of the leg are not well shaped and the left foot is missing. Although a lot of noise has been removed around the leg, the left foot cannot be recovered even after decoding 13 bands.

The updated SI frames using the proposed pixel granularity approach can be seen in Figure 4.3. It can be observed from Figure 4.3 (b) that after decoding the first band, the new approach has removed much more noise around the left leg from the person on the right and looks better shaped, although still blurry. It can also be noticed that the whole foot has been recovered after band 2 is decoded. It can also be seen that after decoding band 13 (Figure 4.3 (d)), most of the noise has been removed from the scene and the PSNR increased significantly by 10.9 dB compared with the initial SI PSNR (Figure 4.3 (a)). However, in the block based approach there is only 4.55 dB gain.

4.3.5 Adaptive Virtual Channel Modelling

SI is updated during the decoding process, which suggests the correlation noise is also changing over time. Therefore, a virtual channel model that can adapt to this change is able to fit the real noise distribution better. Recall the weight given in (4.9) measures the importance of a certain SI candidate and we use the sum of the weighted means to synthesize the new SI. The larger the weight, the bigger proportion of the corresponding candidate is used for SIS. It is proposed to adaptively update the virtual channel model taking into account the new information obtained from the refined SI. This gives the decoder a better knowledge of the virtual noise distribution.

Residue Frame Re-estimate: The newly synthesized SI can bring an essential change on the SI content. Affected by this change, the initially estimated Laplacian parameters using the motion compensated reference frames may not be accurate any more. The re-estimate of the residue frame is crucial for the final RD performance, especially for high motion content under high bit rate. Since the weighting factor in (4.9) gives the proportion of influence of each SI candidate, we can still use it to update the simple noise frame in (4.2) to a blended noise frame as depicted in (4.11) such that the Laplacian parameters can be improved substantially. The adaptive virtual noise is given by

$$N(x,y) = \sum_{c=1}^{3} [\left| Y_c(x,y) - \hat{X}^{(t)}(x,y) \right| \times \overline{\omega_c}] \tag{4.11}$$

where $|x|$ is the absolute value of x.

In a similar manner, the adaptive probability density function, p is calculated as follows,

$$p(X(x,y) - Y(x,y)) = \sum_{c=1}^{3} (\overline{\omega_c} \frac{\alpha(x,y)}{2} e^{-\alpha(x,y)|X(x,y) - Y_c(x,y)|}) \tag{4.12}$$

where $\alpha(x,y)$ is the Laplacian parameters at position (x,y) computed through correlation noise modeling.

The weighted factor is integrated into the calculation to give higher priority for more possible SI candidate but also allows the other candidates to contribute to the final noise distribution. This sum of weighted distribution changes the initial noise estimation which gives more accurate conditional bit probability for LDPCA decoding and will thus reduce requested parity bits and decoding time significantly. It can be noticed that the synthesized SI also takes into account the priority of the selected SI candidates and

therefore could be used to re-estimate the virtual noise distribution instead of reusing the weighted factors.

4.4 Parallelized Software Implementation

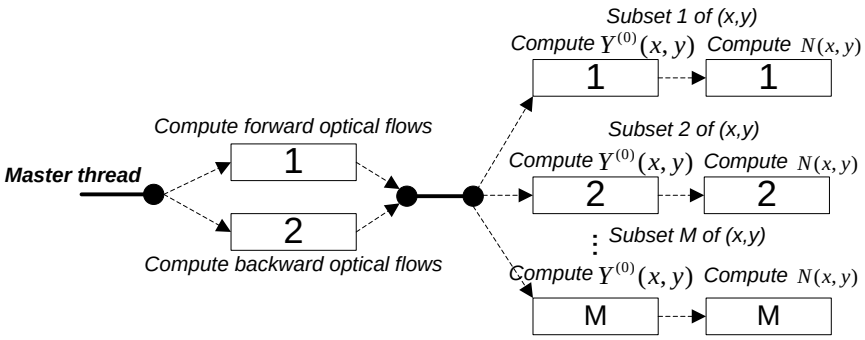
The complexity of PGSIS depends mainly on the chosen optical flow algorithm. For efficient implementation of PGSIS and to advance the speed limit on state-of-the-art DVC codec, we implement PGSIS-DVC with highly efficient parallelized design using Open Multiprocessing API (OpenMP) technology. OpenMP provides a simple and flexible interface for parallel programming and supports multi-platform on most processor architectures and operating systems and is therefore fully portable among different platforms.

The key for efficient implementation highly depends on the organization of the data structures since data in parallel regions have to be fully independent to each other.

4.4.1 Initial SI Creation

We create two parallel regions for initial SI creation, one region for computing the optical flows and the other one for frame interpolation and residual frame generation. The forward and backward optical flows can be computed separately and require two threads/processors. The resulting motion vectors are used for frame interpolation, which is carried out using pixel-by-pixel bi-directional motion compensation. It can be noted that motion compensation for each pixel can be calculated without the knowledge of other pixels in different locations. Therefore, they can be divided into a group of subsets, and each subset can be handled by a separate thread. This procedure can be depicted by the flow chart in Figure 4.4 (a). The master thread creates two threads for computing the optical flows and they join to the master thread after both finish the task, then the master

thread creates M threads (according to the number of processors) in which each subset of pixels are processed separately in parallel.



(a) Flow chart

```
#pragma omp parallel sections {
    #pragma omp section
{ compute forward optical flows (dx_F, dy_F) }
    #pragma omp section
    { compute backward optical flows (dx_B, dy_B) }
}
#pragma omp parallel for
for each pixel (x, y) {
    compute Y^(0)(x, y) using equation (4.1);
    compute N(x, y) using equation (4.2);
}
```

(b) C++ pseudo code using OpenMP

Figure 4.4 Initial SI creation

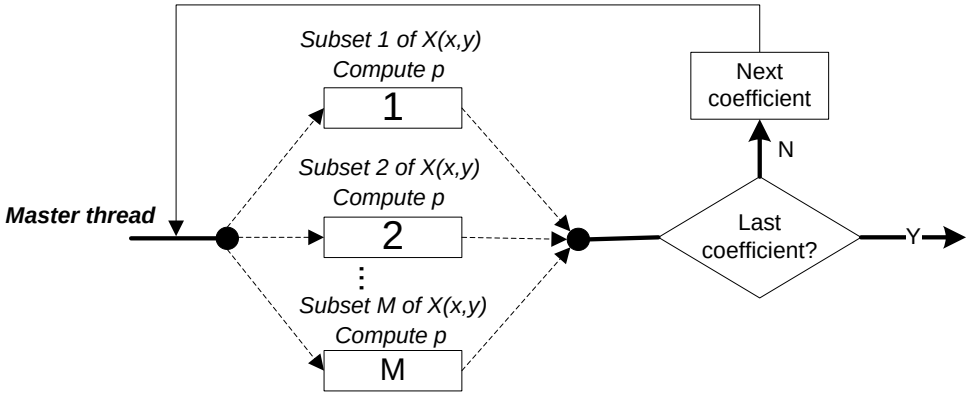
The corresponding C++ pseudo code implementation of this routine using OpenMP is given in Figure 4.4 (b). Two parallel sections to compute forward and backward optical flows are created. Since each parallel section here costs similar time for execution, one section does not need to wait the other for long time of synchronization. Then pixels are divided into subsets such that each subset of the initial SI and residual frame can be computed in parallel.

4.4.2 Adaptive Correlation Noise Modelling

Similarly, it can be noted that the calculation of p in (4.12) is independent to (x, y) which suggests that it can be distributed equally among multiple threads/processors.

The flow chart for adaptive correlation noise modelling is shown in Figure 4.5 (a), where the possible WZ coefficients $X(x, y)$ are divided into M subsets and each of them is run by a thread/processor to form a parallel region.

The C++ pseudo code using OpenMP for this routine is shown in Figure 4.5 (b). All the possible DCT coefficients of WZ frame are divided into subsets. Each subset is handled by a separate thread. The sum of weighted Laplacian distribution p is declared as "private" so that each thread will have its own instance of p to avoid occuring race conditions.



(a) Flow chart

```
for each DCT coefficient i {
    #pragma omp parallel for  private(p)
for each possible X(x,y) {
        for each SI candidate c
            compute p using equation (4.12);
}
}
```

(b) C++ pseudo code using OpenMP

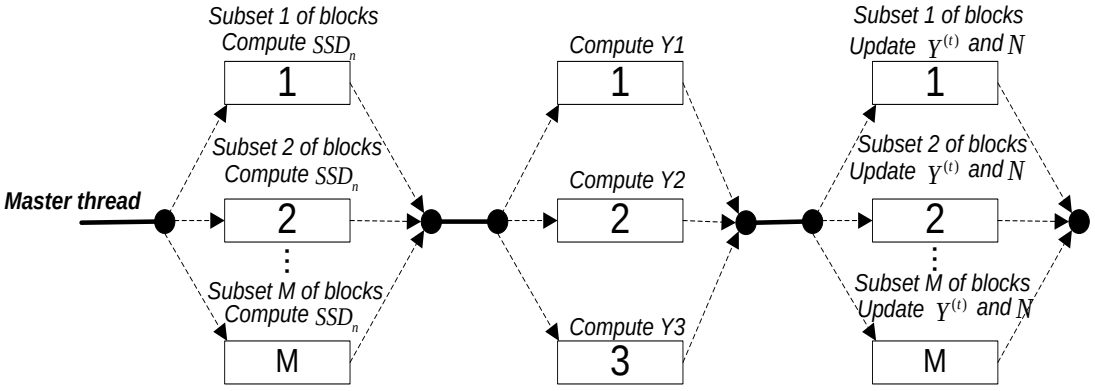
Figure 4.5 Adaptive correlation noise modeling

4.4.3 PGSIS

Figure 4.6 (a) shows the flow chart of PGSIS modules which consists of three parallel regions. The first region computes SSD_n for M subsets of blocks. Each subset can be handled by an independent thread/processor. Then, the three SI candidates are

generated in parallel as well. With the computed SSD_n and three SI candidates, blocks are divided into subsets again for M threads/processors to update SI and residual frame.

Since computing different SI candidates actually takes very similar time, they are allocated into parallel sections to reduce waiting time for synchronization.



(a) Flow chart

```
#pragma omp parallel for reduction(+:E[SSD])
for each 4-by-4 block n {
    compute SSD_n using equation (4.4);
}
compute E[SSD] using equation (4.5);
#pragma omp parallel sections {
#pragma omp section
{ compute Y_1 using equation (4.6); }
#pragma omp section
{ compute Y_2 using equation (4.6); }
#pragma omp section
{ compute Y_3 using equation (4.6); }
}
#pragma omp parallel for
for each 4-by-4 block n {
if SSD_n > E[SSD] {
        compute ε_c using equation (4.7);
        compute ω_c using equation (4.8);
        compute ω̄_c using equation (4.9);
        update Y^(t) using equation (4.10);
        update N using equation (4.11);
    }
}
```

(b) C++ pseudo code

Figure 4.6 PGSIS algorithms

The C++ pseudo code using OpenMP for this routine is shown in Figure 4.6 (b). Blocks are divided into subsets and SSD of each subset of blocks can be computed in parallel. Each thread keeps a separate copy of E[SSD] but they will be summed together using “reduction” syntax to calculate E[SSD] for all blocks.

4.5 Experimental Results and Performance Evaluation

4.5.1 Test Condition

The proposed framework is evaluated by our transform domain DVC codec presented in Section III. We compare the RD performance and the complexity performance with and without the proposed PGSIS algorithm, using the following test conditions.

1) Video sequences: Foreman, Hall Monitor, Coastguard, and Soccer.

2) Number of frames: all frames of the test sequences have been used to evaluate the RD performance which means 150 for Foreman, 165 for Hall Monitor, 150 for Coastguard, and 150 for Soccer. However, since the average complexity performance only shows negligible differences using various numbers of frames, we only use the first 30 frames of all test sequences to evaluate the decoding complexity of PGSIS-DVC codec.

3) Spatial and temporal resolution: QCIF at 15 Hz which means 7.5 Hz for the WZ frames when GOP=2.

4) GOP length: 2, 4 and 8.

5) Eight RD points are considered for DVC codec, corresponding to eight 4×4 quantization matrices widely used in literature [19][37][39].

6) Key frames are coded by H.264/AVC Intra with constant quantization parameters as defined in [19].

7) Software and hardware configuration: the decoding tasks are performed on the Bright Beowulf Cluster Environment using 12 CPU processors of the running node under Linux operating system. The codec is written in C/C++ code and compiled by GCC 4.7.0 using OpenMP 3.1.

The RD performance of the proposed PGSIS-DVC codec will be compared with the same transform domain DVC codec without PGSIS framework. It is also compared with the state-of-the-art conventional coding solutions H.263+ Intra, H.264/AVC Intra and H.264/AVC Inter No Motion. Under similar condition of encoder complexity, i.e. the computationally intensive motion search is not performed by any of them.

4.5.2 RD Performance

1) DVC with PGSIS vs. DVC without PGSIS

The RD performance of the chosen coding solutions for all the selected video sequences is presented in Figure 4.7 and Figure 4.8. The results show that the transform domain DVC codec with PGSIS consistently out performs the same codec without PGSIS for all test sequences and conditions, especially for higher bit-rate and longer GOP sizes. It is expected that the average RD gains increase with bit-rate and the GOP sizes since we use finer quantizer under high bit-rate, which gives better estimates of WZ frame after each reconstruction and therefore PGSIS is able to produce more accurate SI. Similarly, motion interpolation based SI generation technique becomes less effective when the temporal distance between key frames increases (i.e. when the GOP size becomes large), which degrades the quality of the SI but leaves more room for improvements by the PGSIS algorithms. This is an attractive property as most of WZ video codecs do not perform well under long GOP sizes. Another important feature observed from the experimental results is that PGSIS performs better for the sequences with more complex motion, which is also a desirable property since most of other

transform domain WZ codecs perform poorly under this condition. This is as a result of lower quality SI obtained from the poor motion interpolation process so that PGSIS algorithms are able to exploit more correlated information during the decoding process, whereas for low motion video contents, there is not much room for improvement in SI quality and thus their gains of RD performance are little.

The Foreman and Soccer sequences are considered to be of the high motion video contents, whereas Hall Monitor and Coastguard sequences are seen as relatively low motion video contents. As expected, the Foreman and Soccer sequences give better gains in RD performance, particularly the Soccer sequence that achieves the highest RD gains with regards to the DVC codec without proposed algorithms. Taking a close look at Figure 4.7 for the Soccer sequence, notably for the last RD point, there is approximately 1.25 dB for GOP size 2, 1.6 dB and 1.5 dB for a GOP size of 4 and 8, respectively. Similar gains can be observed from the Foreman sequence with about 1 dB for GOP size 2 and 1.3 dB for GOP size 4 and 8.

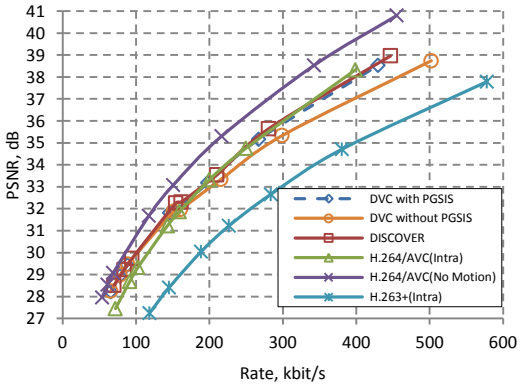
However, the RD curves in Figure 4.8 show slightly lower performance for the sequences of Hall Monitor and Coastguard. The DVC codec with PGSIS performs very close to the one without for the Hall Monitor sequence under most RD points except the last one that gives around 0.7 dB gain for GOP size 2, 1.2 and 1.5 dB gains for GOP size 4 and 8, respectively. Similar gains but within wider range of bit-rates can be seen from the Coastguard sequence. Up to 0.7 dB gain for GOP size 2 and around 1 dB gain for GOP size 4 and 8.

2) PGSIS vs. Standard video coding solutions

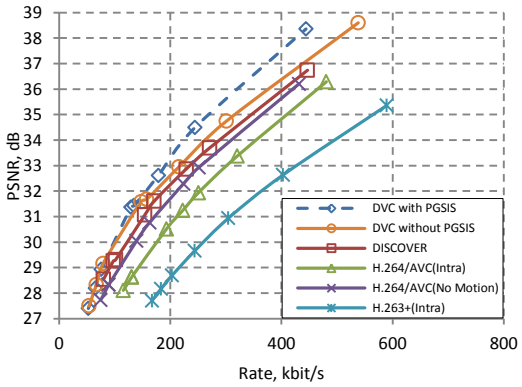
The conventional video coding solutions evaluated here are those widely used standard video codecs. When compared with the RD performance of the PGSIS video codec, it can be concluded that the PGSIS codec out performs H.264/AVC Intra for low

motion sequences, especially for lower GOP sizes. There is also performance gain for more complex video sequence such as Foreman, under low bit-rate and high GOP size.

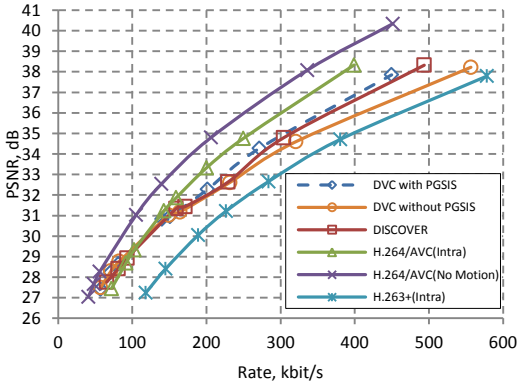
(a) Foreman: GOP=2



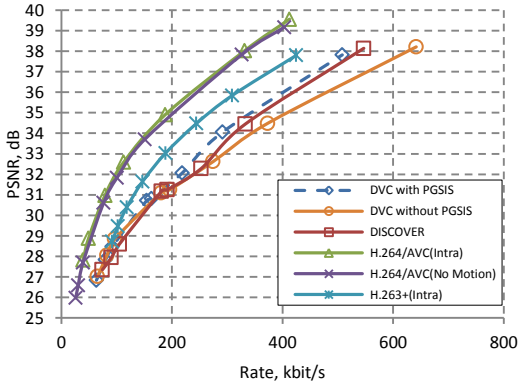
(b) Soccer: GOP=2



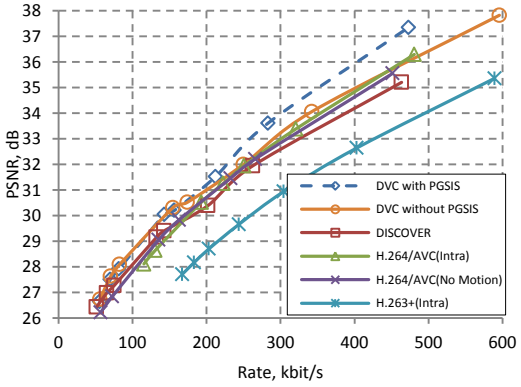
(c) Foreman: GOP=4



(d) Soccer: GOP=4



(e) Foreman: GOP=8



(f) Soccer: GOP=8

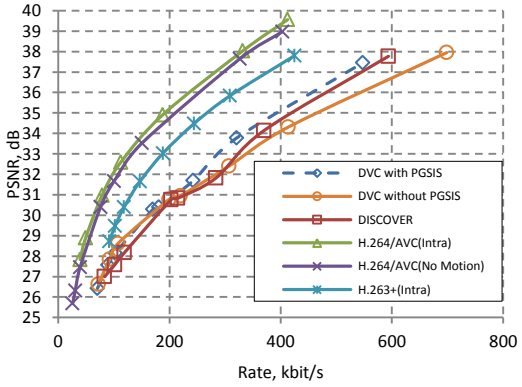
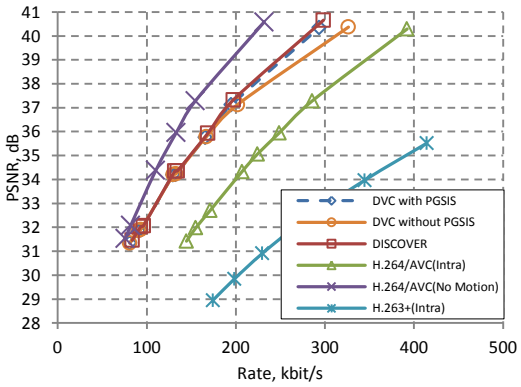


Figure 4.7 RD Performance for Foreman and Soccer sequences

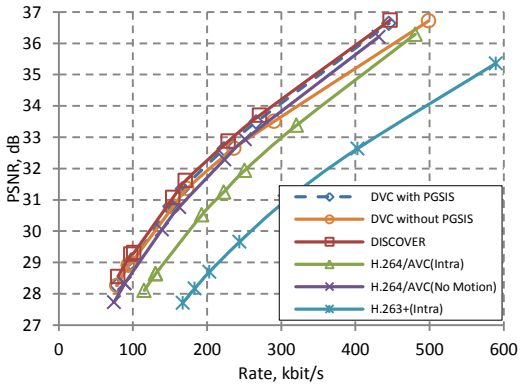
It is usually expected that WZ codec can hardly beat the performance of H.264/AVC No Motion. However, the PGSIS codec shows remarkable RD gains for high motion video sequences. Foreman sequence with GOP size 8, Soccer sequence with GOP size 2

and Coastguard sequence using GOP sizes 2 and 4, all performs better than H.264/AVC No Motion. However, there are no significant RD performance changes for the Hall Monitor sequence, i.e. the performance remains above H.264/AVC Intra and still below H.264 No Motion.

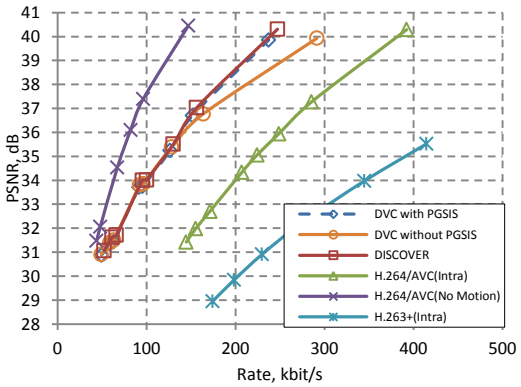
(a) Hall Monitor: GOP=2



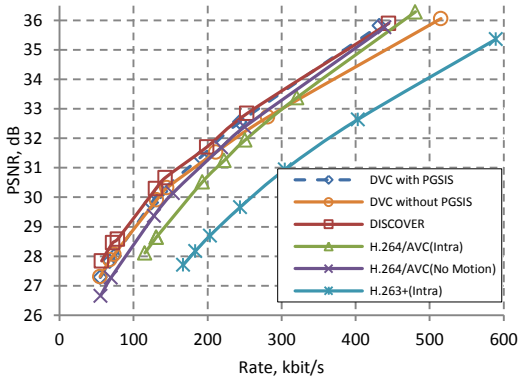
(b) Coastguard: GOP=2



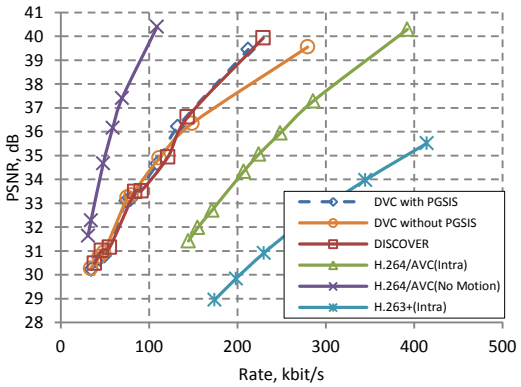
(c) Hall Monitor: GOP=4



(d) Coastguard: GOP=4



(e) Hall Monitor: GOP=8



(f) Coastguard: GOP=8

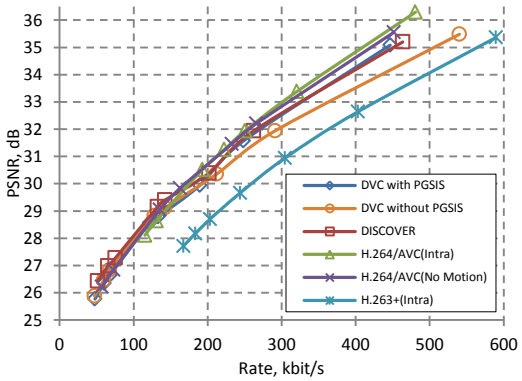


Figure 4.8 RD Performance for Hall Monitor and Coastguard sequences

It can also be observed that, for low motion sequences such as Hall Monitor and Coastguard, PGSIS-DVC remains above or similar to H.264/AVC Intra under most situations. However, for high motion sequences such as Foreman and Soccer, the RD performance is still below H.264/AVC Intra for most settings. Comparing with H.263+ (Intra) codec, PGSIS is consistently better with exception for the most complex sequence Soccer, which only shows superior RD performance using GOP size 2.

We have also presented the RD performance of DISCOVER codec as a benchmark. However, DISCOVER uses some more advanced modules, such as a dead-zone quantizer. For simplicity these modules were not used in PGSIS, so they cannot be directly compared. Despite these RD curves of PGSIS DVC are in general above or similar to the performance of DISCOVER codec. A notable 2 dB gain can be observed from the last point of Foreman sequence of GOP size 8. However, more gains can be expected if PGSIS-DVC utilizes the same modules as DISCOVER.

4.5.3 Complexity Analysis

In terms of encoding complexity, a thorough analysis of a DVC codec which shares similar encoding architecture as the encoder of this chapter is presented in [26]. The results show that for GOP size of 2, DVC encoding complexity is about 60-70% of H.264/AVC Intra and H.264/AVC (No Motion). Much more gains can be obtained with longer GOP sizes, but even with GOP size 2, it already has much lower encoding complexity and defeats the RD performance of H.264/AVC Intra for most test sequences, not to mention the performance of H.264/AVC (No Motion) which has slightly higher complexity than H.264/AVC Intra.

Table 4.1 PGSIS-DVC Decoding Time For the Parallel and Serial Implementations. The Main Components and Total Decoding Time (in seconds) are Presented for Different Quantization Parameters Using Fixed Group of Picture Size of 2. The Parallel Architecture Employs 12 CPU Cores. Parallel (P), Serial (S), Initial SI Creation (I), Correlation Noise Modelling (C), PGSIS (G), LDPCA Decoding (L), Total Decoding Time (T)

Sequences	**Components**	*Q1(P/S)*	*Q4(P/S)*	*Q8(P/S)*
Foreman	**I**	12.92/25.06	13.04/24.68	14.03/24.36
	C	1.78/7.33	3.89/13.95	5.34/16.07
	G	47.30/195.89	156.91/642.75	244.74/959.38
	L	7.24/53.65	19.27/143.59	43.95/337.86
	T	69.33/282.36	193.25/825.68	308.25/1338.57
Soccer	**I**	13.59/23.32	13.11/26.76	13.10/24.29
	C	1.69/7.04	3.85/13.50	6.12/15.41
	G	50.61/187.70	165.46/689.76	244.17/959.19
	L	10.25/65.79	24.64/190.50	42.31/332.65
	T	76.23/284.21	207.22/921.32	305.89/1332.35
Coastguard	**I**	13.17/27.67	13.48/30.60	13.20/30.92
	C	1.94/7.88	4.33/15.96	6.67/18.41
	G	48.93/208.28	165.17/733.99	245.26/1093.22
	L	2.13/16.55	10.68/78.28	48.08/372.31
	T	66.28/260.98	193.86/860.10	313.43/1516.44
Hall	**I**	13.94/26.35	13.34/26.85	16.24/27.45
	C	1.98/9.12	5.41/18.96	8.02/22.17
	G	50.17/203.25	164.28/684.67	298.73/1037.24
	L	2.08/16.01	7.60/56.39	25.98/184.76
	T	68.30/255.37	190.84/788.14	349.33/1273.32

The PGSIS-DVC codec is implemented in parallel as well as in serial, where the serial implementation can be seen as the parallel version that uses only one CPU core.

The complexity performance of only the parallelized modules: initial SI creation (I), correlation noise modeling (C), PGSIS (G), and LDPCA decoding (L), are presented in Table 4.1. It can be seen that the majority of the computational time is spent on I, G, and L. The parallelized initial SI generation is about 2 times faster than the serial version, and the channel modeling and PGSIS are both about 4 times faster, whereas the LDPCA decoding module is about 7 times faster. In addition, the impact of video content on the decoding time meets the common expectation, i.e. the more complex the video content is, the more time requires decoding that sequence. The video sequence that contains highest motion content here is Soccer, which therefore takes longest time to decode. However, it is not always the opposite for the slowest video content tested here.

We have also included the decoding time for LDPCA decoder for both parallel and serial implementations in Table 4.1 as this module is usually considered to be the most complex component in DVC. For further details on the decoding algorithm and parallel implementation the reader can refer to [62] as similar methods are used here.

The limitations of the proposed PGSIS-DVC codec and their possible solutions are summarized here. It can be seen that PGSIS and SI generation takes more time than the LDPCA decoding. In addition, for slow motion sequences (e.g. Hall Monitor) the computing time is not remarkably less than that of the faster motion sequences (e.g. Soccer) despite the fact that fewer blocks are processed during the refinement process. Furthermore, the time reduction brought by the parallel implementation is not proportional to the number of CPU used. The above are mainly due to the following reasons. Firstly, due to the source code availability, we have adopted a highly complex serial optical flow algorithm, which takes about 80-90% of the total computational time of PGSIS and therefore, the time reduction brought by block filtering can hardly

observed here. Secondly, from section 4.4.1 and 4.4.3, it can be seen that only 2 and 3 parallel sections (Figure 4.4 (a) and Figure 4.6 (a)) are used for initial SI creation and PGSIS, respectively, although 12 CPU cores have been used. Therefore, it can be expected to have speedup of only about 2× and 3× compared with the serial implementation. However, we stress that the optical flow component is completely independent of the proposed algorithm so it can be replaced with a much more efficient one and apply parallel techniques to reduce the complexity cost further. For example, a fast parallel implementation of an optical flow algorithm presented in [75] is able to compute both forward and backward flows in about 3 seconds per frame for about 12 times larger frame size (640×480) than QCIF used in our experiments, running at a single machine equipped with cheap GPU hardware. Furthermore, in our implementation the optical flows are computed for all blocks within the WZ frames. However, this could be reduced by performing the computations on the selected blocks that are already available within the PGSIS refinement process, particularly for slow motion sequences.

4.6 Conclusion

This chapter presents a pixel level SIS framework and parallel implementation within a state-of-the-art transform domain DVC codec. The experimental results show significant improvements on RD performance over the same codec without the proposed algorithms. The parallel implementation also shows high utilization of resources and substantial speedup when compared with the serial implementation. The updated SI frames during the SIS process demonstrate considerable improvement in both subjective and objective image quality against the widely used block based SIS algorithms. The proposed SIS framework can be integrated into any modern transform domain DVC codec to achieve a better RD performance especially for video sequences

with complex motion and coded with long GOP sizes. The framework can also be re-configured to exploit more efficient optical flow algorithms to improve the performance and further reduce complexity. Furthermore, the proposed parallel implementation brings the state-of-the-art DVC codec one step closer to practical use.

Chapter 5
Consistent Quality Control for Wireless Video Surveillance

5.1 Introduction

Video transmission over wireless links is unreliable and can be characterized by bursty and high channel error probability. Since channel coding is adopted in DVC, this brings another appealing property that it is resilient against transmission errors. As wireless networks have limited bandwidth, rate control algorithms are usually required to achieve the best overall quality at the minimum bit rate cost. However, the priority is given to control the target bit-rate without regard to a stable visual quality along time. It is also important to notice that conventional DVC systems are weak in coordinating the key frames encoder and WZ frames encoder due to the separation of the encoding process. This can be characterized by using fixed quantization settings [19][26][35] for the coding of key frames and corresponding WZ frames. The fixed quantization parameters are typically obtained from iterative offline experiments. However, constant quantization configurations cannot adapt the changes in visual content and offline training approaches are impractical for real time video surveillance systems. Furthermore, inappropriate distortion distribution in key frames and WZ frames can seriously degrade RD performance.

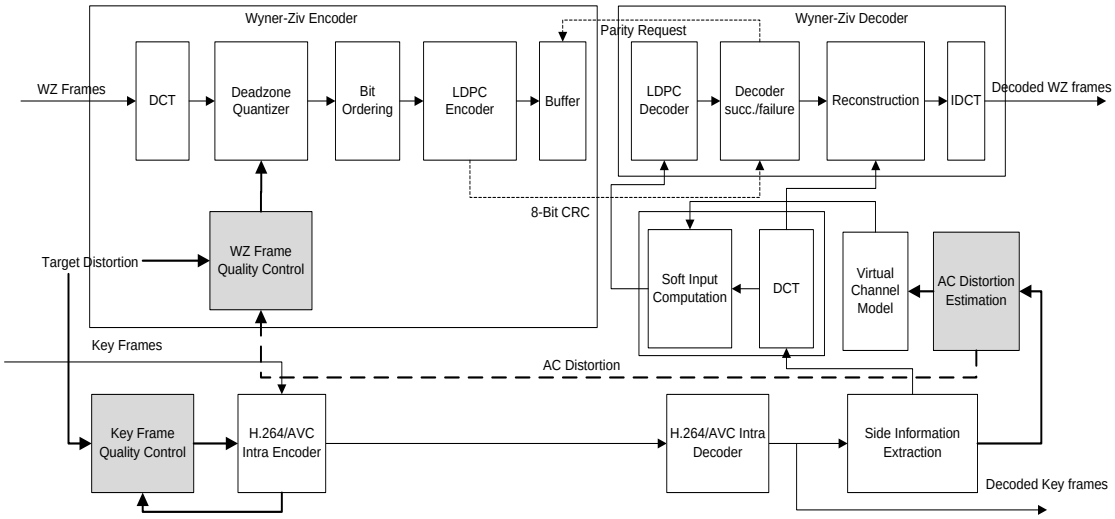


Figure 5.1 Overall System Architecture

In this chapter, we propose a novel algorithm to facilitate key frames and WZ frames encoder quality control. The proposed algorithm adjusts the quantization parameters according to the visual content and the user defined target quality online without any external control. A DQ model derived from MPEG-2 distortion estimation model [76] is employed. With the proposed algorithm, low complexity encoding is still guaranteed by performing the distortion estimation partly at the decoder side. The information required from the decoder is sent through the existing feedback channel.

The rest of this chapter is organized as follows. Section 5.2 presents the overall system architecture of our DVC codec. In Section 5.3 and Section 5.4, we describe the proposed quality control solutions for the key frames and WZ frames, respectively. The simulation results are given in Section 5.5 and finally, Section 5.6 concludes this chapter and gives a brief outlook on future work.

5.2 System Architecture

The proposed DVC codec depicted in Figure 5.1 is based on the Stanford architecture [11] summarized as follows:

(1)The input video is divided into key frames and WZ frames. Key frames are inserted periodically determined by group of pictures (GOP) size.

(2)The WZ frames are further divided into 4-by-4 blocks and in each block, discrete cosine transform (DCT) and a uniform quantization with dead-zone are performed.

(3)The quantization matrix (QM) is determined by the WZ frames quality control algorithm, given user defined target distortion in terms of peak signal noise ratio (PSNR).

(4)The quantized DCT coefficients are grouped into frequency bands and converted into bit-planes.

(5)Each bit-plane is separately encoded using low-density-parity-check (LDPC) codes and stored in a buffer for decoder requests. An 8-bit cyclic redundancy check (CRC) code is also generated for each bit-plane to confirm decoding is success.

(6)Key frames are encoded by an efficient conventional coder such as H.264/AVC Intra, where the quantization parameters (QP) are determined by the key frame quality control algorithm given the same target distortion.

(7)At the decoder side, decoded key frames and WZ frames are interpolated to generate SI.

(8)The correlation noise between SI and WZ frames are assumed to be Laplacian distributed and modelled by the virtual channel model.

(9)Distortion of AC coefficients is estimated by residual statistic information of the decoded key frames and sent back to the encoder to aid WZ frame quality control. These results are further utilized in the virtual channel modelling process.

(10) The soft input to the LDPC decoder in terms of conditional bit probability is calculated, using the statistical information provided by the virtual channel model.

(11) An iterative decoding process is performed until the syndrome check and CRC check are both successful. More parity bits can be requested if the above stopping criterion is not met.

(12) Finally, all the decoded quantized symbols are optimally reconstructed [23] and inverse transformed.

5.3 Key frames Quality Control

The key frames quality control algorithm shown in Figure 5.2 is mainly derived from [57] and works in frame level. It consists of two main modules. The key frame DQ model estimates the distortion of key frames and selects a proper QP for the conventional intra encoder. Its parameters are online updated using previously encoded key frames.

5.3.1 Key Frame DQ Modelling

The key frames distortion as a function of quantization step size Q_s is estimated by the DQ model in Equation (5.1),

$$D(Q_s) = a \cdot {Q_s}^{c_0} + b \tag{5.1}$$

where a and b are frame dependent model parameters. c_0 is typically a constant and $c_0 = 1.32$ as in [57] is used here.

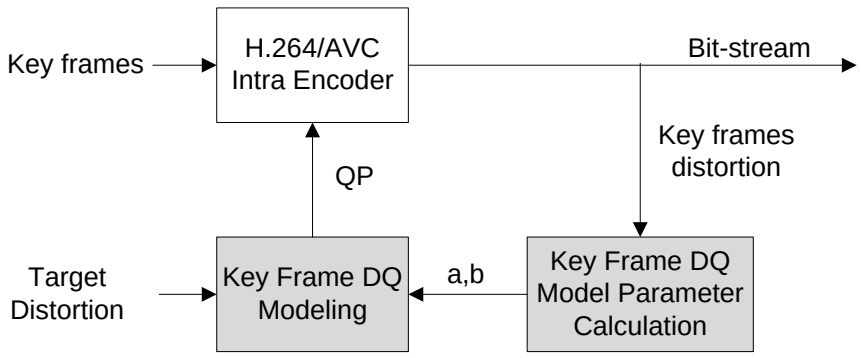


Figure 5.2 Key frames quality control

According to H.264/AVC standard [1], quantization is controlled by an integer QP. Each QP value corresponds to a Q_s value. The ratio between successive Q_s values is $\sqrt[6]{2}$, so that Q_s doubles in value when QP increases by six. Therefore, any $Q_s$value can be derived from Equation (5.2) using the first six $Q_s$values in Table 5.1,

$$Q_s(QP) = Q_s[QP \ mod \ 6] \times 2^{\lfloor QP/6 \rfloor} \tag{5.2}$$

where mod is the modulo operation, $Q_s[x]$ is a value in Table 5.1 indexed by x and $\lfloor . \rfloor$ denotes the nearest integer smaller than the given number.

Table 5.1: The First 6 Q_s Values

QP	***0***	***1***	***2***	***3***	***4***	***5***
Q_s	0.625	0.702	0.787	0.884	0.992	1.114

The QP is chosen as follows,

1) Encode the first 2 key frames with some predefined initial QP values;
2) Calculate the model parameters $a$and b, which will be discussed in the following section;
3) Use the model parameters to estimate key frames distortion for each given QP as in Equation (5.2), in case of H.264/AVC Intra $QP \in [0,51]$;
4) The QP that produces the key frame distortion D_{KF} best matches the target distortion D_T is chosen, i.e.

$$QP = arg \min_{QP \in [0,51]} |D_{KF} - D_T| \ with \ D_{KF} < D_T \tag{5.3}$$

5) Encode the first 2 key frames again with the new QP and all the rest of the key frames will be encoded following the same procedures in 2) to 4).

We restrict the distortion of key frames in Equation (5.3) to be the closest but lower than the target distortion to guarantee a better quality of key frames, since the quality of

decoded WZ frames are strongly dependent on the quality of the key frames. This unbalanced relationship is a trade-off to provide a good overall RD performance.

5.3.2 Key Frame DQ Model Parameters Calculation

The parameters a and b in Equation (5.1) can be calculated from previously encoded key frames. Assuming the GOP size is 2, then the distortion $D^t$and D^{t+2} for frame t and frame $t+2$, respectively, introduced by QP^t and QP^{t+2} can be calculated as Equation (5.4),

$$\begin{cases} D^t = a \cdot Q_s (QP^t)^{c_0} + b \\ D^{t+2} = a \cdot Q_s (QP^{t+2})^{c_0} + b \end{cases} \tag{5.4}$$

The model parameters can thus be online updated using Equation (5.5),

$$\begin{cases} a = \dfrac{D^t - D^{t+2}}{Q_s (QP^t)^{c_0} - Q_s (QP^{t+2})^{c_0}} \\ b = D^t - \dfrac{(D^t - D^{t+2}) \cdot Q_s (QP^t)^{c_0}}{Q_s (QP^t)^{c_0} - Q_s (QP^{t+2})^{c_0}} \end{cases} \tag{5.5}$$

However, due to the similarity of visual characters in adjacent frames, the QP obtained from Equation (5.3) for current frame t can be the same as in frame $t+2$, which results in zero denominator in Equation (5.5). In this case, a previously recorded different QP with its corresponding distortion D will be chosen to solve Equation (5.5).

5.4 WZ Frames Quality Control

The objective of WZ frames quality control algorithm depicted in Figure 5.3 is to choose a QM which can meet the target distortion. A DQ model for WZ frames is needed to estimate the distortion introduced by a candidate QM. A DQ model derived from [76] is employed in this chapter.

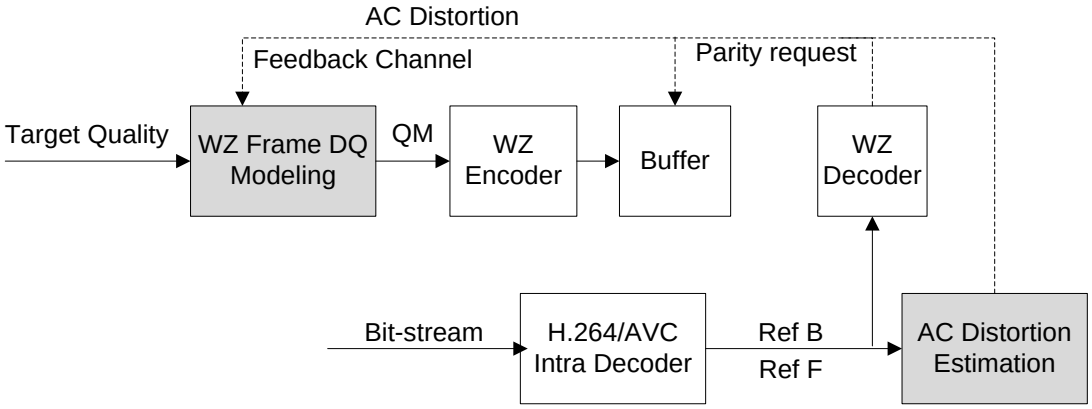


Figure 5.3 WZ frames quality control

5.4.1 WZ Frame DQ Modelling

The model given in Equation (5.6) estimates the average WZ distortion D_{WZ} at frame level, considering the distortion contributions of each coefficient,

$$D_{WZ} = \frac{1}{16}[\varepsilon_{DC}^2 + \sum_{i=1}^{15} \varepsilon_{AC}^2(i)] \tag{5.6}$$

where i is the index of AC coefficients. The overall distortion is divided by two parts, i.e. the distortion of DC coefficients ε_{DC}^2 and the distortion of AC coefficients ε_{AC}^2. The average DC distortion of a MPEG-2 coded frame is calculated using Equation (5.7),

$$\varepsilon_{DC}^2 = 2^{10-IDP} \tag{5.7}$$

where IDP is intra DC precision in MPEG-2 which controls the quantization coarseness of DC coefficients. In our DVC system, quantization coarseness is controlled by the quantization level in QMs. Each quantization level can be represented by κ bits, so IDP in Equation (5.7) is replaced by κ in our calculation.

The estimation of AC distortion will be discussed in detail in the following section.

5.4.2 AC Distortion Estimation

The average squared quantization error ε_{AC}^2 for MPEG-2 coded AC coefficients is obtained as Equation (5.8),

$$\varepsilon_{AC}^2 = 2\lambda^2 - \frac{2\lambda Q_s e^{-\alpha/\lambda} e^{-Q_s/2\lambda}}{1 - e^{-Q_s/\lambda}} [\frac{\alpha}{\lambda} + 1] \tag{5.8}$$

where the original AC coefficients are assumed to follow a Laplacian distribution with parameter λ, Q_s is the quantization step size and α is the offset of the reconstruction window in MPEG-2 TM5. We ignore α in Equation (5.8) by assigning a zero offset so that Equation (5.8) can be refined as follows,

$$\varepsilon_{AC}^2 = 2\lambda^2 - \frac{2\lambda Q_s e^{-Q_s/2\lambda}}{1 - e^{-Q_s/\lambda}} \tag{5.9}$$

λ is given by Equation (5.10) as,

$$\lambda = -\frac{Q_s/2}{\ln[1 - p_0(Q_s)]} \tag{5.10}$$

where $p_0(Q_s)$ is the ratio of the number of zero coefficients over all the coefficients quantized by Q_s.

However, for a coarse QM some of the high frequency bands of AC coefficients are not coded, which means no data for these frequency bands is transmitted from the encoder. We denote these AC coefficients as AC_0. In the reconstruction process, AC_0 are taken directly from the SI. Obviously, Equation (5.9) does not consider this situation. An insight into the distortion of AC_0 shows that the distortion of these frequency bands is actually the difference between SI and WZ frames, i.e. the correlation noise. The correlation noise is typically modelled by the statistical distribution of the residual of reference frames. Here, we use Mean Squared Error (MSE) of the residual frame to estimate the distortion of AC_0. Residual frame R is generated from the motion

compensated versions of backward reference frames K_B and forward reference frames K_F as Equation (5.11):

$$R(x,y) = \frac{K_B(x+dx_b, y+dy_b) - K_F(x+dx_f, y+dy_f)}{2} \tag{5.11}$$

where $K_B(x+dx_b, y+dy_b)$ and $K_F(x+dx_f, y+dy_f)$ represent the backward and the forward motion compensated frames, respectively. In (5.11), (dx, dy) represent the motion vector and (x, y) is the pixel location in frame R. Therefore, the distortion of AC_0, ε_{AC0}^2 is calculated in Equation (5.12),

$$\varepsilon_{AC_0}^2 = E_i[R(x,y)^2] \tag{5.12}$$

where $E_i[.]$ is the expectation operator over all the coefficients in band i. ε_{AC}^2 is now rewritten as in the following,

$$\varepsilon_{AC}^2 = \begin{cases} 2\lambda^2 - \dfrac{2\lambda Q_s e^{-Q_s/2\lambda}}{1 - e^{-Q_s/\lambda}}, if\ Q_L \neq 0 \\ \varepsilon_{AC_0}^2, \qquad else \end{cases} \tag{5.13}$$

where Q_L represents the quantization level for the AC band.

The distortion of DC coefficients combined with all AC distortion contributes the distortion of a WZ frame. Thus, we can choose the QM by Equation (5.14),

$$QM_i = arg \min_{i \in [1,8]} |D_{WZ} - D_T| \tag{5.14}$$

Here, 8 QMs in [26] indexed by i are used in our DVC codec. The QM that gives the closest distortion to the target distortion is selected.

5.5 Simulation Results

The proposed algorithm is evaluated by our DVC codec presented in Section 5.2. We compare the distortion variation and the RD performance of the basic DVC codec

with and without proposed algorithm. Only luminance components of Hall Monitor and Coastguard representing videos of different types of motion are used. Both test sequences are of size QCIF (176×144) at temporal resolution of 15 Hz. All frames in the test sequences are used, which means 165 frames for Hall Monitor and 150 frames for Coastguard. A constant GOP size of 2 is used for all test sequences, i.e. odd frames are key frames whereas even frames are WZ frames. The QMs defined in [26] are applied in our simulation to determine the quantization levels. A regular degree LDPC accumulate code [22] of length 1584 bits is used for virtual channel coding.

All frames are coded with constant QM-QP pairs defined in Table 5.2 [26] when no quality control is performed. Table 5.2 is obtained by iterative offline training process targeting to have almost constant decoded video quality for both key frames and WZ frames. In the following experiments, Hall monitor and Coastguard are coded with QM_1 and QM_4, respectively, when using fixed quantization settings.

5.5.1 Distortion Variation

Both key frames and WZ frames quality control algorithms are verified by the temporal PSNR variation in this section. The distortion of test sequences is also compared with target PSNR which is set to be equal to the average PSNR over all frames obtained in the coder without quality control.

Table 5.2: QP Values for Corresponding QMs of the Basic DVC Codec without Proposed Quality Control

	QM_1	QM_2	QM_3	QM_4
Hall Monitor	37	36	36	33
Coastguard	38	37	37	34

1) Key Frames Distortion Variation

Figure 5.4 and Figure 5.5 show the temporal PSNR variation for the sequences Hall Monitor and Coastguard, respectively. It can be seen from both figures that the key frames distortion varies rather small for slow motion and fast motion sequences, regardless of quality control. More specifically, the key frames PSNR variances of Hall Monitor with and without quality control are 0.0152 and 0.0069, respectively. However, this trivial variance increase introduced by quality control brings PSNR around 1 dB closer to the target. Similar results are obtained from Coastguard sequence. The key frames PSNR variance increased from 0.0424 to 0.0632 using quality control but again, it provides decoded quality more than 1.5 dB closer to the target.

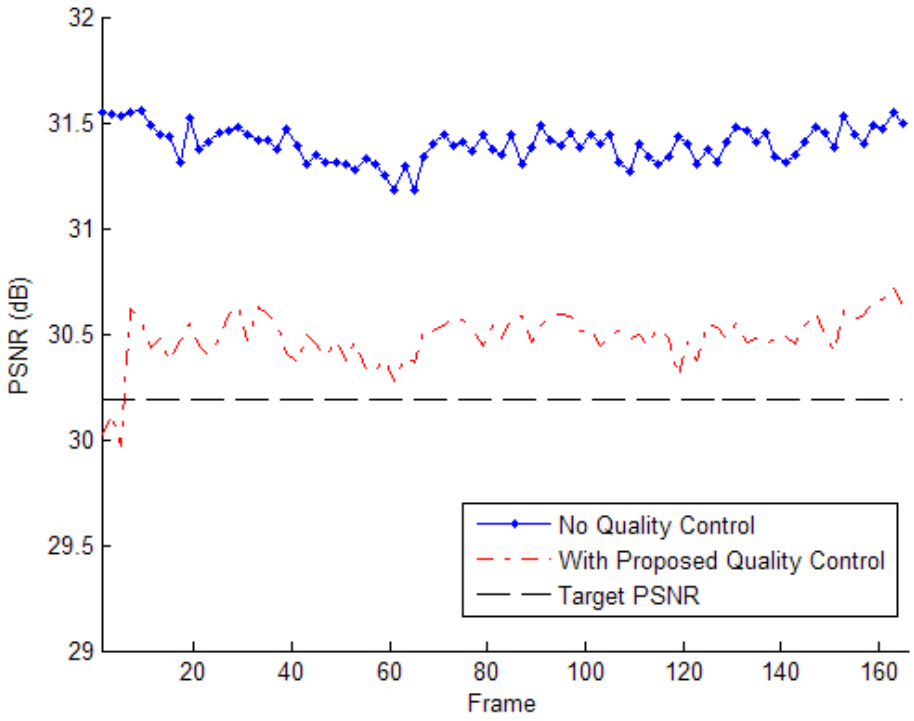


Figure 5.4 Temporal PSNR variation for the key frames for Hall Monitor

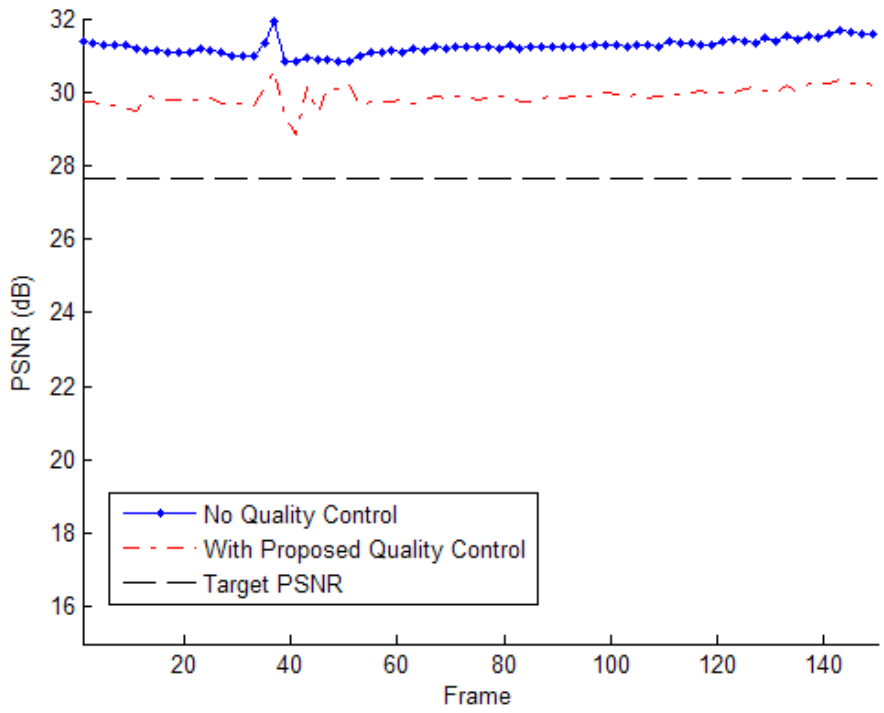


Figure 5.5 Temporal PSNR variation for the key frames for Coastguard

2) WZ Frames Distortion Variation

Figure 5.6 and Figure 5.7 show the results of WZ frames distortion variation for Hall Monitor and Coastguard, respectively. The proposed algorithm reduced the PSNR variance from 0.2138 to 0.0636 and better met the target PSNR by about 1 dB for sequence Hall Monitor. However, the algorithm performs similar to the fixed quantization settings obtained from offline training for the Coastguard sequence. Only a small reduction from 0.5434 to 0.5242 on PSNR variance is obtained using quality control. It gives smoother image quality closer to the target only in the later part (after 100 frames) of the sequence when the scene changes tend to reduce.

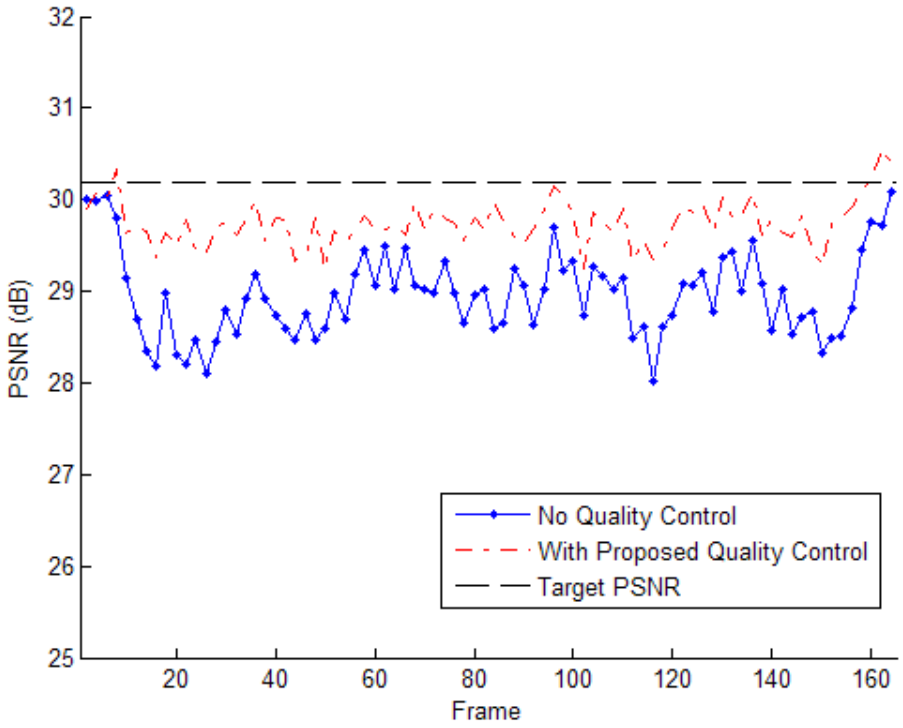


Figure 5.6 Temporal PSNR variation for the WZ frames for Hall Monitor

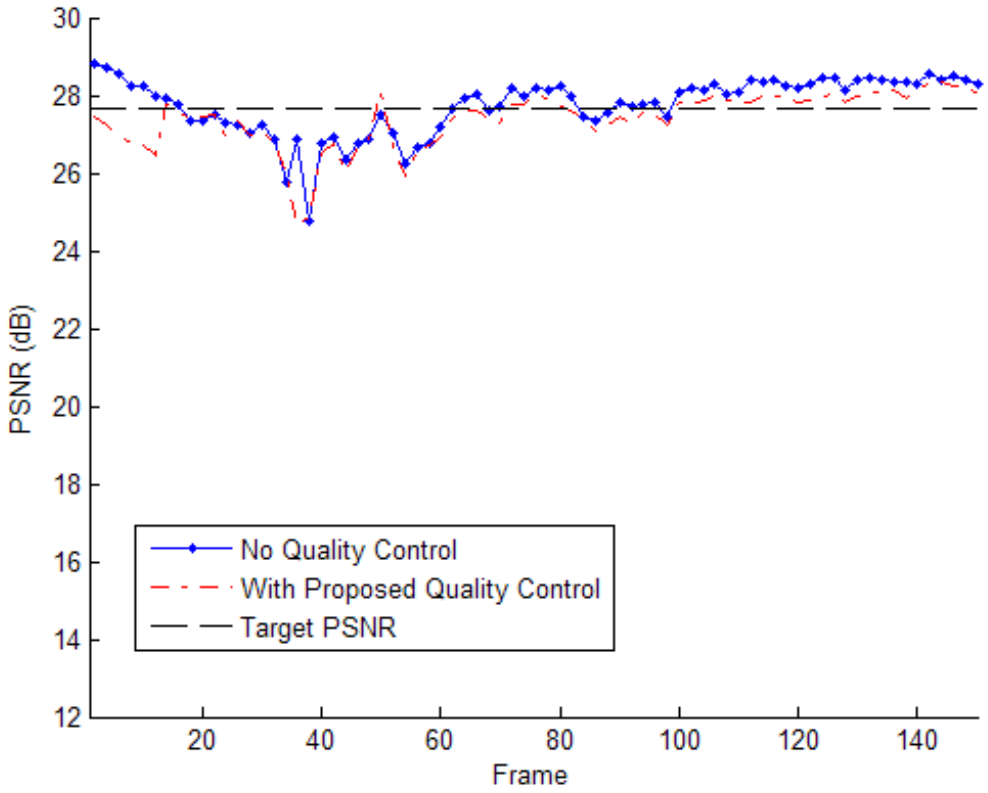


Figure 5.7 Temporal PSNR variation for the WZ frames for Coastguard

5.5.2 RD Performance

Figure 5.8 and Figure 5.9 show the RD performance of sequence Hall Monitor and Coastguard, respectively. The first 4 coarse QMs in [26] with their corresponding QPs defined in Table 5.2 are used in the basic DVC codec without proposed quality control algorithm, which correspond to 4 RD points. RD performance loss of up to about 2.5 dB is observed in Figure 5.8 when using quality control, whereas a smaller loss of up to about 0.6 dB is observed in Figure 5.9. It is also important to notice that the average distortion of key frames and WZ frames are rather similar in Figure 5.4 and Figure 5.6, when using proposed quality control. However, more priority is given to key frames which result in rather big difference in the average distortion of key frames and WZ frames in Figure 5.5 and Figure 5.7. A similar average distortion in both key frames and WZ frames gives a rather degraded RD performance, whereas an unbalanced average distortion gives a smaller loss in RD performance, which implies the need for balance between smooth quality control and RD performance. This has also been reported in [26] that allocating more bits to the key frames at the cost of a less stable video quality may lead to a better RD performance.

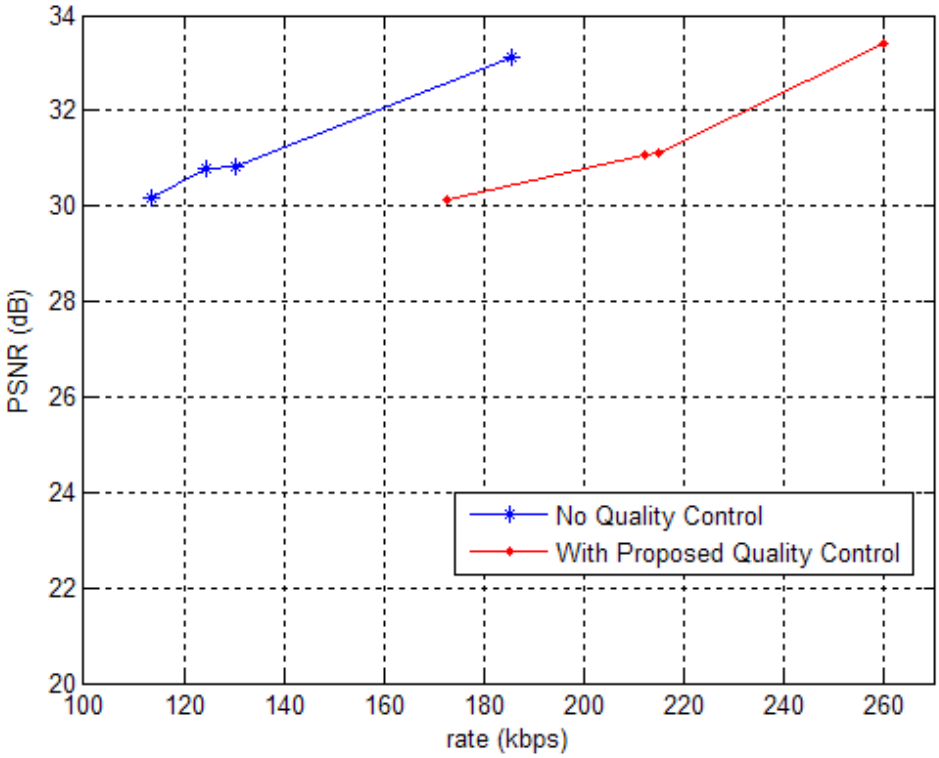


Figure 5.8 RD performance for Hall Monitor

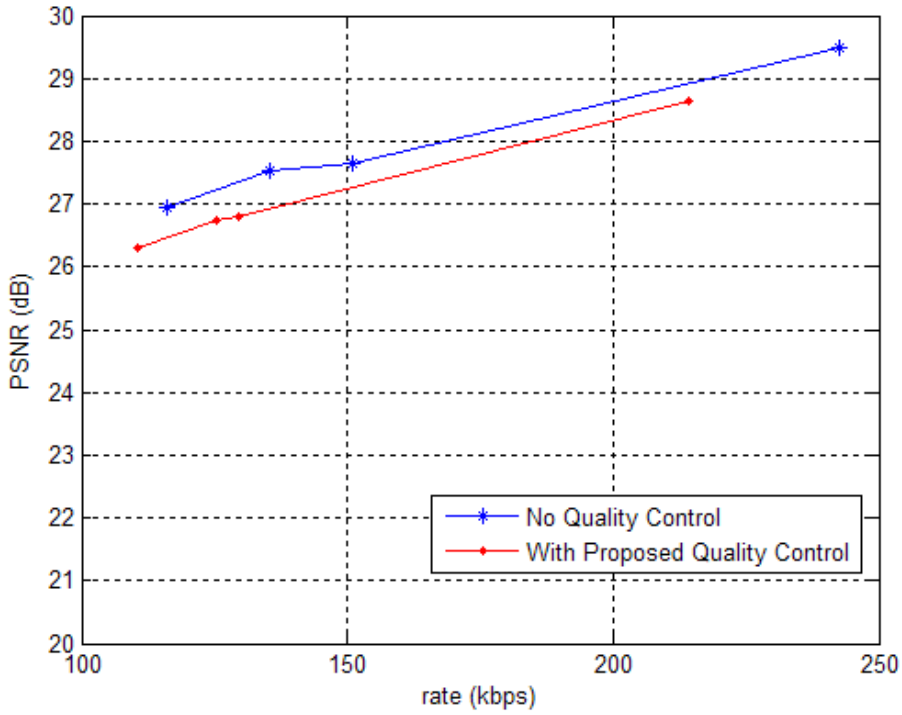


Figure 5.9 RD performance for Coastguard

5.6 Conclusion and Future Works

In this chapter, we have presented an efficient technique to automatically control the video quality for DVC codecs. Simulation results show that the proposed method closely meet user defined target quality and smooth out the distortion variation for slow motion sequences and performs similar to fixed quantization settings obtained from offline trainings for fast motion sequences. However, some RD performance loss is observed in our quality-controlled DVC codec.

A flexible control to balance a smooth quality and RD performance remains as our future work.

Chapter 6
Low Complexity Implementation of DVC Codec

6.1 Introduction

The first software implementation of DVC were developed in 2002, by Stanford University [11][77] and UC Berkeley [9][10], using different frameworks and verified by simulations on general purpose personal computers (PC). In 2007, a European project called DISCOVER [19], based on the Stanford framework, implemented an efficient DVC codec also on a general purpose PC. It presents a state-of-the-art low-complexity DVC codec, as well as a benchmark for DVC implementations. But all the above frameworks have not been verified in a practical system architecture, considering the memory and processors restriction.

In this chapter, we present the first implementation of a DVC encoder using low-density-parity-check accumulative codes (LDPCA) on Texas Instruments TMS320C6437 fixed point DSP. We present an efficient implementation utilizing the DSP hardware features and optimization techniques particularly in-place Discrete Cosine Transform (DCT) transform, software pipelining and built-in LDPCA codes. The decoder is running on a general purpose PC. Furthermore, the Stanford DVC framework is verified on a DSP based encoder and PC based decoder, together referred to as DSP-PC architecture.

Furthermore, we present a parallel implementation of DVC decoder based on a PC based encoder and HPC (high performance cluster) based decoder, together referred to

as PC-HPC architecture, where the encoder is running in a general purpose PC and the decoder is running in a multicore HPC.

The rest of this chapter is organized as follows. In Section 6.2, we describe the DSP-PC system architecture and the target platforms of the encoder and the decoder. Implementation details of encoder components such as DCT transform, coefficients to bit-stream conversion, LDPCA encoding and key frames encoding are provided in section 6.2.6 to section 6.2.9. And the performance evaluation of the encoder complexity and the overall RD performance are presented in section 6.2.10. In Section 6.3, the focus is moved to the PC-HPC architecture where we implement a highly efficient decoder using parallel technology. Section 6.3.2 to section 6.3.5 provide technical details for the encoder, especially covers the quantizer design, LDPCA encoding and file structure organization. Details of the parallel implementation for the modules of SI generation, correlation noise modelling, conditional bit probability computation and LDPCA decoding are presented in section 6.3.7 to section 6.3.10. The decoding complexities as well as the RD performances are given for different experimental scenarios in section 6.3.11. Final conclusions and future works are given in Section 6.4.

6.2 DSP-PC DVC Implementation and Optimization

6.2.1 System Overview

The system level diagram of our implementation is presented in Figure 6.1. The WZ encoder using LDPCA codes and a conventional intra-frame encoder employing JPEG Baseline [78] coding approach are implemented on a DSP, connected to a PC through an embeded JTAG emulator. The bit-streams are generated and stored on PC, in which the WZ decoder and the conventional decoder are implemented. The WZ encoder deals

with all the even frames while the conventional encoder processes the remaining frames. This DVC coding scheme works in transform domain. This chapter focus more on the implementation issues, the reader is referred to [3] for further information and details on DVC.

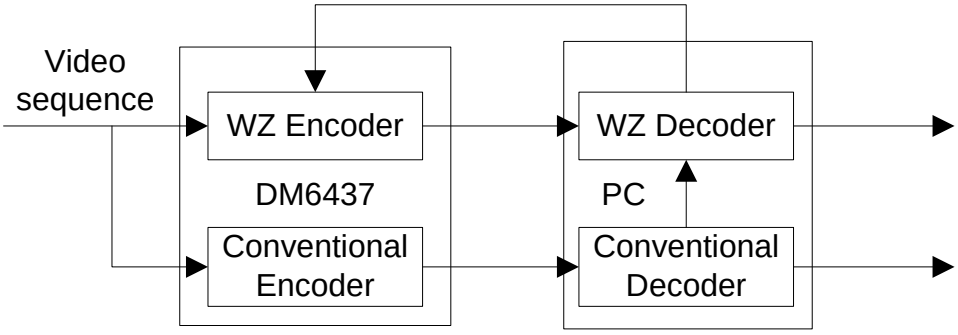


Figure 6.1 System Architecture

6.2.2 Encoder Architecture

The encoder functional blocks are shown in Figure 6.2. The JPEG encoder divides a video frame into square blocks with equal block length. DCT and a fixed quantization (DCT&Q) are performed over each block. The quantized DC coefficients are encoded by differential pulse-code modulation (DPCM). The encoding of the quantized AC coefficients is performed by run-length encoding (RLE) algorithm. The DPCM encoded DC coefficients and RLE encoded AC coefficients are further compressed by an entropy encoder.

The WZ encoder also performs block-wise DCT and quantization on the video frames. The quantization table used in JPEG encoder is employed in WZ encoder as well. The quantized coefficients are then converted into bit-streams and coded using a LDPCA code.

Parity bits and compressed key frame bit-streams are generated on the DSP board and sent to PC through an emulator.

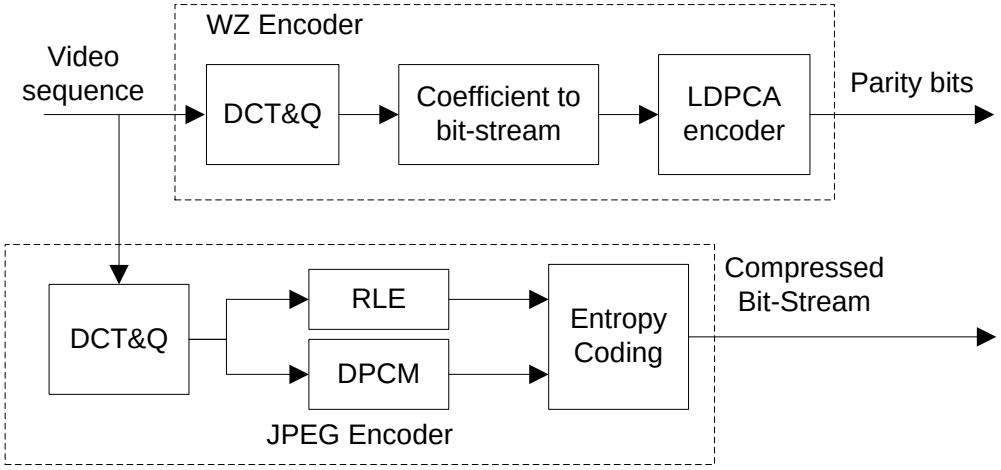


Figure 6.2 Encoder Functional Block Diagram

6.2.3 Decoder Architecture

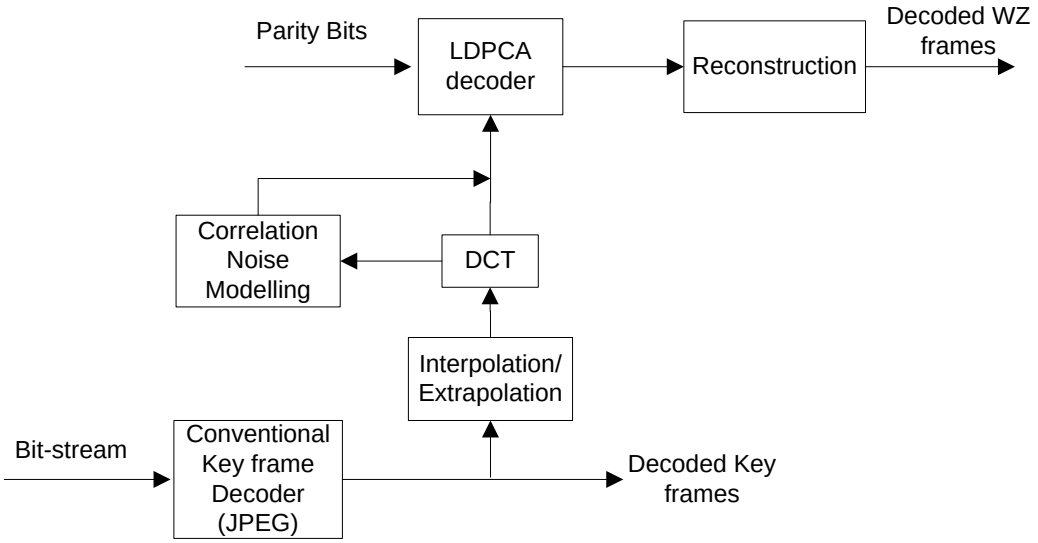


Figure 6.3 Decoder Functional Block Diagram

The DVC decoder main functional blocks are shown in Figure 6.3. The JPEG decoding consists of doing all the above JPEG encoding process in reverse. One or more already reconstructed frames (either WZ frames or key frames) will serve as side information for the WZ decoder. The correlation between the key frames and the WZ frames is modelled by Laplacian distribution. The WZ decoder receives successive

chunks of parity bits from the encoder following the requests made through a feedback channel.

If the parity bits are not sufficient to successfully decode a certain DCT coefficient, the decoder requests more bits from the encoder.

6.2.4 System Design Flow

The DM6437 EVM is a development platform that enables fast applications evaluation and development for the TI DaVinci™ processor family [79]. The block diagram of the internal architecture of the DM6437 EVM is shown in Figure 6.4.

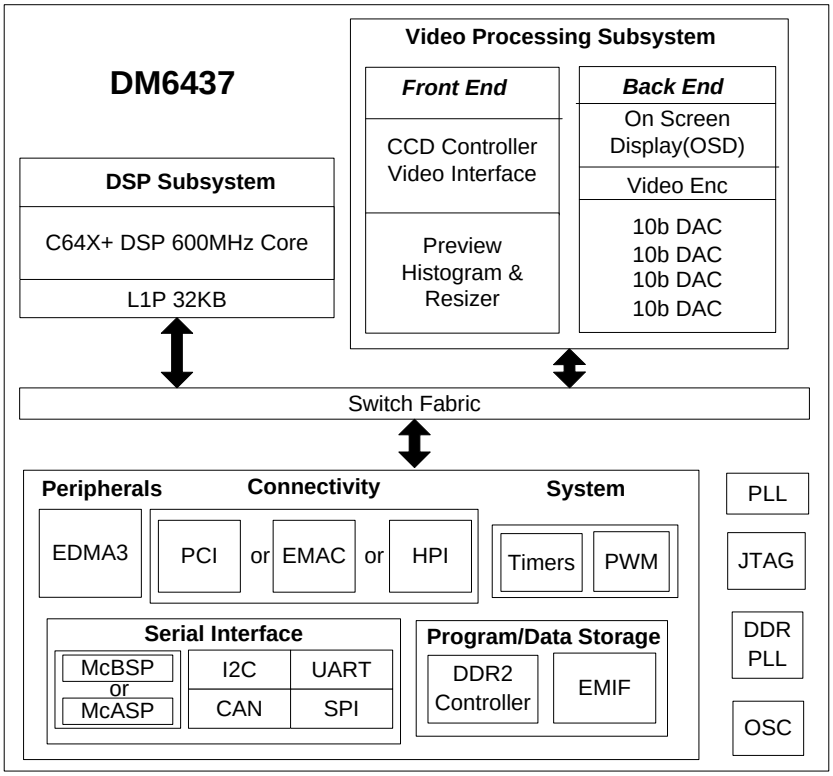


Figure 6.4 DM6437 EVM Architecture

Its powerful CPU allows efficient hardware pipelining under certain conditions. It is able to dispatch up to eight parallel instructions in each CPU cycle. The big Level 1 cache (L1), configurable Level 2 memory (L2), Internal DMA (IDMA) plus EDMA 3.0 enable fast data transfers with external device or memory to offload CPU. Moreover,

the Switched Central Resource (SCR) can route up to four transfers between CPU, EDMA, device peripherals and memory at the same time.

A rich set of available libraries and tools for DM6437, including debugging, development and performance analysis not only shorten the development cycles, but also improve the efficiency and robustness of the target applications. Other features can be found in [80].

6.2.5 Encoder Implementation

The implementation of the encoder is optimized at different levels by different techniques. A set of optimized image/video processing libraries such as IMGLIB [81], provided by Texas Instruments (TI), are used. These libraries provide high performance codes and an efficient and robust way of development. TI also provides high performance code generation tools to aid developers to debug and optimize codes. Code Composer Studio (CCS) is used throughout our implementation.

6.2.6 Discrete Cosine Transform

The DCT component of the WZ encoder is implemented by DSP Image/Video Processing Library with all operations in image blocks performed entirely in place. The DCT transform algorithm presented in [82] is efficiently implemented in the library. The number of operations is less than 1/6 of the conventional DCT algorithm using a 2-sided (Fast Fourier Transform) FFT. At the programming level, techniques like instruction scheduling and pipelines, registers reuse, etc. are employed to improve instruction-level parallelism.

6.2.7 Coefficients to Bit-stream

The DCT transformed coefficients have to be converted to a bit stream for LDPCA encoding. We first combine the two layer loops of the horizontal and vertical access of each coefficient. Second, a coefficient register is used to hold the coefficient value for each bit, avoiding re-calculation of array index. Also, the inner loop calculating the bit value of each bit-plane is unrolled to facilitate software pipeline. Finally, the modulo-2 operation is replaced by a faster bitwise and operation.

6.2.8 LDPCA Encoding

To facilitate adaptive coding rate, LDPCA codes are used. LDPCA codes can be represented by a sparse matrix. When dealing with image data, the size of this sparse matrix can be huge. Thus, most PC-based implementations of LDPC coding store the matrix in external memory [19][35][21][83]. A LDPCA code typically contains a set of codes with various code rates, range from the lowest code rate to the highest code rate, so that the decoder can start decoding with a lower rate bound, gradually increasing the rate if the parity is not sufficient. However, the code rate can be pre-estimated. Hence, it is not necessary to load the codes with all the available code rates. But even if large amount of unnecessary LDPCA codes are avoided from loading to the memory, the external memory reading, parsing, allocating and copying of a single LDPCA code are still time costing.

In our implementation, we compile the LDPCA code into the executable file, making a built-in LDPCA code in the data segment instead of storing it in external memory disk. Thus, high speed loading of LDPCA codes can be achieved by increasing a bit of program size.

The LDPCA encoder consists of an LDPC syndrome-former followed by an accumulator. The bits of the quantized DCT coefficients are first multiplied by the parity check matrix, yielding syndromes. These syndromes are in turn accumulated modulo (MOD) 2, producing the accumulated syndrome bits [21][22]. We denote the adjacent syndromes as $S^{(t-1)}$ and $S^{(t)}$, thus the accumulated syndrome bits $AS^{(t)}$ can be calculated as follow:

$$S^{(t)} = S^{(t-1)} + S^{(t)}$$
$$AS^{(t)} = S^{(t)}\ MOD\ 2$$

It can be noted that doing modulo 2 and accumulating at the same time does not affect the result, and the addition modulo 2 corresponds to bitwise exclusive or (XOR) operation, therefore the above calculations can be simplified as below in the implementation, where bitwise and with 1 (AND 1) is an alternative but faster operation for modulo-2.

$$AS^{(t)} = \left(S^{(t)}\ XOR\ S^{(t-1)}\right)\ AND\ 1$$

6.2.9 Key Frames Encoding

The key frames are encoded by JPEG using the Codec Engine API. The Codec Engine is an extendable and configurable framework that provides developers a common interface to access eXpressDSP-compliant codecs and algorithms [84].

6.2.10 Performance Study and Analysis

1. Complexity Performance

Our transform-domain DVC codec divides the video sequence into groups of pictures (GOP) with GOP size of 2. The odd frames are coded as key frames, decoded without reference to side information. The even frames are WZ coded, decoded using the previous reconstructed frames. A set of tests on both of the encoder and the decoder

has been carried out to verify their performance. Three video sequences, foreman, hall monitor and soccer are used, at QCIF resolution and 15 Hz frame rate. Only the luminance component is considered in the subsequent results.

A block size of 8 is used for the DCT and quantization. A scaled quantization matrix in Annex K of the JPEG standard [78] with scaling factor of 0.5 is used in the two encoders. The WZ frames are coded by a regular degree-3 LDPCA code of length 50688 bits [21]. Each QCIF-sized WZ frame is divided into four regions with equal dimension to match the code length. Therefore, the test results on main functional blocks of the WZ encoder are given for ¼ of a frame. The parity bits are generated by DM6437 and stored in the PC side through the embedded JTAG emulator. After an iterative decoding using Message Passing Algorithm [85], if the decoded bit-stream does not satisfy the syndrome check, the decoder requests additional parity bits from the encoder via a feedback channel.

Table 6.1 Implementation Performance of The DM6437 Based WZ Encoder

AVERAGE NUMBER OF INSTRUCTION CYCLES(CYCLES×10^6)			
	Foreman	**Hall Monitor**	**Soccer**
	Non-optimized/Optimized		
DCT&Quantization (1/4 frame)	415.4/3.9	415.4/3.9	415.4/3.9
Coefficients to Bit-stream (1/4 frame)	2.2/1.8	2.2/1.8	2.2/1.8
LDPCA Encoding (1/4 frame)	564.4/27.7	562.4/27.8	562.1/27.6
WZ Encoder (whole frame)	3960.1 /134.2	3959.9/134.0	3959.7/134.6
Overall Improvement with Optimization	29.5 times	29.6 times	29.4 times

We compared a non-optimized and an optimized WZ encoder on the DSP. The non-optimized DCT process uses a 2-sided FFT algorithm which works with floating point data results in slower processing at the algorithm level. In addition, it allocates a

temporary memory to store the transformed coefficients for each 8-by-8 block. This frequent memory allocation and release operation is also time-costing. The optimized DCT process uses a faster in place DCT transform algorithm which can work with fixed point data with only a little loss in accuracy. The optimized implementation not only speeds up the transform process but saves the memory usage.

The non-optimized conversion of DCT coefficient to bit-stream accesses each coefficient by two layer loops. The outer loop indexes the rows of a frame whereas the inner loop indexes the column. These two loops are combined into a single loop in the optimized implementation. Furthermore, loop unrolling technique is employed on bitwise access to help software pipelining.

In LDPCA encoding process, we avoid loading the large LDPCA codes from the external memory by compiling the codes into the executable file. Therefore, the codes are integrated in the binary program and are loaded to the DSP memory together with the program resulting in high speed loading of LDPCA codes. This approach increases the program size by around 728KB which is negligible for the huge total memory capacity, but eliminates the time for string parsing, external memory access as well as dynamic memory allocation. Furthermore, the syndrome accumulator is simplified by doing accumulating and modulo-2 at the same time with bitwise operations.

Table 6.1shows the profiling results of the DM6437 based encoder, given in average number of instruction cycles. The performance is tested on the WZ encoder and its main functional blocks (DCT and quantization, coefficients to bit-stream, LDPCA encoding). The last row shows the improvements achieved with the optimized implementation. An overall reduction of more than 29 times of WZ encoder complexity in terms of average number of instruction cycles is obtained. The average number of instruction cycles

reflects the actual amount of operations in the algorithms. Therefore, the reduction on the instruction cycles has a great impact on practical power consumptions. For resource restricted applications, this means less processing power is needed or longer battery life can be achieved.

2. Rate-Distortion Performance

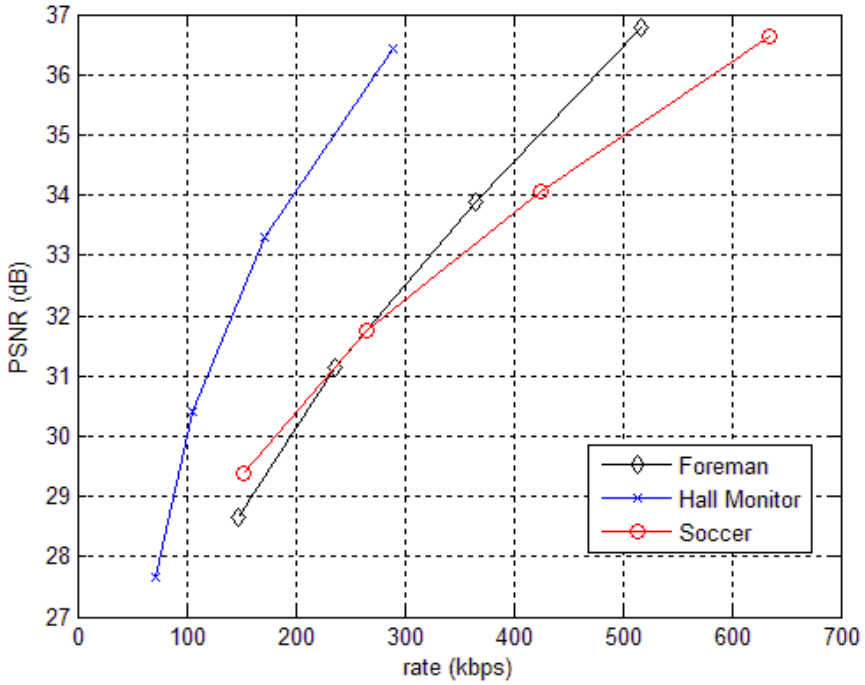


Figure 6.5 RD Curves for DSP-PC based DVC codec implementation for different sequences

We also verified the rate-distortion (RD) performance of this DSP-PC based DVC architecture. Our experiments use 50 frames of the above video sequences. The quantization matrix in the tests is scaled by factor Q = 0.5, 1, 2 and 4, respectively. Figure 6.5 compares the RD curves for Foreman, Hall Monitor, and Soccer, which represent different motion speed. The results are given in average rate and PSNR values. It can be seen from the figure that sequence with faster motion (Soccer) are often inferior in RD performance due to more "errors" between side information and WZ frames introduced by motion. It should also be noted that the results are for fixed point

implementation which explains why it is slightly lower than the expected using floating point.

Further reduction in computational cost can be achieved by employing Q-format representation in the quantizer to facilitate the rounding and truncation process, and refining the C/C++ code with C64x intrinsic or linear assembly to fully exploit the potential of the target platform. Furthermore, for a regular degree LDPCA code, further memory reduction can be achieved utilizing the code-word index structure. In addition, a better alternative for key frames coding such as H.264/AVC Intra codec can be used to improve the quality of side information and thus improve the final RD performance.

6.3 PC-HPC DVC Parallel Implementation

Due to high complexity of DVC decoder, it is supposed to be implemented on high performance base station. We have implemented a DVC codec in a PC-HPC (High Performance Cluster) architecture which simulate the scenario of running a DVC decoder at a base station. This also compliments our encoder implementation work to provide a comprehensive suggestion and full evaluation of the state-of-the-art DVC codec implementation under a practical software and hardware setup. Since this chapter mainly focuses on implementation aspects, SI refinement is therefore not taking into consideration, but readers who are interested in this topic can refer to Chapter 4 for more details.

6.3.1 System Overview

The system architecture of our PC-HPC DVC codec is shown in Figure 6.6. Video frames are split to key frames and WZ frames. The number of WZ frames is decided by GOP size. Key frames are encoded by H.264/AVC Intra encoder and the resulting bit-stream is stored along with all the relevant encoding parameters. WZ frames are divided

into 4×4 blocks and each block is DCT transformed and uniformly quantized. The quantized symbols are then split into bit-planes in zigzag order, starting from the most significant bit-planes (MSB) to the least significant bit-planes (LSB). A CRC code is generated for each bit-plane before they are encoded by LDPCA codes. LDPCA encoded bits, also referred to as syndrome bits, are stored into a file with associated coding parameters. For simplicity reason, this file is separately stored from the intra coded key frame files.

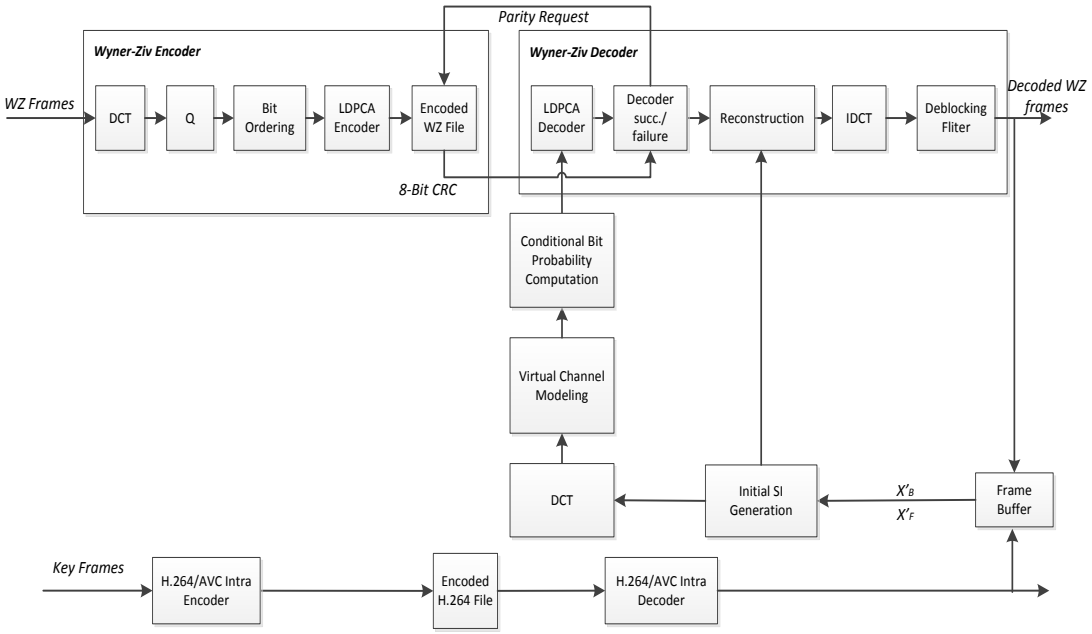


Figure 6.6 System Architecture

At the decoder side, decoded key frames are stored in buffer to provide information for initial SI generation. Since our DVC codec is working in transform domain, the initial SI has to be DCT transformed. Any decoded WZ frames are also stored into the same buffer. These frames are usually called reference frames and their residue frame computed during initial SI generation will be utilized to estimate the correlation noise. The noise is assumed to follow Laplacian distribution. The resulting Laplacian parameters are then used to compute conditional bit probabilities which will be used for LDPCA decoding. The accumulated syndrome bits are decoded using an iterative

message passing algorithm, which converges when the syndrome checks are fulfilled or the maximum number of iterations is reached. It requests for more parity bits if the current parity bits are not sufficient and a successful decoding is always tested by the CRC check. Decoded bit-streams are combined to form the quantized symbols, optimally reconstructed and inverse DCT transformed. Decoded frames are always smoothed by a deblocking filter to reduce block artifacts. Finally, decoded key frames and WZ frames are re-ordered in the same sequence as the input video sequence.

6.3.2 Encoder Implementation

Although the target platform of our encoder implementation is in a general purpose PC, it can be easily ported to digital signal processors, mobile phones or other resource critical devices. The encoding flow chart is depicted in Figure 6.7. It can be seen that the encoding process is a recursive loop since the encoding starts from the middle frame of a GOP, then move the right and left boundaries to the middle and repeat this process until all the frames in a GOP are encoded.

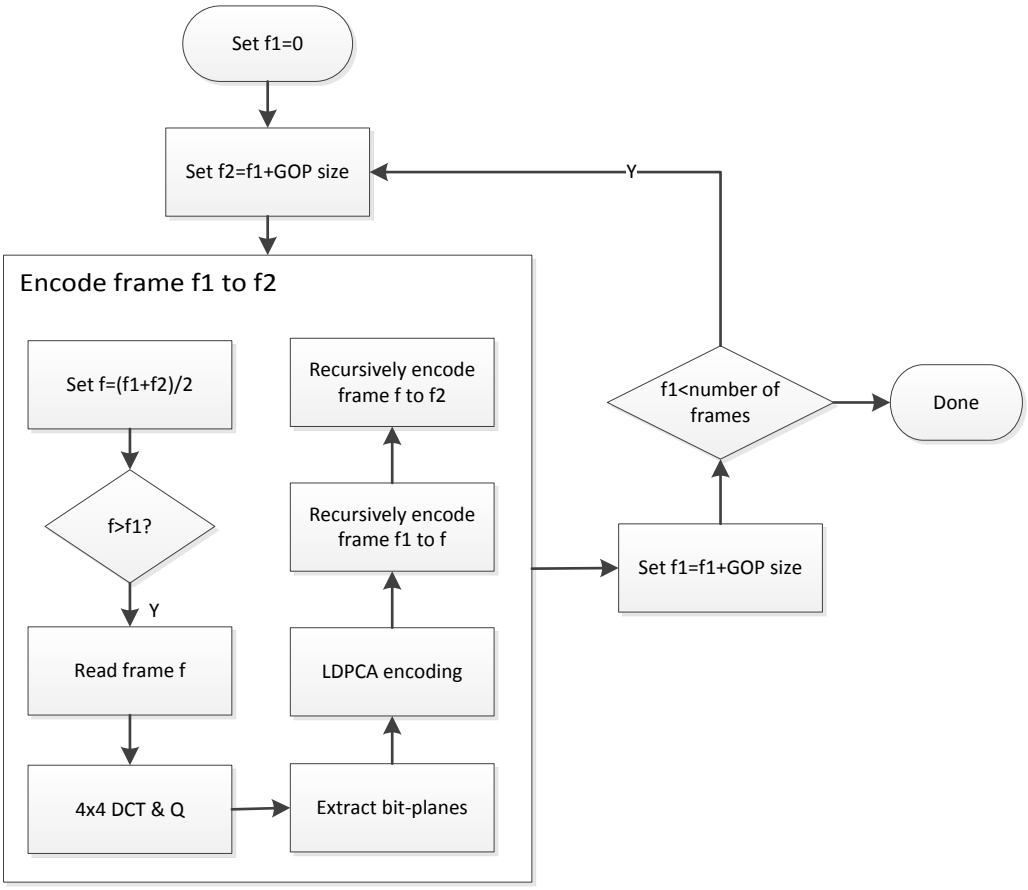


Figure 6.7 WZ Encoder Flow Chart

Key aspects of encoder implementation include quantizer design, LDPCA encoder implementation and file structure organization, which will be discussed in the following sections.

6.3.3 Quantizer Implementation

The quantizer is one of the key components that determines the quality of decoded frames as well as the decoding speed. Since DVC aims at achieving low complexity encoding, the quantizer design is usually simple and straightforward. Typical quantizer used in literatures is the uniform scalar quantizer, sometimes with an additional dead-zone to achieve better compression rate. Due to complexity reason, a uniform quantizer without a dead-zone is used in the implementation.

Our encoder quantizer is based on the quantizer from DISCOVER codec but without a dead-zone. The dynamic range for DC coefficients is assumed to be $[0, 2^{11})$, although the actual range should be $[0, 2^{10})$. A bigger value range with the same number of quantization levels means bigger step size and hence higher compression rate (at the sacrifice of image quality). The dynamic range for AC coefficients is assumed to be $[-Max_{AC}, Max_{AC})$, where Max_{AC} is the absolute maximum value of all the coefficients in a band. Each Max_{AC} has to be stored along with the compressed bit-stream.

The quantization step size Δ for DC coefficients is calculated as below, where *L* is the number of quantization levels decided by one of the quantization tables defined in Figure 6.8.

$$\Delta = \frac{2^{11}}{L} \tag{6.1}$$

The selection of the above quantization table is decided by the target rate/distortion requirement. And zero quantization level means these bands are not encoded into bit-stream and will be recovered from corresponding bands in SI directly.

Calculation of the step sizes for AC coefficients is,

$$\Delta = \begin{cases} \left\lceil \frac{(Max_{AC} + 1) \times 2}{L} \right\rceil & if\ L > 0 \\ 0 & else \end{cases} \tag{6.2}$$

where $\lceil x \rceil$ means rounding to the smallest integral value that is not less than x. For zero quantization level, the step size is marked as zero as well and these bands will be ignored in quantization.

16	8	0	0
8	0	0	0
0	0	0	0
0	0	0	0

Q_1

32	8	0	0
8	0	0	0
0	0	0	0
0	0	0	0

Q_2

32	8	4	0
8	4	0	0
4	0	0	0
0	0	0	0

Q_3

32	16	8	4
16	8	4	0
8	4	0	0
4	0	0	0

Q_4

32	16	8	4
16	8	4	4
8	4	4	0
4	4	0	0

Q_5

64	16	8	8
16	8	8	4
8	8	4	4
8	4	4	0

Q_6

64	32	16	8
32	16	8	4
16	8	4	4
8	4	4	0

Q_7

128	64	32	16
64	32	16	8
32	16	8	4
16	8	4	0

Q_8

Figure 6.8 Eight quantization tables for eight RD points

Therefore, DCT coefficients can be quantized to $\frac{c}{\Delta}$ where c is a DCT coefficient. However, AC coefficients can contain negative numbers. The quantized symbols q are usually shifted to all positive levels, since only positive numbers are usually supported for array indexing and this also simplify the coding process as the sign bits are not necessary any more. The level shift z for an AC coefficient is computed as,

$$z = \frac{L-2}{2} \tag{6.3}$$

6.3.4 LDPCA Encoder

The LDPCA encoder generates syndrome bits for each bit-plane of quantized symbols. This process has been described in 6.2.8, but the syndrome bits are converted to bytes to save storage space.

6.3.5 File Structure Organization

Since there is no standard for the file structure of WZ bit-stream so far, we propose a simple file structure to store the bit-stream and all the relevant parameters. The file structure for encoded WZ frames is depicted in Table 6.2. Encoded frame data are organized according to the order of GOP coding, i.e. the middle frame is always encoded first.

Table 6.2 File structure for encoded WZ frames

Content	Number of bytes
Frame resolution (0:QCIF, 1:CIF)	1
Quantization table index	1
GOP size: 2, 4 or 8	1
Total number of frames encoded	2
Encoded frame data for frame i	-
……	-
Encoded frame data for frame N	-

For each frame, the AC ranges for all the AC bands are stored at the beginning of the encoded frame data, in zigzag scan order, followed by all the encoded bit-plane data in the same order, depicted in Table 6.3. For each encoded bit-plane data, CRC code is always placed ahead of the syndrome bits to help checking if decoding is successful, as shown in

Table 6.4.

Table 6.3 Data layout of encoded frame data for a single frame

Content	Number of bytes
AC range for the first AC band	2
AC range for the second AC band	2
……	-
AC range for the n^{th} AC band	2
Encoded bit-plane data for bit-plane 1	-
Encoded bit-plane data for bit-plane 2	-
……	-
Encoded bit-plane data for bit-plane k	-

Table 6.4 Data layout of encoded bit-plane data for a single bit-plane

Content	Number of bytes
CRC code	1
Syndrome bit-stream	-

6.3.6 Decoder Implementation

A transform domain DVC decoder without SI refinement process is implemented using OpenMP parallel technique. Details of parallel implementation of some selected modules which have significant impact on the decoding complexity are presented below.

6.3.7 Initial Side Information Generation

The initial SI is generated using the method proposed in [30], the implementation of this algorithm is briefly described below, but more details on parallel implementation are discussed here.

The two reference frames, the past and the future reference frames, are first filtered by a 3×3 mean filter such that the motion vectors generated will be more reliable, and then they are both up sampled through a finite impulse response (FIR) filter for half pixel motion estimation.

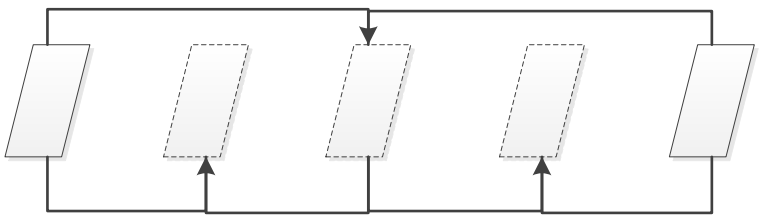
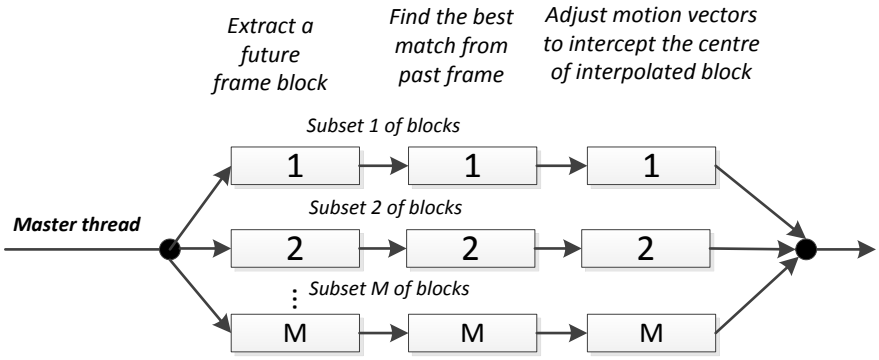


Figure 6.9 Order of SI generation for GOP=4

The initial motion vector for each macro block of size 16×16 is selected by forward motion estimation. However, if the motion vectors intercept the centre of the interpolated blocks, some areas in the interpolated frame may not be filled when motion compensation is performed. A solution proposed in [30] suggests that for each non-

overlapped block in the interpolated frame, the motion vector that has intercepting point closest to the centre of the block will be selected. This approach guarantees that each non-overlapped block in the interpolated frame can be assigned a motion vector and therefore can effectively eliminate holes or blank area in the interpolated frame. The motion vectors obtained in this step are further refined by half pixel bi-directional motion estimations, first using block size 16×16 and then refined again using the size of 8×8. In order to overcome the spatial incoherence of the motion vectors, weighted vector median filters are used to reduce the number of false estimations. It selects the vectors that minimize the sum of weighted distances to all the other neighbouring vectors. The weights are computed according to the ratio of the mean square error of the current block and the neighbouring block. Finally, the initial SI is generated by bi-directional motion compensation using the motion vectors obtained so far.

Due to the large searching range in forward motion estimation, it is the most time costing module in the initial SI generation process. Therefore, this module is implemented in parallel. For each block in the future frame, it compares with all the blocks within the searching range in the past frame to find the best matched block using sum of absolute difference (SAD) as the cost function. Each block can then be assigned a forward motion vector and a backward motion vector, derived from half of the motion vector between the current block and its best matched block. The motion vectors are then re-assigned according to the distance of the intercepting points and the centre of the blocks. Since finding the motion vectors for each block is independent from each other, therefore blocks can be divided evenly among threads and forward motion estimation can be carried out in parallel. The corresponding flowchart and the pseudo-code are depicted in Figure 6.10.



(a) Flow chart

```
#pragma omp parallel for
for each 16×16 block {
    Extract a future frame block;
    Find the best match from past frame;
    Adjust motion vectors;
}
```

(b) C++ pseudo code using OpenMP

Figure 6.10 Initial SI Generation

6.3.8 Correlation Noise Modelling

The correlation noise distribution is modelled using the widely adopted coefficient level method proposed in [4]. For all the coefficients in each band that needs to be decoded, the noise distribution of WZ frame X and SI frame Y over each possible coefficient level at pixel (x, y) is calculated.

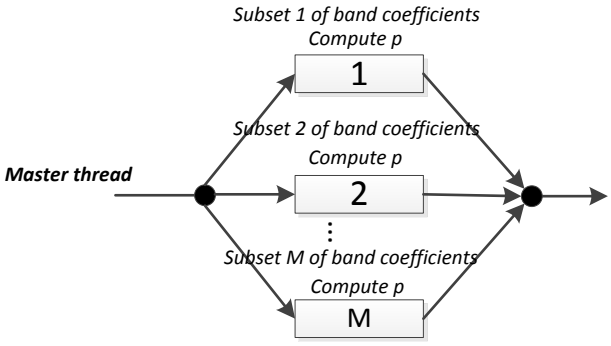
$$p(X(x,y) - Y(x,y)) = e^{-\alpha(x,y)|X(x,y)-Y(x,y)|} \tag{6.4}$$

where α is the Laplacian parameter.

In most literature, the above calculation is carried out using the exact form of the probability density function of Laplacian distribution, which multiplies $\frac{\alpha(x,y)}{2}$ to the right of (6.4). However, in practical implementation this means the resulting probabilities have to be normalized before use. It can be noted that Laplacian distribution is an exponential distribution and equation (6.4) is already an exponential

distribution which can give very close shape of the actual Laplacian distribution. Furthermore, the modeling process itself is only an estimation of the actual distribution and the simpler form of (6.4) saves more computations. It avoids the calculation of multiplication for each possible level of a WZ coefficient and more importantly, it does not need normalization process any more since the resulting probabilities are always in the range of (0,1].

Since the calculation of the noise distribution for each coefficient does not rely on any other coefficients, the coefficients in a band can be divided into M subset to run each subset in parallel, where M is decided according to the number of CPU cores. The flowchart and the pseudo code of parallel implementation of this module are depicted below.



(a) Flow chart

```
#pragma omp parallel for
for each coefficient in a band {
    compute p using equation (6.4);
}
```

(b) C++ pseudo code using OpenMP

Figure 6.11 Correlation noise modeling

6.3.9 Conditional Bit Probability Calculation

This module provides the probability of a bit of a WZ coefficient given the SI. In another word, it computes the belief of a bit of a WZ coefficient being 1 or 0 given the SI, which will be used later for belief propagation algorithm in LDPCA decoding.

A. Compute the range of quantization index

The lower bound L and the upper bound U of the b^{th} bit-plane of quantization index q being 0 and 1 given all of the decoded bit-planes can be found in (6.5).

$$\begin{aligned} q_L^b(0) &= Q^b \\ q_U^b(0) &= q_L^b(0) + 2^{b-1} - 1 \\ q_L^b(1) &= Q^b + 2^{b-1} \\ q_U^b(1) &= q_L^b(1) + 2^{b-1} - 1 \end{aligned} \tag{6.5}$$

where $b \in [B, 0)$, B is the number of total bits for current decoding band and Q^b is the decoded quantization values before bit-plane b.

B. Compute the range of coefficient values:

Since the decoder works in non-quantized transform domain, the above range has to be converted to the range of coefficient values. Similar to the DISCOVER codec [19] (but with no dead-zone used in our implementation), the q^{th} quantization interval I_k^q for band k is defined as below:

$$I_k^q = \begin{cases} [(q-1)W_k, qW_k) & q \le 0 \\ [qW_k, (q+1)W_k) & else \end{cases} \tag{6.6}$$

where W_k is the quantization step size for band k. This quantization interval is also the range for the coefficient values. Substitute q in (6.6) with q_L^b and q_U^b in (6.5) we can get the range $[C_L^b, C_U^b]$ for the coefficient values.

$$
\begin{aligned}
C_L^b(0) &= \begin{cases} \left(q_L^b(0)-1\right)W_k & q_L^b(0) \le 0 \\ q_L^b(0)W_k & else \end{cases} \\
C_U^b(0) &= \begin{cases} q_U^b(0)W_k - 1 & q_U^b(0) \le 0 \\ \left(q_U^b(0)+1\right)W_k - 1 & else \end{cases} \\
C_L^b(1) &= \begin{cases} \left(q_L^b(1)-1\right)W_k & q_L^b(1) \le 0 \\ q_L^b(1)W_k & else \end{cases} \\
C_U^b(1) &= \begin{cases} q_U^b(1)W_k - 1 & q_U^b(1) \le 0 \\ \left(q_U^b(1)+1\right)W_k - 1 & else \end{cases}
\end{aligned} \tag{6.7}
$$

C. Compute conditional probability:

With all the possible levels of a WZ coefficient in the defined range above, the conditional probability of a bit of WZ coefficient being 0 and 1, denoted as p_0 and p_1, respectively, can be derived as below,

$$
\begin{aligned}
p_0 &= \sum_{x=C_L^b(0)}^{C_U^b(0)} e^{-\alpha|x-y|} \\
p_1 &= \sum_{x=C_L^b(1)}^{C_U^b(1)} e^{-\alpha|x-y|}
\end{aligned} \tag{6.8}
$$

where y is the SI at the same pixel location.

D. Normalization:

However, the probabilities calculated in (6.8) should not exceed 1. To guarantee the resulting probabilities are within a valid range, p_0 and p_1 are usually normalized before being used.

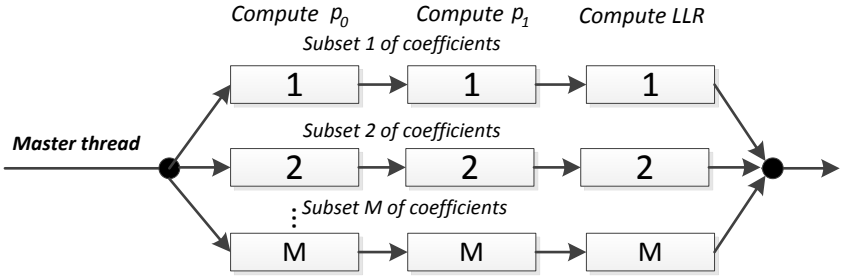
$$
p_0 = \frac{p_0}{p_0 + p_1} \tag{6.9}
$$

$$p_1 = 1 - p_0$$

The conditional probability in terms of likelihood ratio can therefore be computed as $\frac{p_0}{p_1}$, since logarithm gives more numerically stable results, log likelihood ratios LLR $\log(\frac{p_0}{p_1})$ are usually used instead of likelihood ratio.

For AC coefficients, the above algorithm has to be adjusted to consider the level shifts applied at the encoder side.

It can be noticed that the calculation of conditional probability for each coefficient does not require any information of other coefficient. Therefore, for a DCT band under decoding, all the coefficients within that band can be divided into sub-groups for parallel processing. The corresponding flow chart and C++ pseudo code using OpenMP is depicted below.



(a) Flow chart

```
#pragma omp parallel for
for each coefficient in a band {
    compute quantization range using equation (6.5);
    compute coefficient range using equation (6.7);
    compute p0 and p1 using equation (6.8);
    normalize p0 and p1using equation (6.9);
    compute LLR log(p0/p1);
}
```

(b) C++ pseudo code using OpenMP

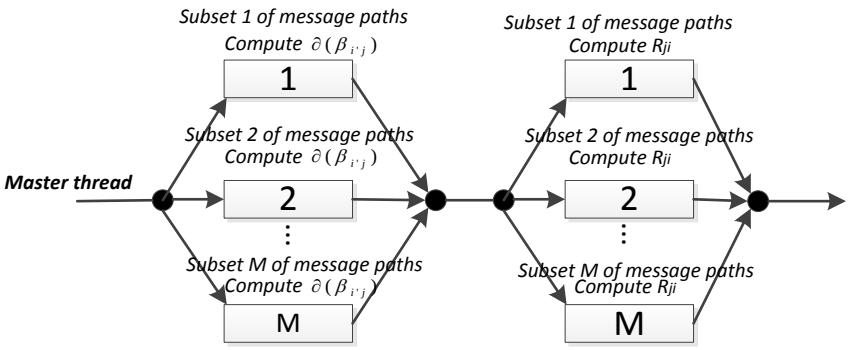
Figure 6.12 Conditional bit probability calculation

6.3.10 LDPCA Decoding

LDPCA decoding is the most time consuming components at the decoder side according to our experimental results. The decoding algorithm adopted here is derived from the belief propagation algorithm explained in [86], in which the computation of the message R_{ji} from each check node j to its associated variable nodes i, depicted in (6.10), has the highest computational complexity since the calculation involves summation of logarithm and hyperbolic tangent.

$$R_{ji} = \prod_{i' \in V_{j \setminus i}} \alpha_{i'j} \cdot \partial\left(\sum_{i' \in V_{j \setminus i}} \partial(\beta_{i'j}) \right) \tag{6.10}$$

where $V_{j \setminus i}$ denotes all variable nodes connected to check node j except node i, $\alpha_{i'j}$ and $\beta_{i'j}$ are the sign and the magnitude of the extrinsic message from each variable node to its associated check nodes, respectively, and $\partial(x) = -\log(\tanh(\frac{x}{2}))$. (6.10) can be computed in two steps, first loop over all the message paths to compute $\partial(\beta_{i'j})$, then sum them up and loop over all the message paths again to get R_{ji}. The computations involved in each message path is independent to any other paths, these two loops can therefore be computed in parallel. The corresponding flow chart and C++ pseudo code for this implementation are depicted below.



(a) Flow chart

```
#pragma omp parallel for
for each message path {
    compute ∂(β_{i,j});
}
#pragma omp parallel for
for each message path {
    compute R_{ji} using (6.10);
}
```

(b) C++ pseudo code using OpenMP

Figure 6.13 Belief propagation

In order to further improve the decoding speed, a minimum rate estimation algorithm proposed in [87] is used to reduce the number of parity bit requests. This rate is computed based on the conditional entropy $H(X|Y)$ between the original data and the SI. The LDPCA decoding always starts from the estimated minimum rate, if it fails the syndrome check or the CRC check, it then requests more parity bits from the encoder buffer until both the syndrome check and the CRC check are successful.

6.3.11 Performance Study and Analysis

A. Test Condition

The proposed parallel implementation of our transform domain DVC codec is evaluated and compared with the sequential implementation in terms of the complexity performance and the RD performance using the following test conditions.

1) Video sequences: Foreman, Hall Monitor, Coastguard, and Soccer.
2) Number of frames: all frames of the test sequences have been used to evaluate the RD performance which means 150 for Foreman, 165 for Hall Monitor, 150 for Coastguard, and 150 for Soccer. However, since the average complexity performance only shows negligible differences using various numbers of frames, we only use the first 30 frames of all test sequences to evaluate the decoding complexity.

3) Spatial and temporal resolution: QCIF at 15 Hz which means 7.5 Hz for the WZ frames when GOP=2.
4) GOP length: 2, 4 and 8.
5) Eight RD points are considered for DVC codec, corresponding to eight 4×4 quantization matrices widely used in literature [19][37][39].
6) Key frames are coded by H.264/AVC Intra with constant quantization parameters as defined in [19].
7) Software and hardware configuration: the decoding tasks are performed on the Bright Beowulf Cluster Environment using 24 CPU processors of the running node under Linux operating system. The codec is written in C/C++ code and compiled by GCC 4.7.0 using OpenMP 3.1.

B. Complexity Performance

The CRG DVC codec is implemented in parallel as well as in serial, where the serial implementation can be seen as the parallel version that uses only one CPU core. The total decoding time of the parallel implementation and the serial implementation for all the sequences and GOP sizes are compared in this section. More detailed computational complexity on major decoding components for both parallel and serial implementations are also presented here.

1) Parallel vs. Serial Implementation

Table 6.5 to Table 6.7 show the total decoding time of parallel and serial implementation of CRG DVC for GOP size 2, 4 and 8, respectively. Please note that in the following implementations, no network channel is used for data transmission between the encoder and the decoder, i.e. video data is encoded and decoded locally. However, for practical applications where encoded bit-streams are transmitted over a real network channel, the results shown in Table 6.5 to Table 6.7 should refer to latency

since they include the time that data is requested through the feedback channel from the source. As expected, the decoding time increases as bit-rate and GOP size increase since there are more DCT bands to be decoded under higher bit-rate and more WZ frames to be decoded for larger GOP sizes. It can be observed that the parallel implementation using 24 CPU cores spends about less than 1/10 of the total decoding time with regard to the serial implementation across most GOP settings, rates and sequences. The impact of video content on the total decoding time meets the common expectation, i.e. the more complex the video content is, more time requires to decode that sequence. The video sequence that contains highest motion content here is Soccer, which therefore takes longest time to decode. The same pattern can be seen from the slowest video content tested here. Hall Monitor costs the shortest time to decode which is only about 1/5 for lower bit-rates and ¼ for higher bit-rates compared to Soccer sequence under both parallel and serial implementations.

For real time applications, a frame rate of 10 to 15 fps is required to eliminate any flickering effects, which means 30 frames have to be decoded within 2 to 3 seconds. It can be found in Table 6.5 that decoding Hall Monitor sequence with GOP size 2 in parallel under the lowest quantization matrix using 24 CPU cores can be considered real time.

Table 6.5 Total decoding time (in seconds) for CRG-DVC parallel (P) implementation (using 24 CPU cores) vs. serial (S) implementation for GOP=2

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**P**	6.66	7.14	7.93	14.96	19.85	31.25	31.08	45.65
	S	68.51	71.49	87.04	165.85	201.97	297.47	329.12	468.44
Soccer	**P**	12.26	11.74	12.16	24.43	27.83	38.28	41.39	53.49
	S	120.46	110.78	127.52	244.87	272.49	390.06	408.41	548.84
Coastguard	**P**	3.36	4.29	4.79	9.98	10.48	19.02	27.77	54.00
	S	37.87	46.40	51.72	108.09	113.14	193.49	277.83	552.50
Hall	**P**	2.33	2.91	3.17	6.14	6.34	10.79	11.76	17.96
	S	28.52	32.13	35.97	67.34	71.41	104.30	133.57	186.06

Table 6.6 Total decoding time (in seconds) for CRG-DVC parallel (P) implementation (using 24 CPU cores) vs. serial (S) implementation for GOP=4

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**P**	13.70	15.35	19.17	33.89	39.37	59.02	70.82	99.62
	S	140.07	147.75	180.30	331.66	395.49	570.54	668.10	968.70
Soccer	**P**	21.63	19.78	26.10	44.57	49.08	70.78	74.51	108.54
	S	205.40	198.71	239.76	437.01	488.37	688.07	742.33	1007.72
Coastguard	**P**	5.82	7.08	7.50	18.83	18.15	33.87	48.90	99.42
	S	67.55	73.90	82.63	179.62	188.27	324.29	475.32	967.96
Hall	**P**	3.94	3.93	4.62	9.49	9.86	16.31	18.55	26.94
	S	42.04	45.45	50.85	101.12	103.77	158.36	193.16	277.00

Table 6.7 Total decoding time (in seconds) for CRG-DVC parallel (P) implementation (using 24 CPU cores) vs. serial (S) implementation for GOP=8

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**P**	22.30	24.92	30.47	60.22	62.84	85.88	101.05	153.96
	S	215.39	239.62	282.88	525.46	596.77	819.35	972.45	1382.66
Soccer	**P**	27.85	26.45	33.70	66.37	71.23	88.03	97.30	142.69
	S	265.93	259.11	312.78	591.52	657.21	872.05	942.64	1351.99
Coastguard	**P**	7.72	9.54	10.94	25.37	27.73	47.33	61.68	126.77
	S	86.08	101.77	119.37	248.53	270.39	451.60	621.62	1241.00
Hall	**P**	4.91	4.90	5.73	12.66	12.91	18.40	21.78	36.70
	S	52.61	55.59	61.25	127.55	129.72	190.02	228.36	343.79

2) Component based complexity analysis

Table 6.8 - Table 6.10 show the decoding time for major computational components using serial implementation. It can be seen that the time spend on each component is increasing as GOP sizes increase and the majority computational complexity is spent on SI creation, correlation noise modeling and LDPCA decoding. Among these components, LDPCA decoding is the most time costing module and the time spent on conditional bit probability calculation is trivial. For Foreman sequence coded with GOP size 2, LDPCA decoding costs about 88% of the total decoding time for Q1 and a significant 98% for Q8. For Soccer sequence coded with GOP size 2, LDPCA decoding takes 93% of the total decoding time for Q1 and 98% for Q8 as well. It can also be seen

that the time spent on SI creation is almost constant regardless of bit-rates since the complexity of motion estimation does not change according to the bit-rate. However, the time spent on correlation noise modeling and conditional bit probability computation are increasing with the increase of the bit-rates since there are more bands to be decoded under higher bit-rates. Comparing the results of GOP size 2 and 4, it can be seen that the time costing for each component is almost doubled. However, LDCPA decoding for GOP size 8 is only about 1.3 times faster than using GOP size 4.

Table 6.8 Decoding time (in seconds) for Serial CRG-DVC components: SI Creation (S), Correlation Noise Modeling (C), Conditional Bit Probability Computation (P) and LDPCA Decoding (L) for GOP=2

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**S**	3.73	3.75	3.79	3.78	3.75	3.73	3.76	3.77
	C	4.27	4.27	5.67	6.59	6.80	6.90	7.08	7.04
	P	0.30	0.32	0.48	0.59	0.55	0.64	0.63	0.59
	L	60.14	63.08	76.99	154.73	190.67	285.98	317.43	456.84
Soccer	**S**	3.70	3.75	3.76	3.74	3.71	3.77	3.71	3.73
	C	4.13	4.19	5.27	5.96	6.27	6.60	6.50	6.65
	P	0.29	0.30	0.37	0.50	0.47	0.67	0.53	0.66
	L	112.27	102.48	118.02	234.52	261.85	378.81	397.47	537.60
Coastguard	**S**	3.72	3.73	3.73	3.81	3.80	3.72	3.72	3.99
	C	4.31	4.32	5.77	6.85	7.26	7.24	7.41	7.75
	P	0.53	0.59	0.69	0.85	0.92	0.87	0.99	1.15
	L	29.23	37.69	41.43	96.42	100.97	181.44	265.49	539.41
Hall	**S**	3.84	3.70	3.70	3.72	3.88	3.73	3.93	3.73
	C	4.73	4.55	6.22	7.54	8.39	8.42	8.88	8.60
	P	0.55	0.53	0.72	0.90	0.99	0.96	1.48	1.41
	L	19.32	23.28	25.23	55.01	57.97	90.98	119.06	172.11

TABLE 6.9 Decoding time (in seconds) for Serial CRG-DVC components: SI Creation (S), Correlation Noise Modelling (C), Conditional Bit Probability Computation (P) and LDPCA Decoding (L) for GOP=4

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**S**	5.45	5.40	5.40	5.36	5.38	5.37	5.39	5.42
	C	6.36	6.37	7.94	9.20	9.74	10.20	10.12	10.36
	P	0.62	0.55	0.60	0.83	0.77	1.07	0.99	1.09
	L	127.53	135.32	166.21	316.02	379.30	553.57	651.28	951.51
Soccer	**S**	5.39	5.56	5.45	5.38	5.38	5.43	5.42	5.35
	C	6.09	6.24	7.66	8.70	9.23	9.65	9.71	9.59
	P	0.44	0.48	0.56	0.68	0.72	0.99	0.99	0.83
	L	193.39	186.33	225.94	422.02	472.75	671.68	725.90	991.68
Coastguard	**S**	5.90	5.32	5.36	5.39	5.34	5.38	5.34	5.43
	C	7.16	6.38	8.54	10.09	10.66	11.01	10.92	11.53
	P	1.01	0.89	1.03	1.15	1.58	1.30	1.29	1.67
	L	53.32	61.20	67.54	162.75	170.39	306.29	457.45	948.99
Hall	**S**	5.36	5.33	5.35	5.33	5.35	5.33	5.48	5.34
	C	6.91	6.86	9.40	11.18	12.08	12.47	13.11	12.71
	P	0.78	0.81	1.16	1.40	1.51	1.55	2.27	1.64
	L	28.89	32.35	34.80	82.97	84.56	138.70	171.96	257.00

Table 6.10 Decoding time (in seconds) for Serial CRG-DVC components: SI Creation (S), Correlation Noise Modelling (C), Conditional Bit Probability Computation (P) and LDPCA Decoding (L) for GOP=8

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**S**	6.35	6.42	6.32	6.32	6.35	6.33	6.33	6.32
	C	7.42	7.52	9.31	10.93	11.57	11.76	11.93	12.06
	P	0.61	0.78	0.71	0.84	0.93	0.95	1.20	1.05
	L	200.89	224.77	266.36	507.09	577.57	799.95	952.62	1362.87
Soccer	**S**	6.30	6.43	6.33	6.31	6.50	6.30	6.29	6.32
	C	7.11	7.29	8.87	10.24	11.04	11.29	11.20	11.43
	P	0.51	0.55	0.65	0.79	0.88	0.90	0.93	0.99
	L	251.90	244.72	296.76	573.91	638.45	853.20	923.86	1332.91
Coastguard	**S**	6.37	6.41	6.54	6.34	6.44	6.37	6.28	6.34
	C	7.74	7.75	10.60	11.95	13.06	13.22	13.08	13.61
	P	0.96	1.08	1.35	1.27	1.53	1.70	1.59	1.86
	L	70.87	86.41	100.68	228.70	249.01	429.92	600.30	1218.81
Hall	**S**	6.32	6.35	6.31	6.33	6.32	6.31	6.34	6.27
	C	8.20	8.17	11.20	13.50	14.39	15.00	15.07	15.11
	P	0.97	1.03	1.49	1.98	1.92	2.91	2.47	2.00
	L	36.99	39.91	42.07	105.46	106.78	165.40	204.08	320.04

Table 6.11 - Table 6.13 show the decoding time for the same major computational components using parallel implementation. Comparing with serial implementation,

parallel implementation using 24 CPU cores can achieve about 10 times faster for each component using all the quantization matrices and GOP sizes.

Table 6.11Decoding time (in seconds) for parallel CRG-DVC components (using 24 CPU cores): SI Creation (S), Correlation Noise Modeling (C), Conditional Bit Probability Computation (P) and LDPCA Decoding (L) for GOP=2

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**S**	0.29	0.28	0.28	0.28	0.28	0.28	0.28	0.29
	C	0.19	0.19	0.26	0.32	0.35	0.37	0.37	0.37
	P	0.01	0.01	0.02	0.02	0.03	0.03	0.03	0.04
	L	6.12	6.62	7.33	14.28	19.12	30.51	30.33	44.89
Soccer	**S**	0.29	0.30	0.29	0.29	0.29	0.28	0.28	0.29
	C	0.19	0.19	0.25	0.30	0.34	0.35	0.36	0.37
	P	0.01	0.01	0.02	0.02	0.03	0.03	0.03	0.03
	L	11.73	11.20	11.57	23.76	27.10	37.55	40.66	52.74
Coastguard	**S**	0.29	0.28	0.27	0.27	0.27	0.27	0.27	0.28
	C	0.20	0.20	0.27	0.34	0.38	0.40	0.40	0.41
	P	0.06	0.06	0.07	0.08	0.09	0.08	0.09	0.10
	L	2.77	3.71	4.12	9.22	9.67	18.20	26.94	53.15
Hall	**S**	0.28	0.28	0.28	0.28	0.27	0.27	0.28	0.28
	C	0.21	0.21	0.30	0.38	0.42	0.44	0.45	0.45
	P	0.03	0.03	0.04	0.05	0.05	0.05	0.06	0.06
	L	1.77	2.35	2.51	5.37	5.52	9.94	10.90	17.10

Table 6.12 Decoding time (in seconds) for parallel CRG-DVC components (using 24 CPU cores): SI Creation (S), Correlation Noise Modelling (C), Conditional Bit Probability Computation (P) and LDPCA Decoding (L) for GOP=4

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**S**	0.40	0.39	0.39	0.40	0.40	0.40	0.40	0.39
	C	0.29	0.29	0.38	0.48	0.52	0.55	0.56	0.56
	P	0.02	0.02	0.03	0.04	0.04	0.04	0.05	0.05
	L	12.93	14.59	18.30	32.89	38.31	57.93	69.72	98.53
Soccer	**S**	0.41	0.41	0.40	0.40	0.40	0.40	0.40	0.40
	C	0.28	0.28	0.37	0.45	0.50	0.53	0.54	0.54
	P	0.02	0.02	0.03	0.03	0.04	0.04	0.04	0.05
	L	20.87	19.02	25.23	43.60	48.04	69.70	73.43	107.46
Coastguard	**S**	0.39	0.39	0.38	0.38	0.39	0.39	0.39	0.40
	C	0.30	0.30	0.41	0.51	0.57	0.60	0.60	0.60
	P	0.09	0.09	0.10	0.11	0.11	0.12	0.12	0.12
	L	4.97	6.24	6.54	17.73	16.97	32.66	47.69	98.21
Hall	**S**	0.38	0.38	0.38	0.39	0.39	0.38	0.39	0.38
	C	0.32	0.32	0.45	0.56	0.62	0.66	0.67	0.67
	P	0.04	0.04	0.06	0.08	0.08	0.09	0.09	0.09
	L	3.12	3.12	3.65	8.36	8.66	15.06	17.29	25.69

Table 6.13 Decoding time (in seconds) for parallel CRG-DVC components (using 24 CPU cores): SI Creation (S), Correlation Noise Modelling (C), Conditional Bit Probability Computation (P) and LDPCA Decoding (L) for GOP=8

		Q1	*Q2*	*Q3*	*Q4*	*Q5*	*Q6*	*Q7*	*Q8*
Foreman	**S**	0.46	0.46	0.46	0.46	0.47	0.46	0.46	0.46
	C	0.34	0.34	0.45	0.57	0.62	0.65	0.65	0.66
	P	0.03	0.03	0.03	0.04	0.05	0.05	0.06	0.06
	L	21.40	24.03	29.44	59.03	61.58	84.60	99.76	152.66
Soccer	**S**	0.48	0.48	0.48	0.48	0.48	0.47	0.47	0.47
	C	0.33	0.33	0.44	0.54	0.60	0.63	0.64	0.64
	P	0.02	0.02	0.03	0.04	0.05	0.05	0.05	0.06
	L	26.95	25.55	32.67	65.21	69.99	86.75	96.03	141.41
Coastguard	**S**	0.45	0.46	0.46	0.45	0.46	0.45	0.45	0.45
	C	0.36	0.35	0.49	0.61	0.68	0.71	0.71	0.72
	P	0.09	0.10	0.11	0.11	0.11	0.12	0.13	0.13
	L	6.74	8.54	9.79	24.08	26.35	45.91	60.27	125.36
Hall	**S**	0.45	0.45	0.46	0.45	0.46	0.45	0.45	0.45
	C	0.38	0.38	0.54	0.67	0.75	0.79	0.80	0.80
	P	0.05	0.05	0.08	0.09	0.10	0.10	0.11	0.12
	L	3.95	3.95	4.56	11.33	11.48	16.93	20.30	35.21

C. Rate-Distortion Performance

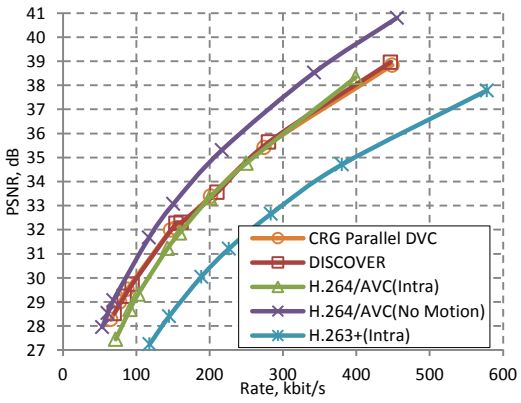
(1) CRG Parallel DVC vs. DISCOVER

The RD performance of the chosen coding solutions for all the selected video sequences is presented in Figure 6.14 and Figure 6.15. The results show that the CRG parallel DVC performs very similar to the DISCOVER codec for slow motion sequences using short GOP size. However, it dramatically outperforms DISCOVER for video sequences with high motion content, especially for higher bit-rate and longer GOP sizes. The RD gains are mainly brought by the better quality of SI. It is a big challenge for most DVC codecs to generate good SI under critical conditions that the motion is intense and GOP sizes are big.

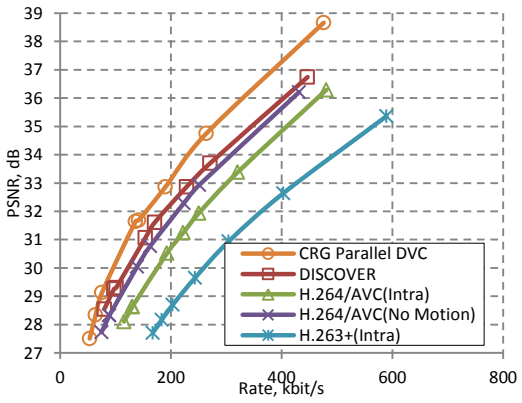
The Foreman and Soccer sequences are considered to be the high motion video contents, whereas Hall Monitor and Coastguard sequences are seen as relatively low motion video contents. As expected, the Foreman and Soccer sequences give better gains in RD performance, particularly the Foreman sequence that achieves the highest RD gains with regards to the DISCOVER codec. Taking a close look at Figure 6.14 for

the Foreman sequence, notably for the last RD point, there is approximately 1.6 dB for GOP size 8. Similar gains can be observed from the Soccer sequence with about 1.26 dB for GOP size 2.

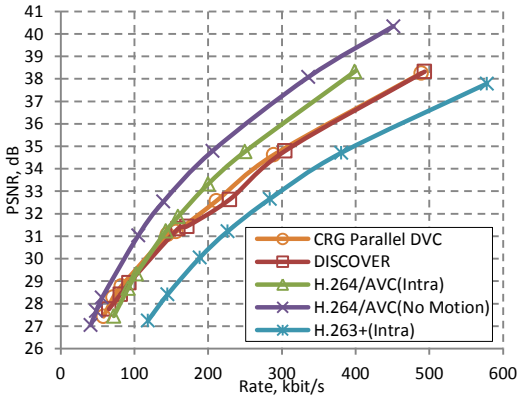
(a) Foreman: GOP=2



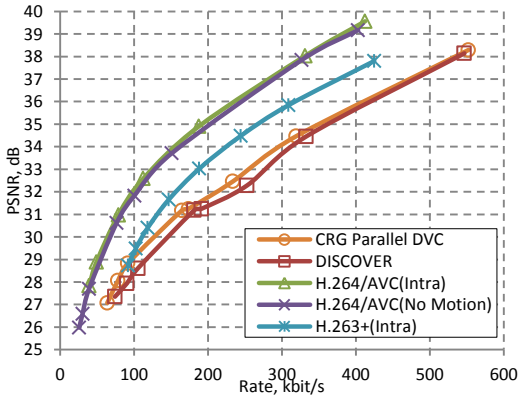
(b) Soccer: GOP=2



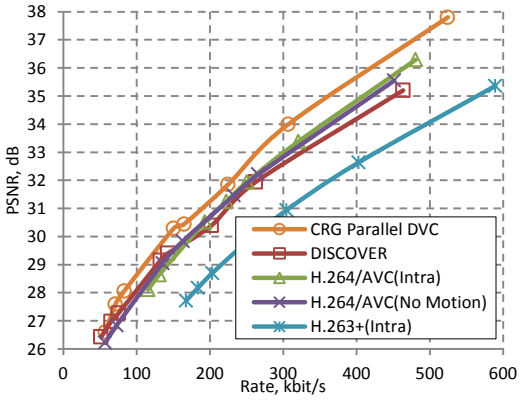
(c) Foreman: GOP=4



(d) Soccer: GOP=4



(e) Foreman: GOP=8



(f) Soccer: GOP=8

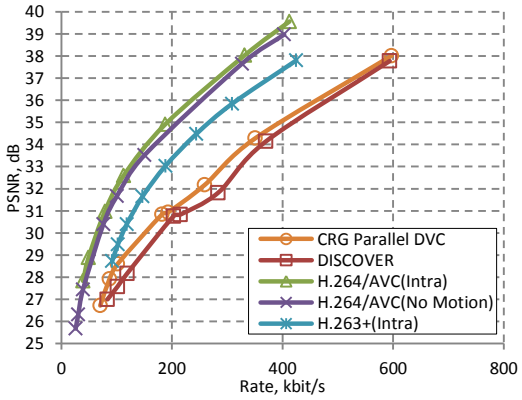
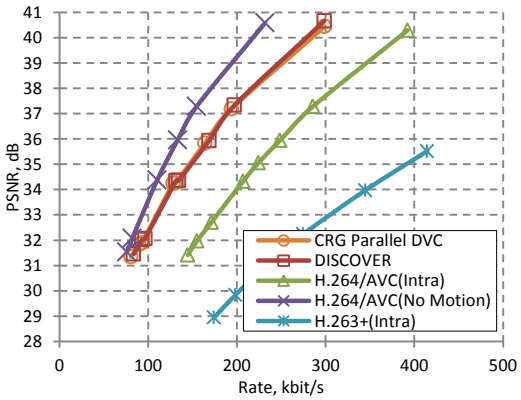
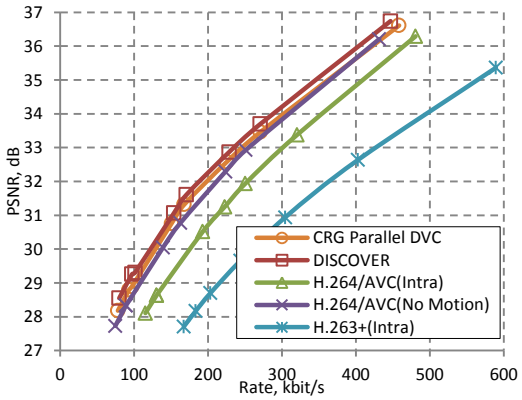


Figure 6.14 RD Performance for Foreman and Soccer sequences

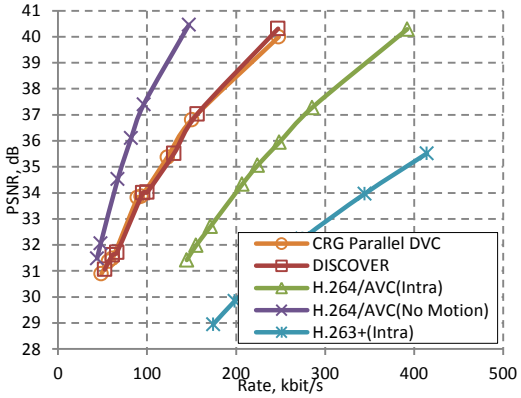
(a) Hall Monitor: GOP=2



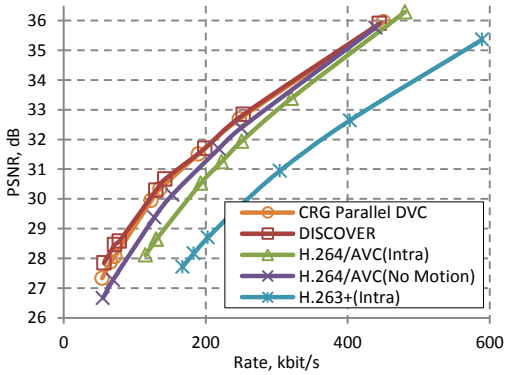
(b) Coastguard: GOP=2



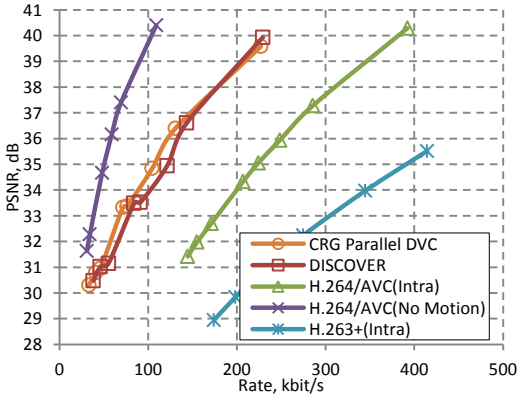
(c) Hall Monitor: GOP=4



(d) Coastguard: GOP=4



(e) Hall Monitor: GOP=8



(f) Coastguard: GOP=8

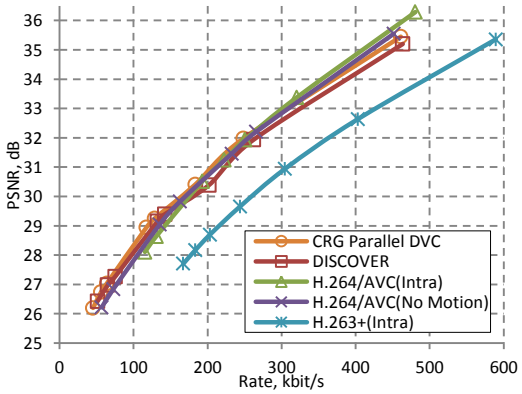


Figure 6.15 RD Performance for Hall Monitor and Coastguard sequences

However, the RD curves in Figure 6.15 show slightly lower performance for the sequences of Hall Monitor and Coastguard. The RD curve of the proposed codec overlaps DISCOVER's for the Hall Monitor sequence for GOP size 2 and 4, but a notable gain of up to 1 dB can be observed from the longest GOP size. Slight

performance loss of up to 0.3 dB can be seen from the Coastguard sequence using GOP size 2, whereas the RD performances for GOP size 4 are extremely close and some small gains of about 0.2 dB can be noted under high bit-rate for GOP size 8.

(2) CRG parallel DVC vs. Standard video coding solutions

The conventional video coding solutions evaluated here are those widely used standard video codecs. When compared with the RD performance of the CRG parallel DVC codec, it can be concluded that it out performs H.264/AVC Intra for low motion sequences under almost all the test conditions, except GOP size 8 for Coastguard sequence. There is also performance gain for more complex video sequence such as Foreman coded with GOP size 8 and Soccer with GOP size 2.

It is usually expected that WZ codec can hardly beat the performance of H.264/AVC No Motion. However, the CRG parallel DVC codec shows remarkable RD gains for high motion video sequences. Foreman sequence with GOP size 8, Soccer sequence with GOP size 2 and Coastguard sequence using GOP sizes 2 and 4, all performs better than H.264/AVC No Motion. However, there are no significant RD performance changes for the Hall Monitor sequence, i.e. the performance remains above H.264/AVC Intra and still below H.264 No Motion.

It can also be observed that, for low motion sequences such as Hall Monitor and Coastguard, CRG parallel DVC remains above or similar to H.264/AVC Intra under most situations. However, for high motion sequences such as Foreman and Soccer, the RD performance is still below H.264/AVC Intra for most settings. Comparing with H.263+ (Intra) codec, CRG parallel DVC is consistently better with exception for the most complex sequence Soccer, which only shows superior RD performance using GOP size 2.

6.4 Conclusion

First of all, a low complexity implementation of a LDPCA code based DVC encoder, on a TMS320DM6437 (DaVinci) DSP is presented in this chapter. We also implemented the DVC decoder on a general purpose PC. The performance of the DVC codec is verified on a DSP-PC architecture. Test results show that an improvement in speed of more than 29 times can be obtained against a non-optimized implementation. The optimized implementation shows that DM64x DSP is a suitable platform for the implementation of DVC encoder. As future work, this DSP based encoder can be used for online coding task, enabling real-time DVC applications.

A PC-HPC system architecture with emphasis on the parallel implementation of a transform domain DVC decoder is presented in the rest of this chapter. The experimental results show that the decoder exploiting 24 CPU cores in parallel processing can achieve about 10 times speedup under various bit-rates and GOP sizes compared to the serial implementation. Although the decoding speed is still far from real time requirement, it is strongly believed that a number of approaches can be considered to bring huge speedup to the decoder side to meet the real time requirements. Examples of these approaches include introducing a simple rate estimation module at the encoder side to remove the feedback channel, skip blocks that does not have significant changes over time, using early stop criteria for LDPCA decoding, which remain as our future work.

Chapter 7
Conclusions & Future Work

This thesis has proposed contributions to investigate and exploit side information to improve DVC RD performance, provided solution to achieve consistent quality of decoded video frames over time, proposed efficient DVC implementations across different hardware platforms. These will be summarized in Section 7.1 followed by future works on open research challenges in Section 7.2.

7.1 Conclusions

Chapter 2 reviewed the theoretical background and the development of DVC. This review identified the major research challenges in this field and addressing these problems is the objective and the main contribution of this thesis. The conclusions from this Chapter are 1) DVC can significantly benefit numerous emerging applications that require very low encoding complexity and hence it deserves wide attention from the research communities; 2) Most recent DVC developments show that despite DVC can already achieve very similar RD performance for some test sequences when compared to the conventional video codecs, it is still not ready from practical use and deployments. Therefore, further research work in this field is necessary; 3) Some of the major challenges in DVC have been identified and they are in the areas of side information creation and refinement, consistent quality control and efficient codec implementations.

To address the challenges identified in Chapter 2, Chapter 3 investigates the impacts of using reference frames as side information on RD performance and proposed a solution to improve the accuracy of motion search and reduce decoding complexity. The experimental results reveal that if the conditional bit probability for LDPCA decoder is

computed for quantized WZ frame, using reference frames as SI is capable to achieve similar or sometimes even better coding efficiency than the widely used MCI frames. The proposed MRP method can significantly reduce decoder complexity with no loss in RD performance. This work brings new insight and strength to the use of reference frames. It opens attractive perspectives that allow us to better understand the role of reference frames in DVC.

Chapter 4 presents a novel pixel granularity SI refinement framework to reduce the block artifacts introduced by widely used block based frame interpolation solutions. It also suggests a parallel implementation to improve the decoding speed within a state-of-the-art transform domain DVC codec. The experimental results show significant improvements on RD performance over the same codec without the proposed algorithms. The parallel implementation also shows high utilization of resources and substantial speedup when compared with the serial implementation. The updated SI frames during the SIS process demonstrate considerable improvement in both subjective and objective image quality against the widely used block based SIS algorithms. The proposed SIS framework can be integrated into any modern transform domain DVC codec to achieve a better RD performance especially for video sequences with complex motion and coded with long GOP sizes. The framework can also be re-configured to exploit more efficient optical flow algorithms to improve the performance and further reduce complexity. Furthermore, the proposed parallel implementation brings the state-of-the-art DVC codec one step closer to practical use.

As aforementioned, consistent quality of decoded video frames is sometimes favoured in real applications and therefore, a quality control mechanism is very much needed. To address this problem, Chapter 5 proposes a solution to control the video frame quality for coding both key frames and WZ frames through two distortion-

quantization models. As expected, simulation results show that the proposed method closely meet user defined target quality and smooth out the distortion variation for slow motion sequences and performs similar to fixed quantization settings obtained from offline trainings for fast motion sequences. However, it is also expected to have some RD performance loss as DVC usually require slightly better frame quality for key frames over WZ frames to achieve a better RD performance [19], but the quality control algorithm may not meet this condition.

An efficient implementation taking into account both software and hardware features and restrictions is essential for practical use and deployments of DVC. Chapter 6 demonstrates two fully implemented DVC codecs using different hardware architectures. The DSP-PC architecture shows that under the restriction of memory and processing power, DVC encoder is still capable to perform in a rather fast speed. The proposed optimization shows more than 29 times speedup against a non-optimized implementation. The conclusion drawn from this implementation is that DM64x DSP is a suitable platform for the implementation of DVC encoder. The PC-HPC architecture demonstrates a highly efficient parallel implementation to maximize the utilization of system resources at the decoder side. The experimental results show that the parallelized decoder can achieve about 10 times speedup under various bit-rates and GOP sizes compared to the serial implementation. The RD performance of this implementation beats one of the best-performed DVC codec (DISCOVER codec). Although the decoding speed is not yet satisfactory for real time requirement, it is strongly believed that a number of approaches can be considered to bring huge speedup to the decoder side for real time applications. We have also provided a thorough specification for the file structure of encoder output. This has not yet been discussed elsewhere in the

literature but it is a key protocol for encoder and decoder to communicate information and it is also essential for future standardization of DVC codec.

The research work presented in this thesis has resulted in 2 IEEE journal papers [36][88], 2 conference papers [89][90] and a highly efficient DVC codec deployed in the CRG research group.

7.2 Future Works

In summary, this thesis has proposed various solutions to bring DVC one step forward to practical use. The suggested techniques are capable of improving the overall RD performance and accelerating both of the encoding and decoding speed. However, future research work is still necessary to further enhance DVC performance. Possible research areas are summarized below.

7.2.1 Further Investigation on Computation of Conditional Bit Probability in Quantized Coefficient Domain

This thesis found that using reference frames as SI directly performs not worse than widely used interpolated frames, when the computation of conditional bit probability is in quantized symbol domain. When this information is calculated in non-quantized symbol domain, the RD performance is expected to be better as there is no information loss in the process. However, under certain distortion restraint, if the computation of the bit probability in quantized domain is already sufficient to meet the target distortion, it is not necessary to carry out this highly complex computation in non-quantized domain, which can bring significant complexity gains, especially when EM algorithm is adopted to consistently update bit probabilities.

7.2.2 Extend the Pixel Granularity SI Synthesis Framework to Use Extrapolated Frames

For real-time applications, using frame interpolation for SI generation may not be applicable as frame interpolation changes the original frame order and this requires frame buffering which may not be desirable. Furthermore, longer GOP sizes will have need of bigger buffer size which can increase the delay. Frame extrapolation can be one of the possible solutions. The proposed SI synthesis framework in this thesis is based on frame interpolation, but it can be extended in the future to use extrapolated frames to generate SI. Previously decoded frames can be used to extrapolate the SI for decoding the next frame according to the original frame order.

7.2.3 Efficient Quality Control Algorithm without Feedback Channel

The quality control algorithm proposed in this thesis requires sending back some information from the decoder to the encoder side to facilitate the distortion-quantization modelling process. Since a feedback channel is usually not desired for practical applications, an encoder side rate allocation algorithm can be integrated into our DVC codec. Therefore, our quality control algorithm will not be able to obtain the residual information from the decoder side. However, the fact that the conventional video decoder is usually far simpler than the encoder can be exploited. Therefore, in the future, key frames decoder can be added to the encoder side to generate the residual statistic information of the decoded key frames, facilitating the estimation of the distortion of AC coefficients. And hence, no information is required to be sent back to the encoder any more.

7.2.4 More Efficient and Practical DVC Implementation

Like most research work in the literatures, the DVC implementations in this thesis have only considered the brightness colour component. However, for realistic applications coloured output frames are usually preferred. As a future work, all the colour components can be taken into account in DVC. In this case, the correlation between different colour components and the correlation of the same colour component between neighbouring frames can be exploited.

As mentioned in section 7.2.3, an encoder rate control is usually required for practical DVC applications. It can remove the feedback channel and significantly reduce the decoding complexity. To further speedup the DVC decoder, a fast stopping criteria can also be introduced to accelerate the iterative decoding procedure. In addition, more hardware features such as GPGPU can be exploited to improve the parallel implementation as well.

Furthermore, more computationally efficient channel codes such as polar code [91] can be considered for practical DVC codec design. Like turbo and LDPC codes, polar codes facilitate near-capacity operation. However, polar codes do not require an iterative decoder, and hence can provide much lower coding complexity. This may increase the opportunities to use DVC for real-time applications.

Last but not least, since power restricted devices cannot afford to run a DVC decoder due to its high complexity, a transcoder can be introduced to achieve a “simple-to-simple” transmission of video data. An intuitive solution is to convert the decoded video frames into conventionally encoded data in a centralized base-station and then forward it to the target terminal. This base-station serves as a transcoder to exploit the

fact that DVC encoder and conventional video decoder can perform fast coding tasks, bringing forth low complexity end-to-end encoding as well as decoding.

References

[1] "Coding of audio-visual objects-Part 2: Visual," in *ISO/IEC 14496-2 (MPEG-4 Visual Version 1)*, Apr. 1999.

[2] "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264/ISO/IEC 14496-10 AVC)," in *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG*, JVTG050, 2003.

[3] B. Girod, A. Aaron, S. Rane and D. Rebollo-Monedero, "Distributed Video Coding," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 71-83, Jan. 2005.

[4] C. Brites and F. Pereira, "Correlation noise modeling for efficient pixel and transform domain Wyner-Ziv video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 9, pp. 1177–1190, 2008.

[5] F. Pereira, C. Brites, J. Ascenso, and M. Tagliasacchi, "Wyner-Ziv video coding: a review of the early architectures and further developments," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '08)*, pp. 625–628, Hannover, Germany, June 2008.

[6] J. D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. IT-19, pp. 471–480, July 1973.

[7] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. IT-22, no. 1, pp. 1–10, Jan. 1976.

[8] C.E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623-656, July, October, 1948.

[9] R. Puri and K. Ramchandran, "PRISM: a new robust video coding architecture based on distributed compression principles," in *Proceedings of Allerton*

Conference on Communication, Control and Computing, Allerton, USA, Oct. 2002.

[10] R. Puri, A.Majumdar, and K. Ramchandran, "PRISM: a video coding paradigm with motion estimation at the decoder," *IEEE Transactions on Image Processing*, vol. 16, no. 10, pp. 2436–2448, 2007.

[11] A. Aaron, R. U. I. Zhang, and B. Girod, "Wyner-Ziv coding of motion video," in *Proceedings of the 36th Asilomar Conference on Signals Systems and Computers*, pp. 240–244, Pacific Grove, Calif, USA, Nov. 2002.

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

[13] Claude Berrou, Alain Glavieux and Punya Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," *International Communications Conference (ICC)*, Geneva, Switzerland, May 1993, pp. 1064-1070.

[14] R. G. Gallager, "Low Density Parity Check Codes," Monograph, *M.I.T. Press*, 1963.

[15] Toby Berger, "Rate-distortion theory: A mathematical basis for data compression, " *Englewood Cliffs*, NJ: Prentice-Hall, 1971.

[16] Peterson, W. W. and Brown, D. T., "Cyclic Codes for Error Detection," *Proceedings of the IRE 49* (1): 228–235, Jan. 1961.

[17] Hocquenghem, A., "Codes correcteurs d'erreurs", *Chiffres* (in French) (Paris) 2: 147–156, Sep. 1959.

[18] R. C. Bose, D. K. Ray-Chaudhuri, "On A Class of Error Correcting Binary Group Codes", *Information and Control*, 3 (1): 68–79, Mar. 1960.

[19] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov and M. Ouaret, "The DISCOVER codec: Architecture, Techniques and Evaluation," *Picture Coding Symposium 2007*, Lisbon, Portugal, Nov. 2007.

[20] The DISCOVER Codec Evaluation, http://www.img.lx.it.pt/~discover/home.html.

[21] D. Varodayan, A. Aaron, and B. Girod, "Rate-adaptive distributed source coding using low-density parity-check codes," in *Proceedings of the 39th Asilomar Conference on Signals, Systems and Computers*, pp. 1203–1207, Pacific Grove, Calif., USA, Nov. 2005.

[22] D. Varodayan, A. Aaron, and B. Girod, "Rate-adaptive codes for distributed source coding," *EURASIP Signal Processing Journal*, vol. 86, no. 11, pp. 3123–3130, Nov. 2006.

[23] D. Kubasov, J. Nayak, C. Guillemot, "Optimal reconstruction in Wyner-Ziv video coding with multiple side information," *IEEE International Workshop on Multimedia Signal Processing*, Crete, Greece, Oct. 2007.

[24] ITU Telecom. Standardization Sector of ITU, "Video coding for low bitrate communication," *Draft ITU-T Recommendation H.263 Version 2*, Sep. 1997.

[25] J. Ascenso and F. Pereira, "Integrated software tools for distributed video coding," *VISNET II Deliverable D1.2.3*, Jun. 2009.

[26] C. Brites, J. Ascenso, J. Pedro and F. Pereira, "Evaluating a feedback channel-based transform domain Wyner-Ziv video codec," *Signal Process.: Image Commun.*, vol. 23, p.269 , 2008.

[27] A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, B. Pesquet-Popescu, "Fusion of Global and Local Motion Estimation for Distributed Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 23, Issue 1, Jan. 2013.

[28] H Luong, L Raket, X Huang, S Forchhammer, "Side Information and Noise

Learning for Distributed Video Coding using Optical Flow and Clustering," IEEE Transactions on Image Processing 21 (12), 4782-4796, 2012.

[29] A. Aaron, S. Rane, E. Setton and B. Girod, "Transform- Domain Wyner-Ziv Codec for Video," *Visual Communications and Image Processing (VCIP 2004)*, San Jose, USA, Jan. 2004.

[30] J. Ascenso, C. Brites and F. Pereira, "Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding," *5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services*, Slovak Republic, Jul. 2005.

[31] L. Natario, C. Brites, J. Ascenso and F. Pereira, "Extrapolating Side Information for Low-Delay Pixel Domain Distributed Video Coding," *Int. Workshop on Very Low Bitrate Video Coding*, Sep. 2005.

[32] Borchert, S., R. P. Westerlaken, R. Klein Gunnewiek, and R. L. Lagendijk, "On Extrapolating Side Information in Distributed Video Coding," *26th Picture Coding System*, Lisbon, Portugal, Nov. 2007.

[33] A. B. B. Adikari, W. A. C. Fernando, W. A. R. J. Weerakkody, "Multiple Side Information Streams for Distributed Video Coding," *IET electronics Letters*, vol. 42, issue 25, pp. 1447-1449, Mar., 2006.

[34] Pearl Judea, "Reverend Bayes on inference engines: A distributed hierarchical approach," *Proceedings of the Second National Conference on Artificial Intelligence*, Pittsburgh, PA. Menlo Park, California, pp. 133–136, 1982.

[35] D. Varodayan, D. Chen, M. Flierl and B. Girod, "Wyner-Ziv coding of video with unsupervised motion vector learning," *Signal Process.: Image Commun.*, vol. 23, pp. 369–378, Jun. 2008.

[36] M. Zheng, F. H. Ali, "Exploration and Exploitation of Reference Frames in Distributed Video Coding," *IEEE Signal Processing Letters*, vol. 19, issue 7, pp. 411 – 414, Jul. 2012.

[37] R. Martins, C. Brites, J. Ascenso and F. Pereira, "Statistical motion learning for improved transform domain Wyner-Ziv video coding," *Image Processing, IET*, vol. 4, no. 1, pp. 28-41, Feb. 2010.

[38] X. Artigas and L. Torres, "Iterative generation of motion-compensated side information for distributed video coding," in *2005 IEEE Int. Conf. on Image Processing*, vol. 1, pp. I-833-6, Sep. 2005.

[39] R. Martins, C. Brites, J. Ascenso, F. Pereira, "Refining side information for improved transform domain Wyner-Ziv video coding," *IEEE Trans. on Circuits and Systems for Video Technology*, pp. 1327–1341, Sep. 2009.

[40] M. B. Badem, M. Mrak and W. A. C. Fernando, "Side information refinement using motion estimation in DC domain for transform-based distributed video coding", *IET Electron. Lett.*, vol. 44, no. 16, pp. 965 -966, 2008.

[41] M. B. Badem, W. A. C. Fernando, J. L. Martinez and P. Cuenca, "An iterative side information refinement technique for transform domain distributed video coding," in *2009 Proc. IEEE Int. Conf. on Multimedia an Expo*, New York, 2009, pp. 177-180.

[42] N. Merhav and V. Bhaskaran, "Fast inverse motion compensation algorithms for MPEG and for partial DCT information," *J. Vis. Commun. Image Repres.*, vol. 7, pp.395-410, 1996.

[43] J. Ascenso, C. Brites and F. Pereira, "Motion compensated refinement for low complexity pixel based distributed video coding," in *Proc. IEEE Conf. Advanced Video Signal Based Surveillance*, pp.593, 2005.

[44] A. B. B. Adlikari, W. A. C. Fernando, W. A. R. T. Weerakkody and H. K. Arachchi, "A sequential motion compensation refinement technique for distributed video coding of Wyner-ziv frames," in *Conf. on Image Processing*, Atlanta, 2006, pp. 597-600.

[45] A. Aaron, S. Rane, and B. Girod, "Wyner-Ziv video coding with hash-based motion compensation at the receiver", *IEEE Int. Conf. Image Processing*, 2004.

[46] A. Trapanese, M. Tagliasacchi, S. Tubaro, J. Ascenso, C. Brites and F. Pereira, "Improved correlation noise statistics modeling in frame-based pixel domain Wyner-Ziv video coding," *Proc. of International Workshop on Very Low Bitrate Video Coding*, pp.1-4, 2005.

[47] C. Brites, J. Ascenso, F. Pereira, "Modeling Correlation Noise Statistics at Decoder for Pixel Based Wyner-Ziv Video Coding," *Picture Coding Symposium (PCS)*, Beijing, China, Apri. 2006.

[48] C. Brites, J. Ascenso and F. Pereira, "Studying temporal correlation noise modeling for pixel based Wyner-Ziv video coding," *Proc. of IEEE International Conference on Image Processing*, pp.273-276, 2006.

[49] J. Slowack, S. Mys, J. Skorupa, P. Lambert, R. Van de Walle, "Accounting for quantization noise in online correlation noise estimation for distributed video coding," *Proceedings of Picture Coding Symposium*, 2009.

[50] X. Huang, S. Forchhammer, "Improved virtual channel noise model for transform domain Wyner–Ziv video coding," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.

[51] H. V. Luong , X. Huang and S. Forchhammer, "Adaptive noise model for transform domain Wyner-Ziv video using clustering of DCT blocks," *Proc. IEEE Int. Workshop Multimedia Signal*, pp.1-6, 2011.

[52] J. Song, K. Wang, H. Liu, Y. Li, C. Wu, "Progressive correlation noise refinement for transform domain Wyner-Ziv Video Coding," *18th IEEE International Conference on Image Processing (ICIP)*, Brussels, Belgium, Sep. 2011.

[53] G. R. Esmaili and P. C. Cosman, "Wyner-Ziv video coding with classified correlation noise estimation and key frame coding mode selection," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp.2463-2474, 2011.

[54] Huynh Van Luong,S. Forchhammer, "Noise residual learning for noise modeling in distributed video coding," *Picture Coding Symposium (PCS)*, Krakow, pp. 157-160, May 2012.

[55] G. Wu, L. Sun and F. Huang, "Consistent-Quality Distributed Video Coding Framework," *PCM'07 Proceedings of the multimedia 8th Pacific Rim conference on Advances in multimedia information processing*, 2007.

[56] A. Roca, M. Morbee, J. Prades-Nebot, and Edward Delp, "A distortion control algorithm for pixel-domain Wyner-Ziv video coding," in *Proc. Picture Coding Symposium*, Lisbon, Portugal, Nov. 2007.

[57] S. Sofke, F. Pereira, E. Müller, "Dynamic quality control for transform domain Wyner–Ziv video coding," in *EURASIP Journal on Image and Video Processing*, Special Issue: Distributed Video Coding, 2009.

[58] Hongbin Liu, Debin Zhao, Siwei Ma and Wen Gao, "Constant quality control of Wyner-Ziv frames in DCT domain distributed video coding," *ICIC Express Letters*, Vol. 5, Number 7, Jul. 2011.

[59] R. Oh, J. Park, and B. Jeon, "Fast implementation of Wyner-Ziv Video codec using GPGPU," in *Proc. IEEE International Symposium on Broadband Multimedia Systems and Broadcasting 2010 (BMSB 2010)*, pp. 1-5, Mar. 2010.

[60] J. Fung and S. Mann, “Computer vision signal processing on graphics processing units,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004)*, pp. 83-89, Montreal, Quebec, Canada, May 17-21 2004.

[61] A. Corrales-Garcia, J.L. Martinez, G. Fernandez-Escribano, F. J. Quiles, W. A. C. Fernando, “Wyner-Ziv frame parallel decoding based on multicore processors,” *2011 IEEE 13th International Workshop on Multimedia Signal Processing (MMSP)*, Hangzhou, China, pp. 1-6, 17-19 Oct. 2011.

[62] Y. S. Pai, Y. C. Shen, J. L. Wu, “High efficient distributed video coding with parallelized design for LDPCA decoding on CUDA based GPGPU,” *Journal of Visual Communication and Image Representation*, vol. 23, Issue 1, pp. 63–74, Jan. 2012.

[63] NVIDIA Corporation, “NVIDIA CUDA C Programming Guide,” Santa Clara, CA, 2010.

[64] J. Park, B. Jeon, "Parallel transform domain Wyner-Ziv video encoding scheme for standard definition video," *2012 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1-4, Seoul, 27-29 Jun. 2012.

[65] R. Chandra, R. Menon, L. Dagum, D. Kohr, D. Maydan, J. McDonald, “Parallel Programming in OpenMP,” Morgan Kaufmann, 2000.

[66] A. C. Garc á, J. L. Mart ínez, G. F. Escribano, F. J. Quiles, “Toward fast Wyner-Ziv video decoding on multicore processors,” *Multimedia Tools and Applications*, Springer, Apri. 2012.

[67] J. Chen, T. Zhang, W. Cui and W. Wu, “Reducing quantisation loss and improving side information in distributed video coding”, *IET Electronics Letters*, vol. 47, issue 1, pp. 30-31, Jan. 2011.

[68] A.P.Dempster, N.M.Laird, D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society*, Series B 39 (1): 1–38, 1977.

[69] H.264/AVC Reference Software, version JM18.2, URL: http://iphome.hhi.de/suehring/tml/download/.

[70] T. Brox, A. Bruhn, N. Papenberg and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. European Conf. on Computer Vision*, vol. 4, pp. 25–36, 2004.

[71] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial* Intelligence, vol.17, pp. 185-203, 1981.

[72] Y. Adato, O. Ben-Shahar, etc. Available: http://code.google.com/p/optflowb/.

[73] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, R. Szeliski, Avaliable: http://vision.middlebury.edu/flow.

[74] K. Chen, D. A. Lorenz, "Image sequence interpolation based on optical flow, segmentation, and optimal control," *IEEE Trans. on Image Processing*, vol. 21, no. 3, 2012, pp. 1020-1030.

[75] N. Sundaram, T. Brox, and K. Keutzer, "Dense point trajectories by GPU-accelerated large displacement optical flow," *European Conf. on Computer Vision*, LNCS, Springer, Heidelberg, 2010.

[76] S. Forchhammer, H. Li and J. Dahl Andersen, "No-reference analysis of decoded MPEG images for PSNR estimation and post-processing," *Journal of Visual Communication and Image Representation*, Vol. 22, Issue 4, pp. 313-324, May 2011.

[77] A. Aaron, E. Setton and B. Girod, "Towards practical Wyner-Ziv coding of video," *Proc. IEEE International Conference on Image Processing*, ICIP-2003, Barcelona, Spain, Sep. 2003.

[78] ITU-T, I. JTC1, “Digital compression and coding of continuous-tone still images,” *ISO/IEC 10918-1 ITU-T Recommendation T.81 (JPEG)*, 1994.

[79] Spectrum Digital, Inc., “TMS320DM6437 Evaluation Module Technical Reference,” Dec. 2006.

[80] Texas Instr., “Digital Media Processor DM6437 Datasheet,” http://www.ti.com/product/tms320dm6437, accessed in Jun. 2013.

[81] “C64x+ Image Library (IMGLIB),” http://www.ti.com/tool/sprc264, accessed in Jun. 2013.

[82] W. H. Chen, C. H. Smith, and S. Fralick, “A Fast Computational Algorithm for the Discrete Cosine Transform,” *IEEE Trans. on Communications*, 25, 1004–1009, Sep. 1977.

[83] A. D. Liveris, Z. Xiong, and C. N. Georghiades, “Compression of binary sources with side information at the decoder using LDPC codes,” *IEEE Commun. Lett.*, vol. 6, no. 10, pp. 440–442, Oct. 2002.

[84] “Multimedia Framework Products (MFP) - Codec Engine and xDAIS Framework Components,” http://www.ti.com/tool/tmdmfp, accessed in Jun. 2013.

[85] S. Y. Chung, T. J. Richardson, and R. Urbanke, “Analysis of sum–product decoding of low-density parity-check codes using a Gaussian approximation,” *IEEE Trans. Inform. Theory*, vol. IT-47, pp. 657–670, Feb. 2001.

[86] W. E. Ryan, B. Vasic, “An introduction to LDPC codes,” *CRC Handbook for Coding and Signal Processing for Recording Systems*, 2004.

[87] D. Kubasov , K. Lajnef and C. Guillemot, “A hybrid encoder/decoder rate control for Wyner–Ziv video coding with a feedback channel,” *Proc. IEEE Multimedia Signal Processing (MMSP 2007)*, pp.251-254, 2007.

[88] M. Zheng, F. H. Ali, “Pixel Granularity Side Information Synthesis Framework

and Parallel Implementation for Distributed Video Coding," submitted to *IEEE Transactions on Multimedia*.

[89] M. Zheng, F. H. Ali, "DSP Implementation of On-Board Distributed Video Coding," *4th European DSP Education and Research Conference, EDERC2010*, 5 pages, Nice, France, Dec. 2010.

[90] M. Zheng, F. H. Ali, "Consistent Quality Control for Wireless Video Surveillance Using Distributed Video Coding," *The 4th International Conference on Imaging for Crime Detection and Prevention (ICDP-11)*, London, UK, Dec. 2011.

[91] E. Arikan, "Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. on Information Theory*, 55(7): pp. 3051-3073, 2009.