國立交通大學

多媒體工程研究所

碩 士 論 文

解碼端影像誤差估測用於分散式視訊編碼法校正優先權的設計

Adaptive Decoder Side Information Error Estimation for Priority-based Error Correcting Distributed Video Coding

研 究 生：連曉玉

指導教授：蔡淳仁 教授

中 華 民 國 九 十 八 年 二 月

解碼端影像誤差估測用於分散式視訊編碼法校正優先權的設計
Adaptive Decoder Side Information Error Estimation for Priority-based Error Correcting Distributed Video Coding

研 究 生：連曉玉　　Student：Shiau-Yu Lian

指導教授：蔡淳仁　　Advisor：Chun-Jen Tsai

國 立 交 通 大 學
多 媒 體 工 程 研 究 所
碩 士 論 文

A Thesis
Submitted to Institute of Multimedia Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer Science

February 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年二月

解碼端影像誤差估測用於分散式視訊編碼法校正優先權的設計

研究生 : 連曉玉　　　　　　　　指導教授: 蔡淳仁

國 立 交 通 大 學
多 媒 體 工 程 所

摘要

本篇論文提出了一種使用於分散式視訊編碼的技術，對於 W-Z frame 提供了有優先順序的 channel coding。在我們所提出的架構中，基於預估的 side information 誤差程度，W-Z frame 的 macro-block 會被分類為幾個不同的群組。分類的資訊經由一個上傳 channel 傳回編碼端，所以誤差程度相似的 macro-block 會被聚集在一起以進行 channel coding。使用這種方法，解碼端可以為 side information 品質比較不好的 macro-block 要求多一點的 parity bits 以更正這些部分的錯誤。而對於 side information 誤差較小的 macro-block 則要求比較少的 parity bits。比起目前最好的 DISCOVER DVC codec，對於 QCIF sequences，在較低的 bitrate 範圍裡(低於 200 kbps)我們所提出的 DVC 架構可以使 R-D performance 增加 0.3 至 0.5 dB。因為低複雜度的編碼器對於像是 sensor network 之類的低 bitrate 監視應用系統是很重要的，所以我們所提出的架構可望應用到實際系統上。

Adaptive Decoder Side Information Error Estimation for Priority-based Error Correcting Distributed Video Coding

Student: Shiau-Yu Lian **Advisor: Dr. Chun-Jen Tsai**

Institute of Multimedia Engineering
National Chiao Tung University

Abstract

In this thesis, a distributed video coding technique with prioritized channel coding of W-Z frames is presented. In the proposed framework, W-Z frame macro-blocks are classified into different groups based on the estimated errors of the side information. The information is transmitted via uplink channel back to the encoder so that macroblocks with similar error statistics can be grouped tighter in same coding blocks for channel coding. With this approach, decoder can request more parity bits to correct macroblocks whose side information quality is worse and request less or no parity bits for macroblocks with small side information errors. The proposed DVC scheme can increase R-D performance about 0.3 to 0.5 dB over the state-of-the-art DISCOVER DVC codec for low bitrate (less than 200 kbps) applications for QCIF sequences. Since low-complexity encoder is important for low bitrate surveillance applications such as those for sensor networks, the proposed scheme is very promising for practical applications.

謝誌

首先我要感謝我的指導教授蔡淳仁老師這兩年半的教導，他總是很細心地對我的研究提出建議與方向，也不厭其煩地解答我們學生學術上的疑惑。再來我要感謝 MMES 實驗室的所有同學，大家互相關心、討論生活與課業上的問題，和你們在一起的時光令人感到開心。特別要感謝域晨學長，在這段時間裡跟我討論有關 DVC 的研究，讓我獲益良多。

我也要感謝我的男友順宇在這段期間的支持，以及我的家人，總是不給我帶來壓力，讓我可以放心地去學習、研究。最後要謝謝我的朋友依婷與孜穎，無論何時想到你們就讓我很溫暖。

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Distributed video coding (DVC) is a new video coding paradigm which allows more flexible coding complexity distribution between encoder and decoder. Traditional video codecs, for example, H.264, is designed for the situation where video is encoded once and decoded many times. The coding efficiency is mainly determined by the computational power of the encoder. But now there are many new applications, such as sensor networks and security camera systems, where the computational power of the encoder (sensors or cameras) is weaker than the decoder (central receiver of recorded videos). For these applications, a reversed paradigm is needed, and distributed video coding is suitable for this situation.

In traditional closed-loop video coding systems, motion estimation at the encoder side is used to eliminate temporal correlation of video data and the information of correlation is transmitted to decoders as motion vectors. The main idea of DVC is to estimate-and-construct the inter-frame correlation at the decoder side with little help from the encoder; therefore, the computation burden of motion estimation is shifted to the decoder [1]. For a typical DVC system, the video source would be divided into two interleaving sub-sequences: key frame subsequences and Wyner-Ziv (W-Z) frame subsequences. Key frames would be encoded using traditional encoder (such as the motion JPEG encoder or any video encoder).

For W-Z frames, the encoder takes the original video frame as input and applies a low-complexity algorithm to predict and generate some data (refer to as W-Z bits) that can help the decoder correct any errors in generation of the target frames (refer to as the side information). On the decoder side, the side information (SI) would be generated first using any information reconstruction technologies based on temporal correlation of neighboring key frames. And then, a W-Z decoder uses the W-Z bits from the encoder to correct any potential errors in the SI such that the resulting W-Z frames would be close to the original frames at the encoder side. The key components in a DVC framework are the W-Z bits generator and the SI generator.

The main complexity of encoder in traditional video codec architecture is due to motion estimation. For distributed video coding, we want a simple, low power encoder and a powerful decoder. So the encoder cannot perform motion estimation anymore; the job of predictive coding should be shifted to the decoder side. Whether this new paradigm can achieve the same coding performance as traditional video codec is an important research topic. According to previous information theory research results, the compression efficiency of distributed video coding should match that of traditional video coding techniques. Two of the most fundamental results related to the concept of distributed video coding from information theory are Slepian-Wolf theorem [2] and Wyner-Ziv theorem [3]. The latter is a lossy version of the former theorem.

Consider when we want to encode two statistically dependent variables, X and Y. According to information theory, fewer bits (H(X, Y)) are needed to describe the two variables if we jointly encode them. For video coding, two successive frames can be coded more efficiently if we consider their predictable relationship by motion estimation, and then encode the unpredictable residuals. But if the two variables X and Y are separately encoded, how many bits are required to describe them? For video coding, what bitrate will be required if frames are intra coded instead of inter coded?

The Slepian-Wolf theorem tell us that even if two variables X and Y are separately encoded, once they are jointly decoded, only H(X, Y) bits are required to decode them. For video coding, when frames are intra coded, if the decoder can jointly decoded them, the same coding efficiency can be reached. So, theoretically, in distributed video coding, if the decoder can jointly decode frames, that is, the decoder know the relationship between the frames and use the information to decode the frames, then only H(X, Y) bits will be sent to the decoder even if the encoder encodes the frames in intra mode. The theorem tells us that the performance of distributed video coding should as good as that of traditional video coding in theory. But until today, there is no technique based on distributed video coding principle whose performance can be close to that of traditional video coding.

In this thesis, a macroblock rearranging method is proposed to enhance R-D performance of distributed video coding. Existing distributed video coding methods performs particularly worse at low bit-rate end of the R-D curve. For QCIF sequence with frame rate 15 Hz, the proposed techniques can improve the performance of existing method by about 0.5 dB in PSNR measure when bit-rate below 200 kbps. This bit-rate range is reasonable for QCIF sequence. The proposed technique uses some cues to detect at the decoder side which part of the predicted side information has potentially high prediction errors and provide the encoder with this information. The encoder then rearranges the macroblocks so that the channel code-based W-Z bits generation process can be more efficient. As a result, the decoder can request W-Z bits to correct the hard-to-predict macroblocks first. Experimental results show that this technique outperforms current distributed video coding techniques, particularly at low bit-rate ends.

The organization of this thesis is as follows. Chapter 2 presents a literature survey on previous work of distributed video coding. In chapter 3, some analyses are conducted to identify the weakness of current techniques. In chapter 4, the proposed block rearranging method and system architecture will be described. And the experimental results will be presented in chapter 5. Finally discussions and future work are given in chapter 6.

Chapter 2: Previous Work

The key components of distributed video coding are generation of side information (predicted target frame at the decoder, using only key frames as the hypotheses) and coding of W-Z bits (data used to correct prediction error at the decoder side). Currently, there are two major approaches on side information generation: projection-based motion compensation and cue-based motion compensation. The first method performs moving block trajectory projection based on motion smoothness assumption. The projection-based compensation has several variations such as multi-frame reference projection [20] or sub-pixel motion projection [5].

An alternative approach on SI generation is the cue-based motion estimation scheme. In this method, the encoder sends not only W-Z correction bits but also some image cues for side information generation. The decoder can thus do more accurate inter-frame correlation discovery using cue bits. Several techniques have been proposed for cue generating functions, such as CRC codes [6], high pass filters in the DCT domain [40], and low pass filters in the DCT domain [13]. Although projection-based methods do not require the encoder to send extra cue bits, comparing to cue-based estimation, projection-based methods only performs well with ideal motion behavior. On the other hand, cue-based estimation could provide more robust and accurate side information but consumes extra bandwidth.

As in traditional close-loop codecs, it is not efficient to use only one coding tool to handle coding of all video signals. For DVC, the motion projection-based tool and the cue-based tool are suitable for constructing SI for different types of video signals. Therefore, low-complexity mode decision at the encoder side is crucial for improving the performance of DVC codecs. Some DVC frameworks use the differences of co-located blocks to distinguish the background from the foreground and to determine the cue bits [13]; however, this approach is limited to stable background patterns.

Kang and Lu proposed a wavelet-based DVC framework [26]. They use Structural Digital Signature (SDS) [30] to determine which wavelet coefficients are important, and only these coefficients are sent. For a wavelet tree structure, SDS records magnitude relation of every node and its child node. If the SDS of wavelet coefficients is different between two frames, then the two frames are perceptually different.

SDS of each frame are calculated and compared with the previous frame. Wavelet coefficients with SDS different from their co-located blocks in previous frame are marked as important coefficients. Only important coefficients are sent to the decoder. For non-important coefficients, the decoder copies coefficients from previous frame. If there are too many important coefficients, then H.264 intra coder will be used instead. This is a framework that both the complexity of the encoder and the decoder are low at the same time.

Figure 1 is the RD performance of this method. The plot above in this figure is R-D curve of Hall sequence, and the plot below is R-D curve of Foreman sequence. Most part of the Hall sequence is the background without motion, so the number of important coefficients is small. The performance is near other DVC methods, but comparing to a traditional video codec such as H.264, there is still a gap. For the Foreman sequence, most part of the frames is composed of the moving face, the PSNR of current frame and previous frame is small, so the number of important coefficients is large. Most blocks are coded using H.264 intra coder, so the performance is near the H.264 intra case. We can see, this method is only suitable for sequences with little foreground moving objects, such as in some security camera applications.

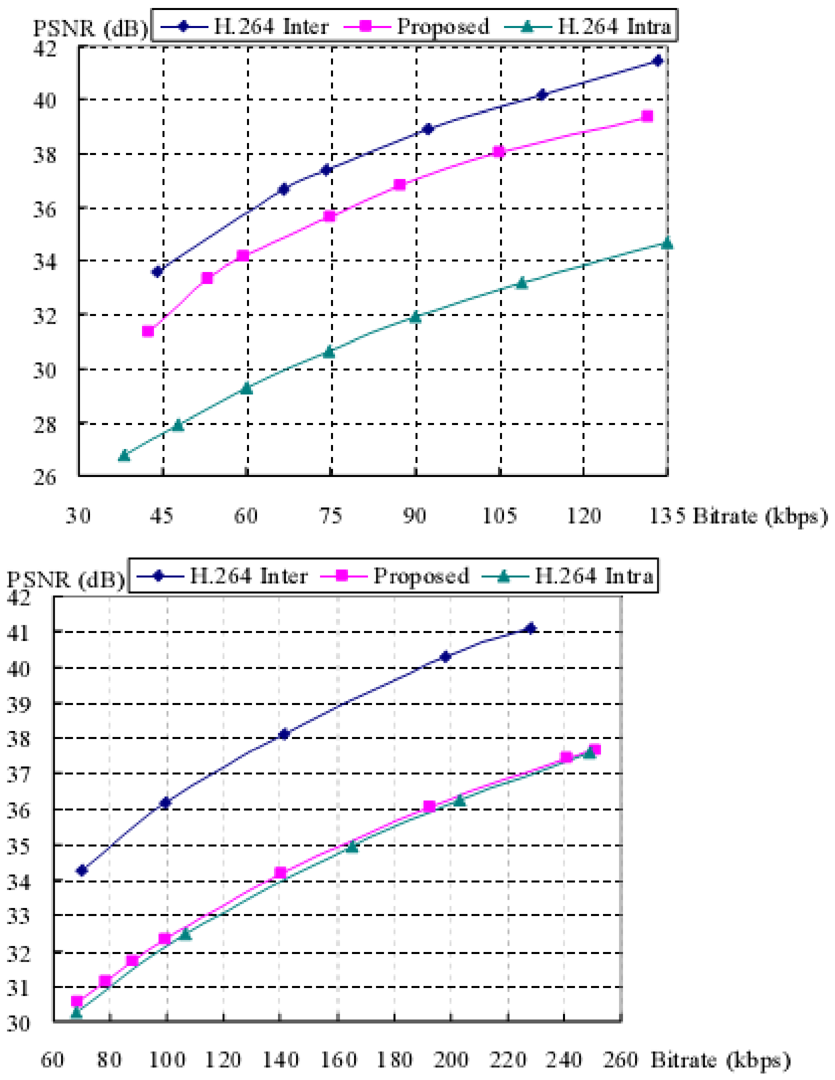


Figure 1. Performance of the cue-based DVC scheme in [22]

2.1 Stanford's DVC Frameworks

One of the most adopted DVC framework is proposed by Aaron et al. [1][7][8][11][12][13][14]. In this work, the encoder consists of a quantizer, a channel encoder, a buffer, and a traditional intra encoder. The decoder consists of a channel decoder, a side information generator, a frame reconstructor, and a traditional intra decoder.

At the encoder side, frames in a video sequence are first divided into even frames and odd frames. The odd frames are intra coded using traditional video codec such as H.264, and these frames are called 'key frames.' The even frames are like B-frames in traditional video codecs, and these frames are called ‘W-Z frames.’ But in DVC the encoder does not perform motion estimation and encode the residual. Instead, it treats these W-Z frames like I frames and encodes them independently by a channel encoder. The systematic channel encoder produces systematic message bits (the original pixel values) and parity bits (refer to as the W-Z bits), but only W-Z bits are stored in the buffer. The systematic bits are all discarded.

At the decoder side, for the ‘key frames,’ the decoder use traditional intra decoder to decode them. For the ‘W-Z frames,’ the decoder use key frames neighboring the W-Z frame to generate “side information” of this W-Z frame. The side information is like predictor in traditional video codec but is only known by the decoder. The decoder can use any information it has to generate (predict) side information. For example, the key frames already received can be used as the motion compensation hypotheses.

After generating the side information, the decoder needs additional bits for correcting the errors in the side information and reconstructing the original frames. This is like receiving residuals from encoder in traditional video codec. But the encoder does not know side information; it only generates W-Z bits of frame. The decoder requests for W-Z bits and use channel decoder to decode frames. The systematic bits are discarded at encoder side, so the decoder uses side information as the systematic bits. By the theory of channel coding, if the channel condition is good (that is, the decoder prediction error is small), the cross probability is small, and then fewer bits are needed for recovering the original signal. To be more specific, for DVC, if the side information of the target frame predicted by the decoder is almost the same as the W-Z frame we want to decode, then fewer W-Z bits are requested, coding efficiency is better.

The first version of DVC framework from Stanford does not perform coding in

DCT domain, and spatial redundancy within a frame is not explored. In [14], DCT domain coding is added to the framework because the performance is better.

In traditional video codecs, when GOP size increases, the coding efficiency will increase too. That is because there will be more P frames and B frames. In the first version of the Stanford DVC framework, GOP size is 2. That is, for every two frames, there is only one W-Z frame. Aaron et al. [11] try to use larger GOP size but the performance decreases as the GOP size increases. This is very different from traditional video codec. This is because when GOP size increases, the distance between W-Z frame and available key frames used to generate side information also increases, hence, the quality of the side information decreases.

To further improve the coding efficiency, Brites er al. from the DISCOVER DVC team [16] proposed a variable step size quantization which can make DCT coefficients less distorted when the dynamic range of coefficient band is smaller. That is, for a fixed number of quantization levels, if we know the dynamic range, then a sufficient and small step size can be used, and the distortion can be smaller. But additional bits are required for representing the value of dynamic range. This method may be inspired from H.264.

2.2 Side Information Generation

Side information quality directly affects coding efficiency in a DVC codec, therefore, a lot of research efforts in DVC are devoted to side information generation algorithms. In [17], some experiments are conducted to show the relation between PSNR of side information (compared with the original W-Z frame), and number of W-Z bits requested. In summary, when PSNR of side information is higher, fewer W-Z bits are needed and the compression ratio is higher.

2.2.1 Interpolation and Symmetric Motion Model

Side information can be interpolated or extrapolated from key frames and previously reconstructed WZ-frames. This is similar to motion-compensated frame interpolation which is used to increase frame rate at decoder side. There are many models being proposed for motion-compensated interpolation. For example, Liu et al. [29] assume that every frame has same motion field. This model of course is too simple so that the side information generated are not good. The symmetric motion model is a simple model but its performance is acceptable for some cases. Many researchers adopt symmetric motion model and use processing to enhance its performance [7][18].

The algorithm of the symmetric motion model is explained as follows. If the W-Z frame is frame Y, and its previous key frame is frame X, next key frame is frame Z. Symmetric motion model assumes that the object moves at a constant speed. So if the position of object O is (x_1, y_1) in frame X and (x_1+mv_x, y_1+mv_y) in frame Z, then the position of object O should be $(x_1+mv_x/2, y_1+mv_y/2)$. When the decoder wants to generate side information of frame Y, it has already received frame X and Z. So it can perform traditional motion estimation on frame X and Z. For macroblock M at (x_1, y_1) in frame Z, there is a best matched macroblock N at (x_1+mv_x, y_1+mv_y) in frame X. Then the decoder will project the average of M and N to $(x_1+mv_x/2, y_1+mv_y/2)$ onto frame Y along the motion path of M and N.

Of course there are many problems when using this model to implement side information generator. First of all, there may be some macroblocks which do not satisfy this single-object constant motion model. A macroblock may contain two or more objects, or objects which do not move at a constant speed. Secondly, when the decoder projects every matched macroblock pair to W-Z frame, there may be some pixels which have no projection, or twice or more projections. For pixels that are projected twice or more, it is not trivial to choose the best projection pair without knowledge of the original W-Z frame. For pixels that have no projections, the decoder must use some algorithms to interpolate their values as well.

Klomp et al. [5] believe that using motion estimation with more accuracy may increase side information quality. However, due to motion model mismatch, the actual improvements could be very little. Li and Delp [20] conducted some experiments on the amount of macroblocks in key frames used to generate side information. They discovered that the more macroblocks used, the better side information is generated, especially when these macroblocks come from different key frames.

In the above two proposals, decoders must do more computations to either increase motion estimation accuracy or to increase the number of reference macroblocks used. Ascenso et al. in DISCOVER team [18] adopt a different approach and develop a pixel domain DVC codec IST-PDWZ and a transform domain DVC codec IST-TDWZ. Their proposed technique is also based on the framework from Stanford with symmetric motion model. But before performing motion estimation on key frames, a low pass filter is used to make the estimated motion vectors less noisy. After obtaining motion vectors, if macroblock Y would be the average of macroblock X and Z. Bi-directional motion estimation is then used to fine-tune the motion vectors. After that, a smoothing filter is applied to the motion vectors because the motion field should be smooth. We can see the R-D performance in 0, this adjustment indeed makes performance better.

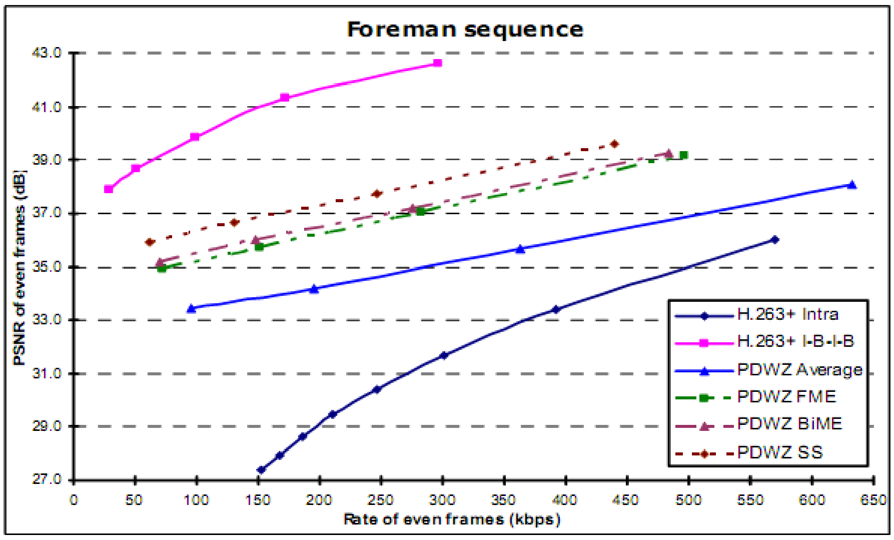


Figure 2. R-D performance of the DVC scheme in [18]

2.2.2 Hash Information as Motion Cue

In previous section we see how to use symmetric motion model to utilize key frames to generate good side information. We know symmetric motion model is limited. To further increase the quality of side information, it is possible for the encoder to compute and transmit some extra information to assist the decoder to generate better side information. Some researchers refer to this information as hash [13]. When decoder wants to generate side information, it does not perform motion estimation on neighboring key frames. Instead, it can obtain motion vector by directly comparing hashes of macroblocks. When hashes of two macroblocks are similar, the content of these two macroblocks are expected to be similar too.

Traditional hashes, for example, CRC and MD5, are very sensitive to content changing. But hashes used to compare image similarity should only be sensitive to perceptual changes in the pixel data. This kind of hash is called media hash, robust hash, soft hash, and image fingerprinting. Distance between hashes can be used as a measure of content similarity. If the hash is good enough, that is, a decent representation of macroblocks, motion estimation by comparing hash shall give us a good result.

Media hash design is an active research topic, and many papers have already been published. But most of these papers design media hash for content-based retrieval, watermarking, image authentication, and image database management. Media hash can be implemented by calculating the histogram of image [34], or the position of edges [35] or feature points [36]. DCT sign information is also useful [37]. The researchers hope that the media hash can be robust to geometric transformation and compression. So calculations of some media hashes are sometimes too complicated, and they are not suitable for DVC. Furthermore, these applications do not concern about the size of hashes, but for DVC, hash size affects bitrate.

For DVC, simple and small hashes are considered. In [13], using hash to help decoder to generate side information is first mentioned. The authors use sub-sampled and coarsely quantized version of macroblock in W-Z frame as their hash. To decrease overhead of hash, when co-located macroblock in previous key frame is similar to macroblock in current W-Z frame, no-hash bit is sent instead.

Girod et al. [8] also mentioned that the possibility of using high frequency portion of a macroblock as a distinct feature of the block. For an 8-by-8 block, 54 most high frequency coefficients (most of them may be zeros) with run-length and Huffman compression are sent to decoder as hashes. The remaining 10 low frequency coefficients are channel encoded and decoded. Although this idea seems reasonable, but in practice, it is not trivial for decoder to perform motion estimation based on high frequency coefficients.

2.3 Channel Coding for Side Information Correction

The key concept in DVC is that the side information predicted at the decoder side can be regarded as a video frame corrupted during the transmission over a (hypothetical) noisy channel. In DVC, the communication channel between the encoder and the decoder is a virtual noisy channel and side information is the noisy version of the original frame transmitted from the encoder across the channel to the decoder. Since the encoder has knowledge of the original frames, it can generate systematic channel codes to correct any potential errors of the side information at the decoder side. In practice, only the W-Z bits of the channel codes have to be transmitted to the decoder. There are many factors of channel coding that affect the coding efficiency of the DVC framework. Two of the most significant factors are the model of the virtual noisy channel and the channel code used.

2.3.1 Virtual Channel Model

For channel decoding, fewer W-Z bits will be requested if we know more about the nosie characteristics of the channel. But in DVC the noise is strongly related to side information quality which is not stationary. In early days, only one single statistics model (usually Laplace distribution), is used to describe the noise distribution for the whole sequence or the whole frame. This is not practical, because actual noise distribution is far more complicated and cannot be modeled by a single Laplace distribution. Borchert et al. [21] propose that if each frame is partitioned into two types of regions, occluded regions and non-occluded regions, and two different statistics models are used to model the errors for different types of regions, the performance will be better than using one statistics model for the whole frame.

Since decoder does not have the original frames, it can not get the exact parameters of the error distribution. The decoder need to estimate the parameters. However, one may not need exact parameters since the decoding algorithm allows some tolerance on the exactness of the statistics model. The tolerance range of the parameters is different from frame to frame. A distribution which is not sensitive to errors in parameter estimates and achieves high compression ratio such that decoder only needs few W-Z bits from the encoder to correct SI prediction errors is desired. Westerlaken et al. [22] have shown that when a single distribution is used to describe channel noise, two-sided Gamma and generalized Gaussian distribution have better decoding performance than Laplace and Gaussian distributions when LDPC codes are used. The compression ratio when using different error models is in Figure 3. They also discovered that for generalized Gaussian, there is a small range of shape parameter that makes LDPC performance less sensitive to the choice of the variance parameter. But they do not mention how to find this small range.

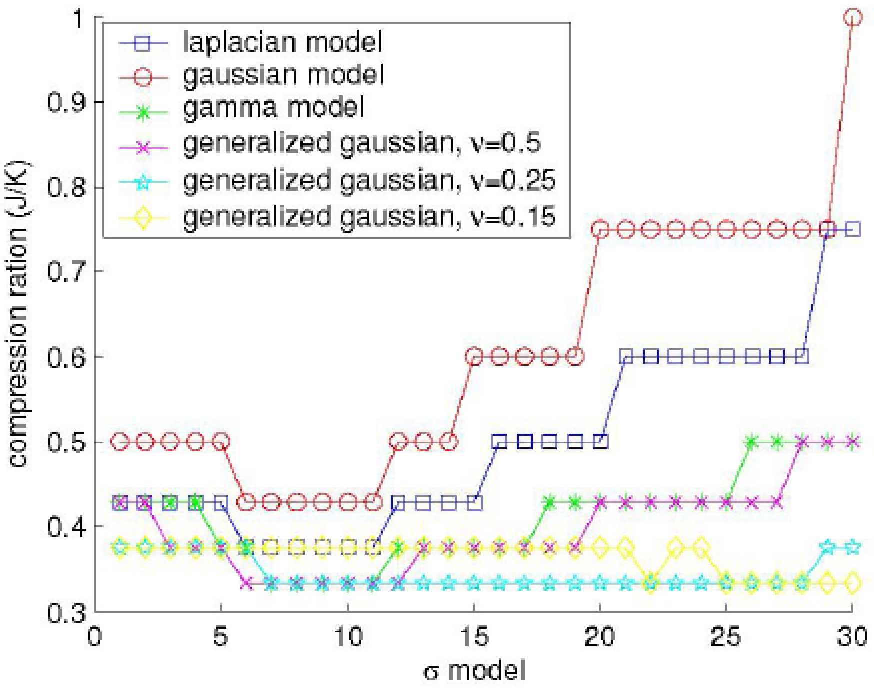


Figure 3. Compression ratio when different error models are used

2.3.2 Channel Code

In DVC, the virtual channel noise is not stationary, so rate-adaptive channel codes are needed in distributed video coding schemes. Channel codes near theoretical bound and rate-adaptive are suitable for DVC. Punctured turbo codes are well known for their burst error correcting capability and their performance is very close to Slepian-Wolf bound [23], so at the beginning, many DVC papers use turbo codes as their channel codes.

Low Density Parity Check (LDPC) codes are also channel codes near Slepian-Wolf bound. They have been used effectively in fixed-rate distributed source coding [25]. But in rate-adaptive cases, syndromes are punctured before they are sent, when compression ratio is high, the performance will be poor because in the decoding graph there are many single-connected or isolated nodes. The Stanford team presents two kinds of LDPC-based rate-adaptive codes: LDPC Accumulate codes (LDPCA) and Sum LDPC Accumulate codes (SLDPCA) [15]. The syndrome bits of LDPCA and SLDPCA codes contain more redundant information because of accumulation, so when a subset of syndrome bits are truncated, the performance will not be affected as much as the original LDPC codes. In Figure 4, circles are source bits and squares are parity bits after encoding. Left picture is the decoding graph of LDPCA code when all parity bits are not discarded. The center picture is the decoding graph of LDPCA code and the right picture is decoding graph of LDPC when half of parity bits are discarded. All nodes in the LDPCA decoding graph are neither isolated nor single-connected, so when compression ratio is high, LDPCA codes still have good performance.

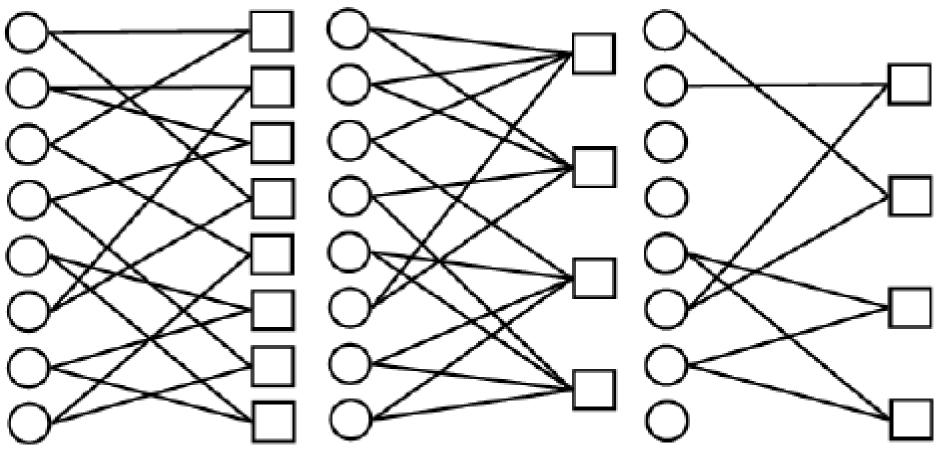


Figure 4. Decoding graph of LDPC codes and LDPCA codes

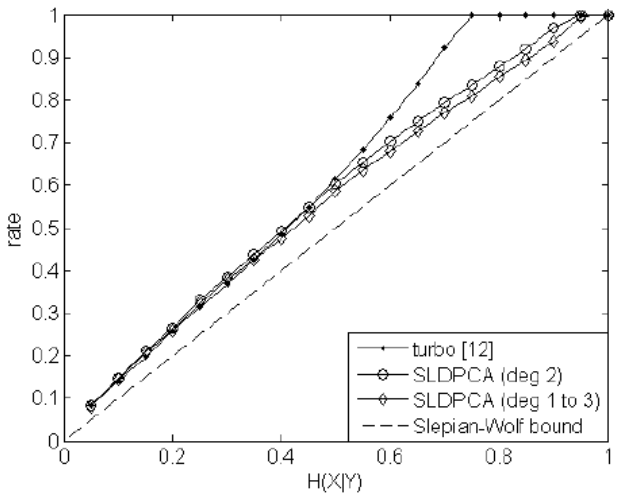


Figure 5. Rate required by turbo codes and SLDPCA codes

In 0, we can see that when conditional entropy H(X|Y) is larger than 0.5, the rate required for turbo codes increases faster and becomes bigger than LDPC codes, especially when H(X|Y) is larger than 0.8. Furthermore, LDPCA has one advantage over turbo codes, that is, syndrome bits can be used to test the correctness after decoding. When the error position can be detected after decoding, we can do something to enhance the regions where errors are serious. In [28], the authors decode the first 3 bit-planes and then perform full searches for side information which significantly reduces decoding errors. After adjusting the side information, the decoder keeps decoding remaining bit-planes, and the performance is better.

Chapter 3: Analysis on DVC Virtual Channel Error Characteristics

In this chapter, we have conducted some investigations on the performance of DVC-based coding mechanisms. In addition to examine the impact of error distribution models on coding efficiency, we also study the validity of the common DVC assumption of using log likelihood ratio as an indication of the reliability of the side information values. All experiments we describe here are tested using our pixel-domain DVC framework with lossless key frames. The frame rate is 30 fps.

3.1 Study on Virtual Channel Models

3.1.1 Types of Channel Models

In DVC coding schemes, a Laplace distribution L (μ, b) is in general used to model the prediction error of side information. The parameter μ is the mean parameter and often set to 0. The parameter b is the scale parameter. So, the probability density function of errors is $f(x) = \frac{b}{2} e^{-b|x-\mu|}$. This definition is often adopted in DVC papers, which is slightly different from the traditional Laplacian probability density function $f(x) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}}$. We use the definition former, so when the errors are more concentrated, the value of scale parameter will be larger. The two parameters may be estimated using some sample video sequences offline or estimated by the decoder adaptively while decoding a sequence. However, our initial experiments show that when a single distribution model with few parameters is used, the coding efficiency of the DVC schemes is relatively insensitive to the distribution being used to model the "channel-imposed" error of side information. In other words, correctly choosing parameters of distribution does not affect number of LDPC correction bits very much.

The reason is that the side-information prediction error model is quite different from the communication channel error model. In side-information prediction, large errors may also occur when energy of original signals are high due to errors in motion compensation models. That is, even when pixel values in original frames are big, it is still possible that the side-information errors are serious.

In order to test the relation between the accuracy of channel error model parameters and the error-correction efficiency on the prediction errors of the side information, we use three different distributions: Laplace, Gaussian, and uniform distributions to model channel errors. The parameters of Laplace distribution and Gaussian distribution are calculated offline. The uniform distribution is used to simulate the situation where we do not know error distributions at all.

In addition, an adaptive channel error model is also used in which, the distribution parameters are estimated not for the whole sequence, but adaptively for every 44×48 region of pixels. Therefore, the adaptive error model should fit the true error distribution mush better than the non-adaptive ones. The histograms of the estimated Laplace parameters of the side-information prediction error models for the 'Foreman' and the 'Mother-and-Daughter' sequences are shown in Figure 6. One can see that when the 'scale' parameter of Laplace distribution is larger, the error distribution is more concentrated around the mean. Many areas in the 'Mother-and-Daughter' sequence have very small residuals, so when using a Laplace model to fit it, the scale parameters are larger than that of the 'Foreman' sequence.

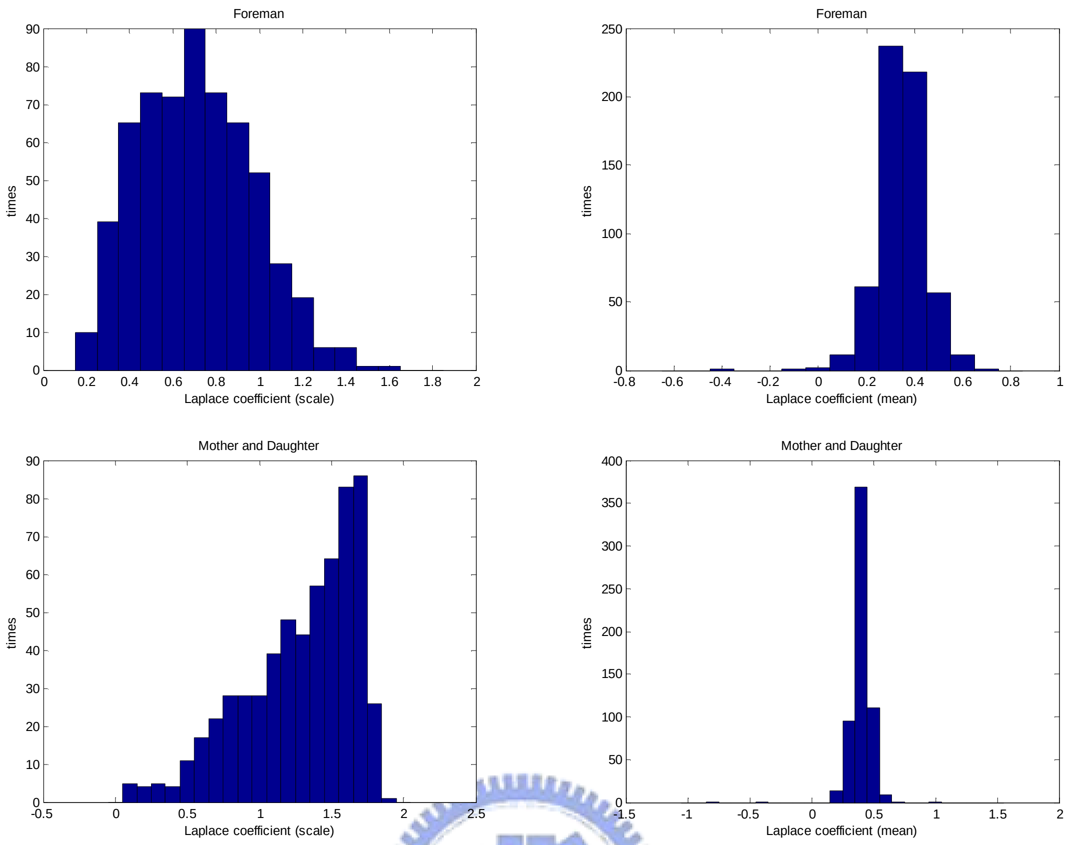


Figure 6. Histograms of Laplace parameters (mean and scale)

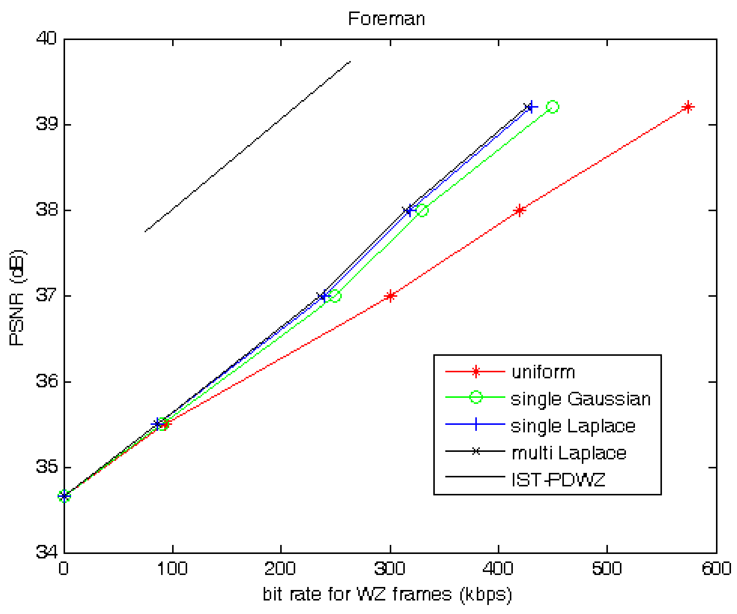
In this experiment, the sequences we use are of QCIF resolution, and only the first 101 frames are tested. Key frames are not compressed, in other words, they are lossless. The motion compensated interpolation procedure similar to that proposed in IST-PDWZ [38] with key frame smoothing and fine-tuning of bi-directional motion vectors is used to increase side information quality. The side information PSNR values using different tool combinations are listed in Table 1. Based on Table 1, tool set (d) is used in the following experiments. We calculate the R-D curves for W-Z frames and see how channel noise model chosen affects coding performance of side information prediction errors.

Table 1. PSNR of side information using different prediction tools

	a	b	c	d	e	f	G	h
Foreman	30.94	30.111	34.3677	34.515	32.7166	35.2435	35.7557	35.1935
Mother And Daughter	36.817	36.9148	39.43	39.6368	40.9836	41.5151	41.4215	44.7172
Coastguard	33.8926	33.9785	35.866	36	36.2894	34.3556	33.5139	33.8707

(a) Our Method
(b) Our Method + nearest mv for hole
(c) Our Method + Bi-directional mv adjust
(d) Our Method + nearst mv for hole +Bi-directional mv adjust
(e) Our Method + nearest mv for hole +Bi-directional mv adjust + low-passed key frame
(f) Our Method + nearest mv for hole +Bi-directional mv adjust + low-passed key frame + median mv
(g) Our Method + nearest mv for hole +Bi-directional mv adjust + median mv
(h) Our Method + nearest mv for hole +Bi-directional mv adjust + low-passed key frame + median mv (using non-low-passed key frame)

One can see in Figure 7, for both sequences, at low bit-rate, the differences between the best and the worst performance is less than 0.1 dB. At higher bit-rate, the difference is usually less than 1dB. Also note that, for a QCIF sequence, 1dB gain of PSNR above 40dB is virtually not perceptible by human observer under normal viewing conditions. It means that for most significant bit-plane, choosing Laplace or Gaussian model dose not affect performance much in practical sense.



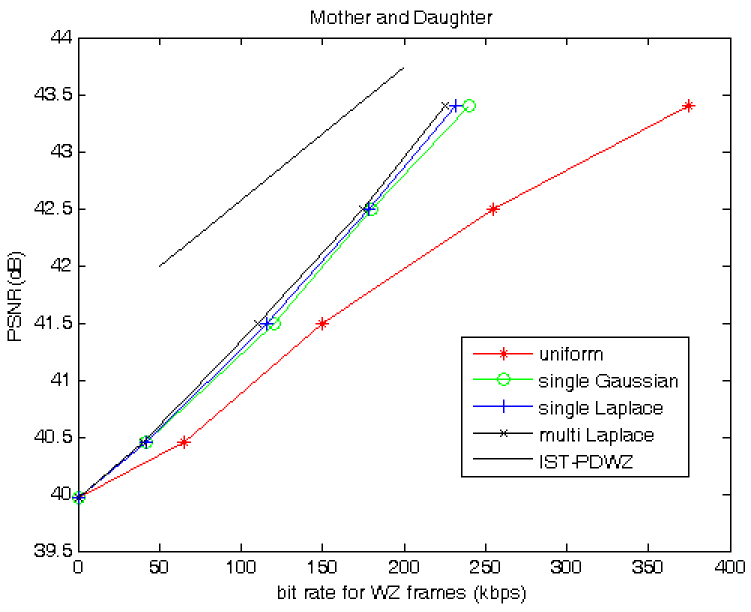
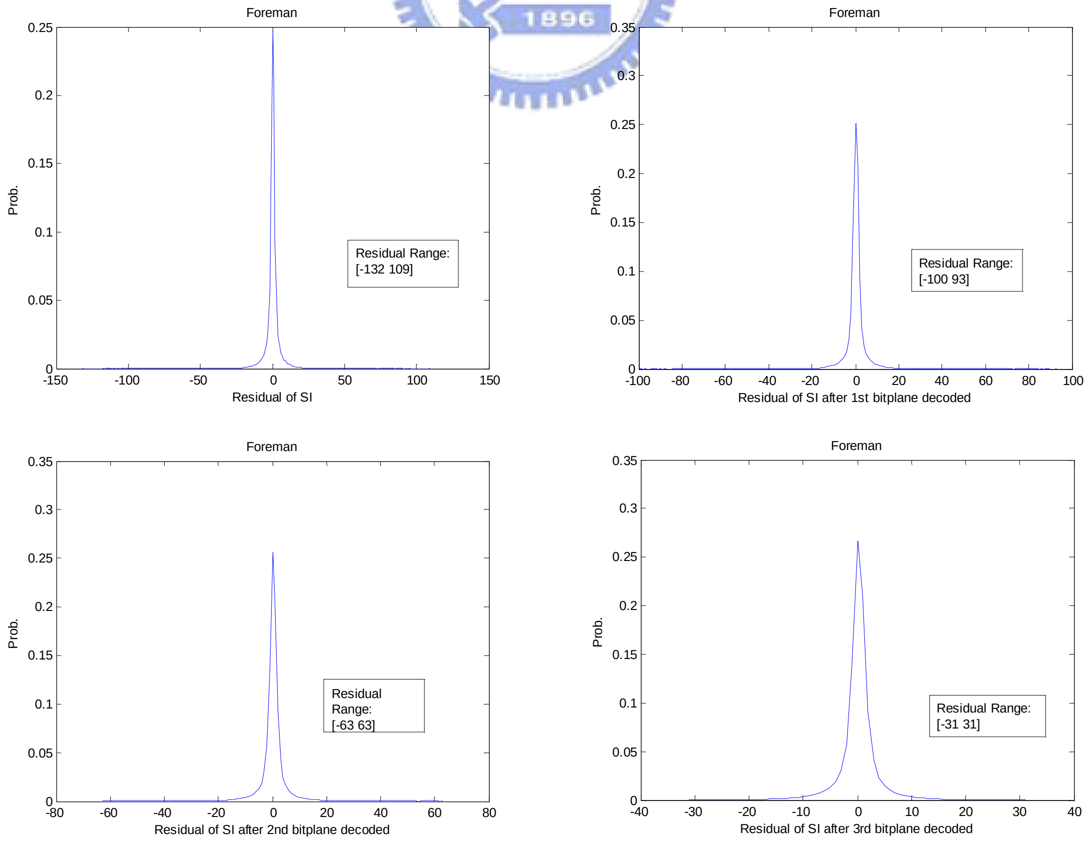


Figure 7. R-D curve of 'Foreman' and 'Mother and Daughter'

The results from IST-PDWZ are also projected in Figure 7. Although the final PSNR of our corrected side information is worse than those published in [38] for about 2.5 dB, the slopes of our R-D curves are the same or even larger than the slopes of the curves of IST-PDWZ. The difference in PSNR is largely due to the initial quality of side information before error correction. The differences in slopes may be due to different channel coding techniques (LDPCA in our experiments, and turbo coder in IST-PDWZ) and it shows that the channel coding efficiency of LDPCA may be better than that of turbo code for DVC.

3.1.2 Adaptive Laplace Channel Model for Each Bit-planes

In previous section we see that the channel behavior-based error model only has marginal influence on DVC coding efficiency. One of the possible reasons is due to the fact that side information errors are not spatially invariant. Therefore, single error channel model is not sufficient for modeling the entire sequence. But in previous graph, we can see that even by choosing different parameters for each small area adaptively, the performance increase is not obviously. Other possible reasons are that single error model is used for every bit-plane decoding, or that the channel decoding method used in many DVC papers is not suitable for DVC.



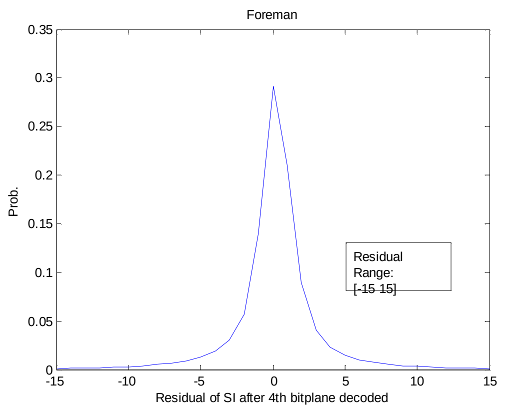


Figure 8. Error distributions when 1st ~ 4th bit-planes are decoded

We can see how the error distribution changes as more bit-planes are decoded in Figure 8. When more bit-planes are decoded, errors are more concentrated and their means are more closed to 0. Now we try to use different Laplace models for each bit-plane. The R-D curve is shown in Figure 9. We can see that the performance is worse than using one distribution parameter value for all bit-plane. It is even worse than using a single Gaussian model for all bit-planes when bitrate exceeds 350 kbps. It may be because that when using different Laplace models for each bit-plane, the Laplace model for less significant bit-plane will be more concentrated because the error distribution of SI when decoding less significant bit-plane is more concentrated. However, actually when decoding less significant bit-plane, the probability of bit error is larger. So, the error model does not fit the actual error distribution, and it decreases the performance.

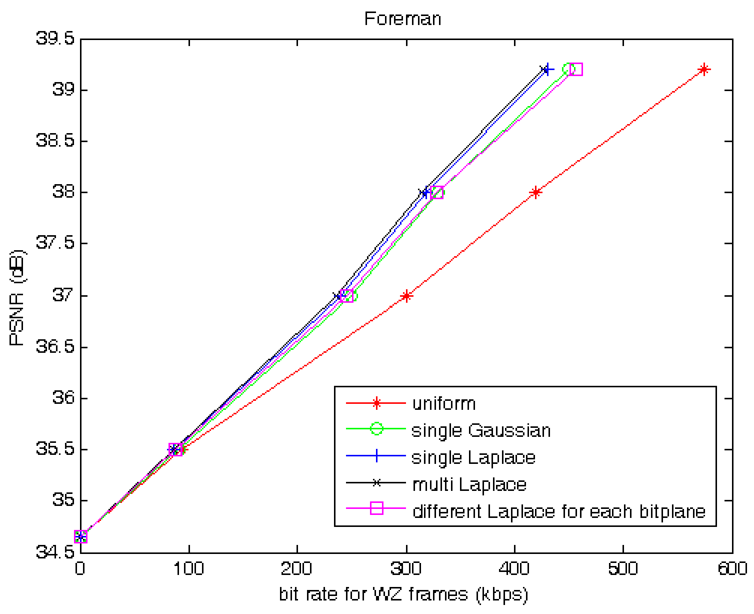


Figure 9. R-D curve of 'Foreman' when different Laplace models are used for each bit-plane

3.2 LDPCA Decoding

3.2.1 LLR (Log Likely-hood Ratio) Values

The LDPC decoding of each bit x in a bitplane of the W-Z frame is based on the LLR (log likely-hood ratio). The relationship of P(x=0) and the corresponding LLR values are shown in Figure 10. When the channel error is small, P(x=0) will not be closed to 0.5, and absolute value of LLR will be far from 0. In this situation, fewer W-Z bits will be necessary to correct the errors in side information. So, in this section, we investigate the LLR values of every pixel in the Foreman sequence to estimate the number of W-Z bits required for error correction.

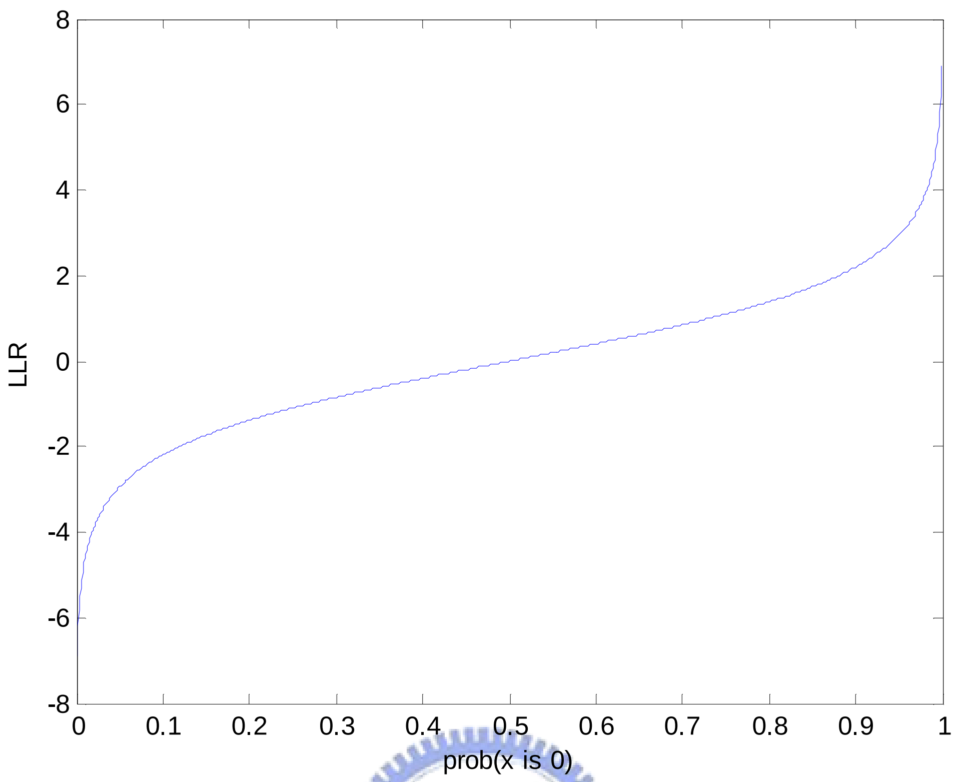


Figure 10. **P(x = 0) and corresponding LLR values**

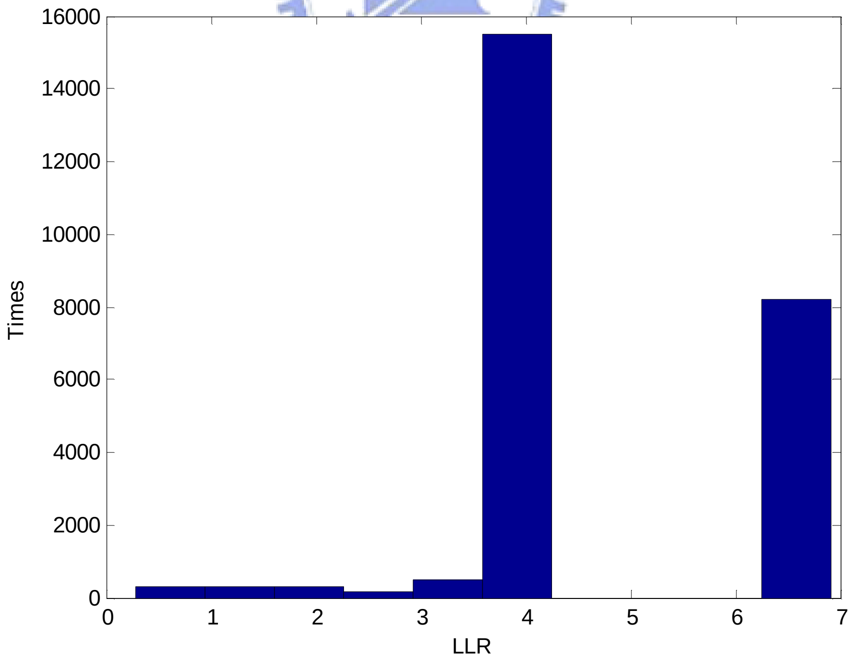


Figure 11. **Absolute values of LLR when decoding 1st bit-plane of 'Foreman' 2nd frame**

We plot the histogram of absolute values of LLR when decoding 1st bit-plane of the 2nd frame of the 'Foreman' sequence in Figure 11. Most of the values are around

3.89 and 6.91. We want to see the amount of pixels with small absolute values of LLR, because when the absolute values of LLR are small, it means these pixels are considered noisy. We discover that these pixels are all with side information pixel values near 127 (from 124 to 133). 127 is the threshold between 0 and 1 in the 1st bit-plane. We show the scatter plot of side information value and absolute value of LLR when decoding 1st bit-plane of 'Foreman' 2nd frame in Figure 12. The scatter plots for 2nd and 3rd bit-planes are in Figure 13 and Figure 14. We can see that pixels with small LLR values are all with side information values near the threshold of particular bit is '0' and '1'. We call this threshold 'crossing threshold.'

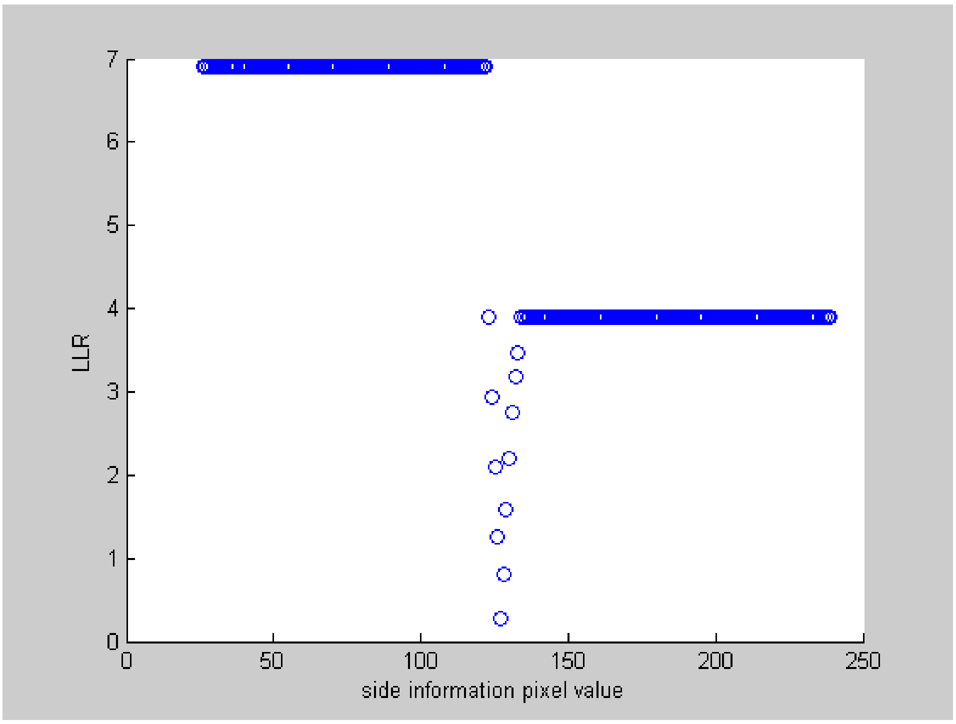


Figure 12. Absolute value of LLR and pixel value of side information when decoding 1st bit-plane of 'Foreman' 2nd frame

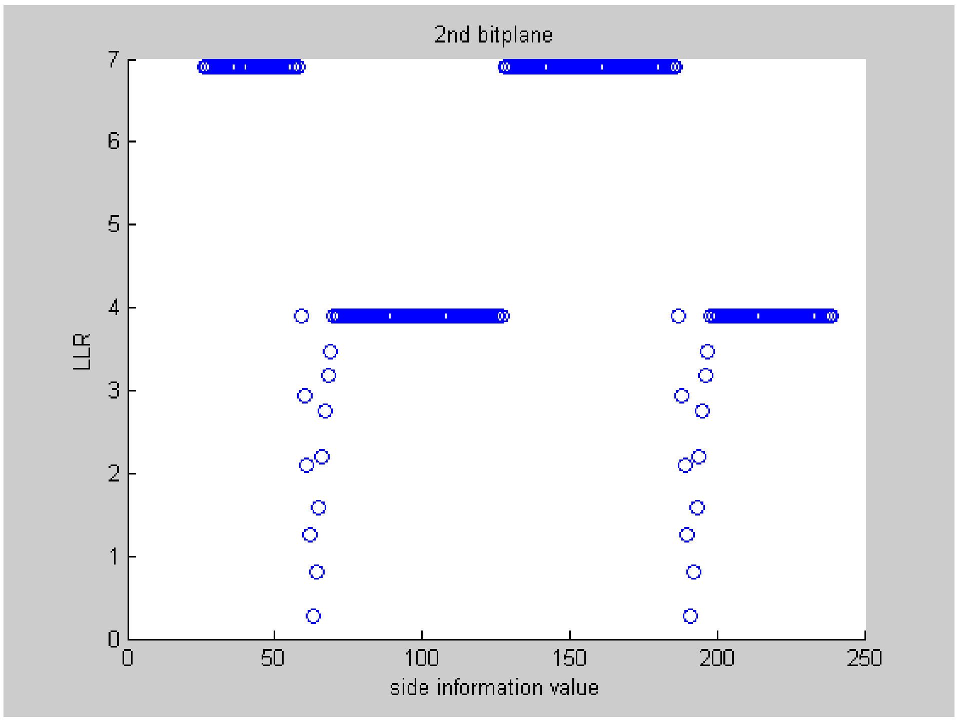


Figure 13. **Absolute value of LLR and pixel value of side information when decoding 2nd bit-plane of 'Foreman' 2nd frame**

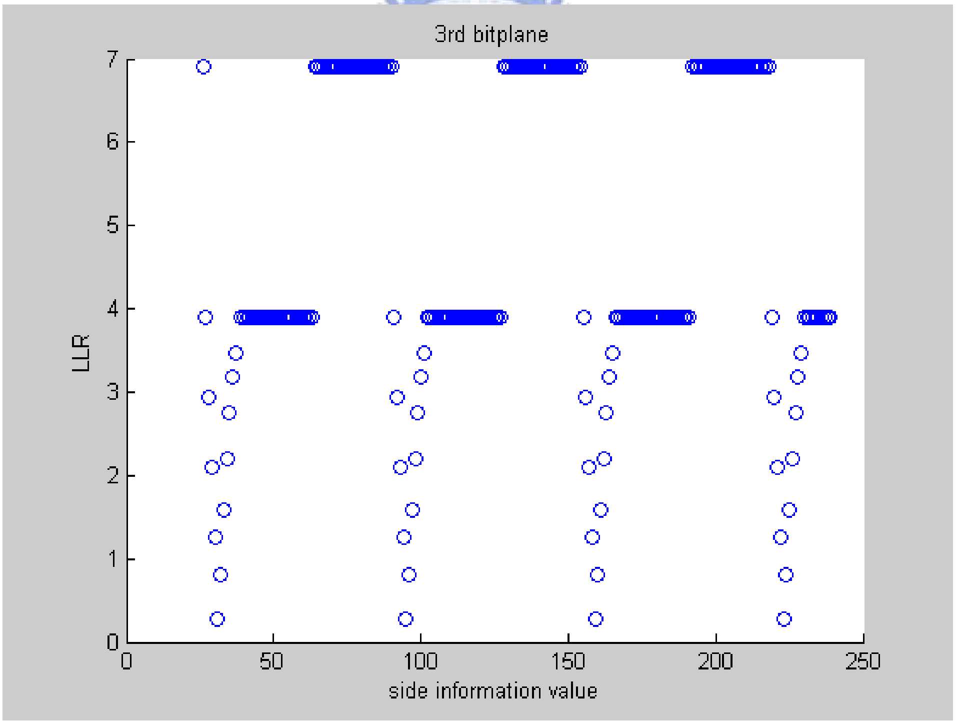


Figure 14. **Absolute value of LLR and pixel value of side information when decoding 2nd bit-plane of 'Foreman' 2nd frame**

Larger absolute values of LLR means less correction bits are needed, but actually it is not certain that their errors are small. Bitrate may be wasted for these pixels.

Those pixels with large absolute LLR but large errors are called under-corrected pixels, and pixels with small absolute LLR but small errors are called over-corrected pixels.

For 2nd frame of foreman sequence, we set the threshold of side information error at 5 and 25. So when side information is large than 25, we call it "Large error", while side information is smaller than 5, we call it "Small error." We also set the threshold of |LLR| at 3.89. These thresholds are decided based on the distribution of side informaion and LLR values of this particular frame. After the thresholds are decided, we can calculate the percentage of over-corrected pixels and under-corrected pixels and the result is in Table 2. When more bitplanes are decoded, the percentage of over-corrected pixels is also increased. Although there is only 10% of pixels are over-corrected, we want to see how these pixels affect the performance.

Table 2. Percentage of over-corrected pixels and under-corrected pixels of foreman 2nd frame

	Over-corrected	Under-corrected
1st bitplane	5.56%	0.6%
2nd bitplane	6.03%	0.68%
3rd bitplane	10.3%	2.13%

3.2.2 Over-Corrected Pixels

An experiment is conducted to see how these over-corrected pixels affect bitrate. To achieve this, we peek at pixel values of the original WZ frames. For pixels whose side information is near 'crossing threshold' and error smaller than 5, we assign side information pixel values to the original frame pixels and do LDPCA encoding and decoding. Here we change LLR of these pixels to 3.89 or -3.89. In this experiment, the key frames are encoded using H.264 intra encoder with QP 25. The two R-D curves are shown in Figure 15. One of them is the original R-D curve. Another one is the R-D curve when we ignore pixels with side information near crossing threshold and small residuals. It represents the ideal situation where for all pixels with small residual, the side information values are not near crossing threshold. From the experimental results, we can see that the value of side information is impartment and affects bitrate much although there is only about 10% of pixels are modified. As bitrate increases, the R-D performance difference increases.

But this experiment could not happen in real world because the encoder cannot detect which pixels need to be modified. A practical solution is for the decoder to ignore pixels near ‘crossing threshold’ when it performs LDPCA decoding. So the encoder does not have to detect and modify these pixels and re-encode them. In this experiment, the decoder set LLR values of these pixels to 3.89 or -3.89. These pixels are not modified after each bit-plane is decoded, and when the LDPCA decoder tests convergence or checks syndromes, these pixels are ignored. Here we use ‘Carphone’ sequence to test, and the R-D curve is in Figure 16.

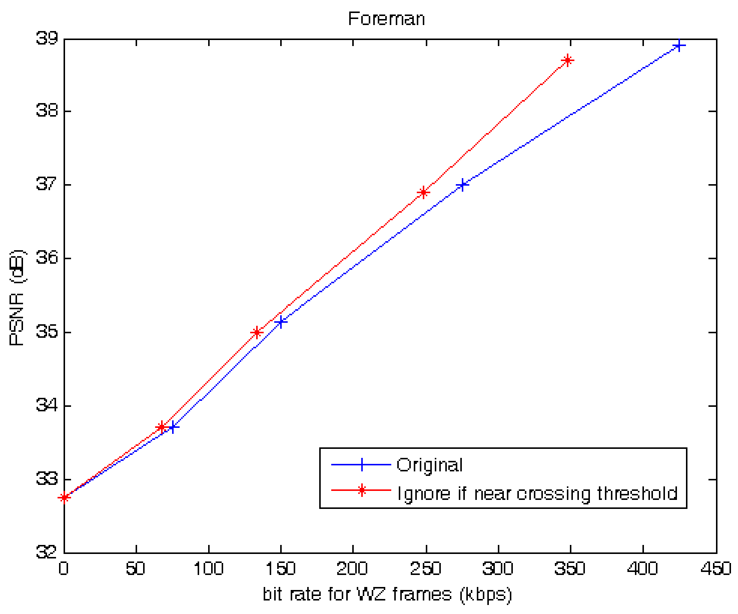


Figure 15. R-D performance when ignoring pixels with side information near 'crossing threshold' and small residual

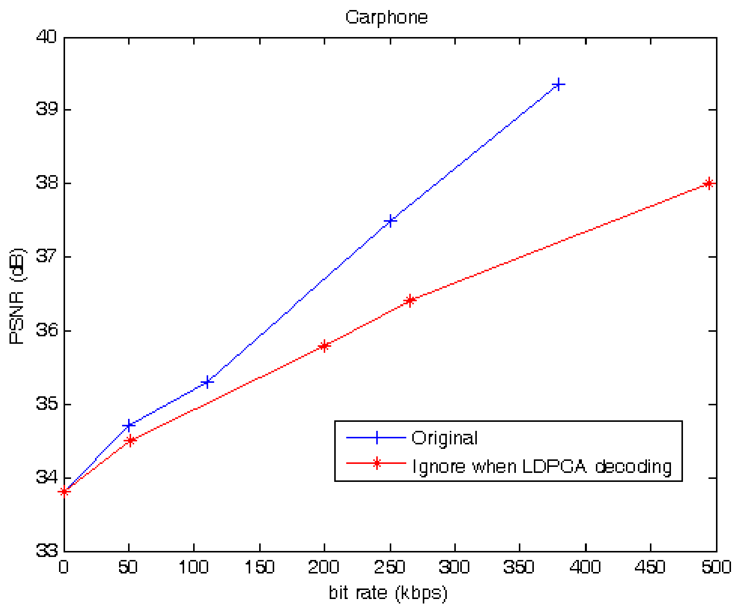


Figure 16. R-D performance when decoder ignore pixels with side information values near 'crossing threshold'

We can see that the performance is worse when decoder ignore pixels with side information values near 'crossing threshold.' This is because there are too many syndromes not checked because they contain information of ignored pixels. So, when

LDPCA iterative termination condition is reached, it finishes decoding with errors, and more syndrome bits are requested.

3.2.3 Prioritized Decoding of Side Information Pixels

Based on the previous experiments, we propose a way to perform prioritized correction of side information. We can divide pixels into two groups, and the two groups are in different LDPCA coding blocks so we can deal with them separately. In order to implement this approach, there are some key issues which need to be resolved. First, how can the DVC codec classify the side information into pixels into two groups? Secondly, how to encode syndrome bits of different groups of pixels separately at the encoder?

The decoder can see the side information; it can partition pixels into two groups and send the group information back to the encoder. If the group information is at pixel level, then the amount of bits used to describe group information would be too high. Therefore, we compute the group information at macroblock level. For each macroblock in a frame, it will be classified into one of two groups.

When the encoder receives group information, it will rearrange macroblocks in a frame. The macroblocks belong to the same group will be channel-coded together, so they will be LDPCA encoded within a block. This way, the LDPCA syndrome bits can be spent on the area where the residual values are large, and errors are corrected more efficiently. In chapter 4 we will describe the proposed method of classifying and re-ordering macroblocks in detail. The experimental results will be given in chapter 5.

Chapter 4: System Architecture

In this chapter, the proposed DVC codec will be described in details. Our DVC codec is implemented in C. Macroblocks in a frame will be classified into different groups according to their side information quality. Each group of macroblocks will be LDPCA-coded in the same coding block. So we can encode and decode each group of macroblocks according to their significance (priority) in R-D improvement. The prioritized channel decoding will enhance the performance because WZ bits are requested more efficiently.

We have implemented both pixel domain and frequency domain DVC systems. In the following sections, we will use frequency domain DVC codec to explain the proposed scheme, and then the difference between pixel and frequency domain approach will be described.

4.1 System Block Diagram

The system block diagram of the proposed frequency domain DVC codec is shown in Figure 17. First, a sequence will be divided into key frames and WZ frames. Odd frames are key frames and even frames are WZ frames. Key frames are encoded and decoded using H.264 main profile intra coder. The version of H.264 codec we use is JM 9.0. The QP values are determined based on quantization matrices used for WZ frames.

The decoder uses key frames it has received to generate side information of WZ frame. Then it will classify macroblocks into two groups, S_A and S_B, based on side information quality. S_A contains 25% macroblocks with worse side information quality and S_B contains 75% macroblocks with less side information error. The decoder does not see the original WZ frame, so it can only classify macroblocks based on some cues available. The classifying result will be sent to the encoder. And then the encoder will group macroblocks in WZ frame while the decoder will group macroblocks in side information according to the classifying result. Macroblocks in same group will be gathered together.

Macroblocks in WZ frame and side information are then transformed and quantized by encoder and decoder. The encoder will calculate quantization interval for each band and send the values to the decoder. So the encoder and decoder use the same quantization interval to quantize coefficients.

The quantized coefficients of WZ frame and side information are then split into bit-planes. Each bit-plane of a WZ frame will be LDPCA encoded by the encoder. The WZ bits are stored in a buffer and they will be requested by the decoder.

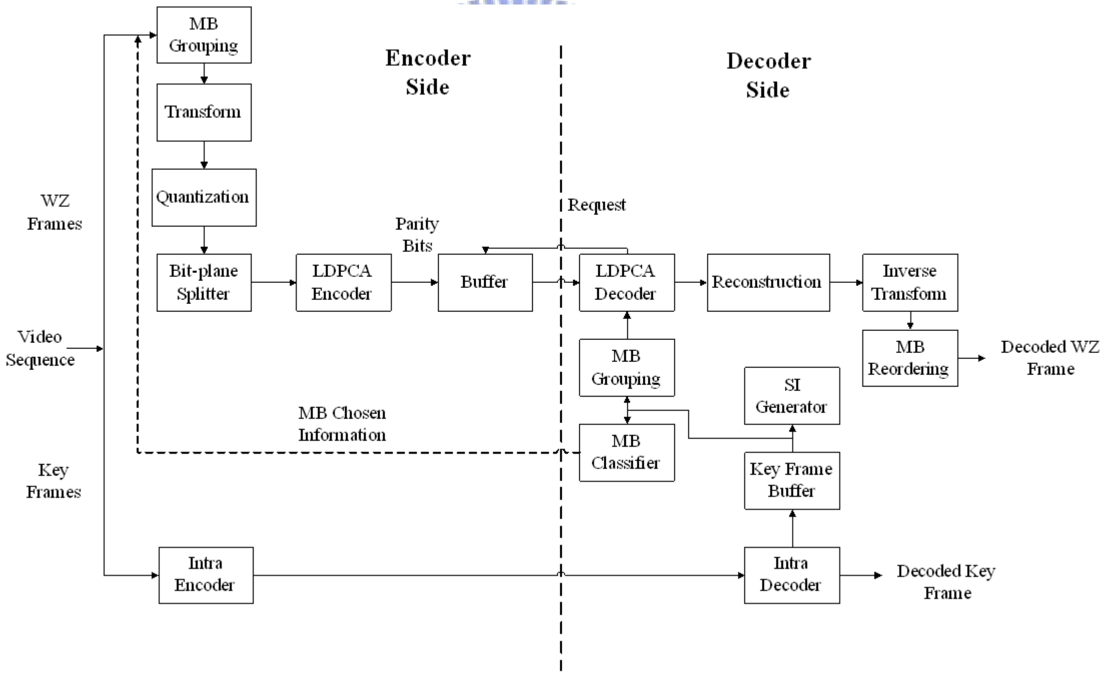


Figure 17. System flow of our transform domain DVC codec

The decoder will request for WZ bits in the buffer and perform LDPCA decoding, and the detail will be described later. Only macroblocks in S_A are decoded. After every bitplanes of every coefficient bands are decoded, these decoded coefficients will be inverse transformed and macroblocks will be rearranged to original order. The following sections will describe the details in each step of the proposed algorithm.

4.2 Side Information Generation

As we describe above, key frames are coded by H.264 intra coder and sent to the decoder. The decoder uses neighboring key frames to interpolate side information of center WZ frame.

Steps of side information generation are described in this section.

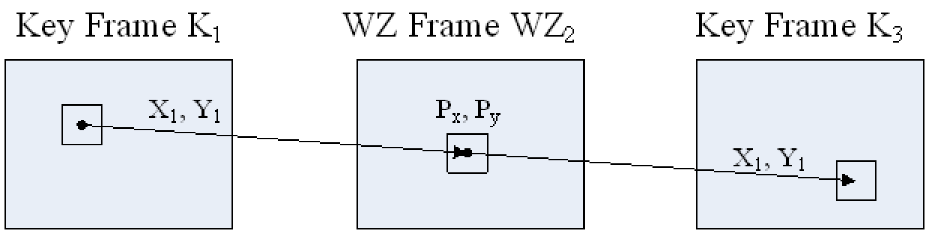


Figure 18. Motion estimation for neighboring key frames

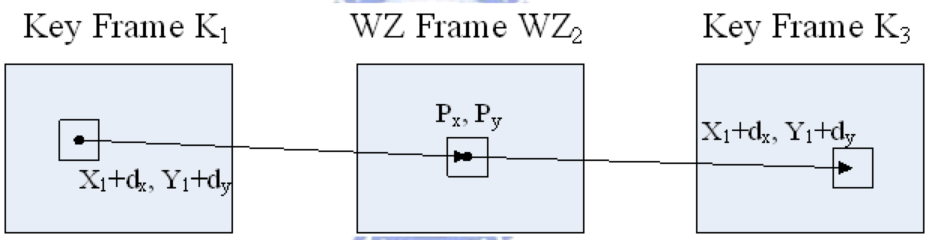


Figure 19. Bi-directional motion adjustment

In step one, motion estimation is performed for two neighboring key frames, as shown in Figure 18. Size of macroblock is 16 by 16, search range is ±32, and motion vector accuracy is at is full pixel precision. The search range is larger than traditional video codec because the time distance between key frames is 2. In this step, we only use forward motion estimation to guess the motion field of WZ frame. There are many works we can do in order to make the motion field more closed to true motion.

In step two, refer to DISCOVER's DVC codec[18], a bi-directional motion adjustment is performed, and the search range is ±10 with half pixel precision. The half pixel values are calculated using H.264 six-tap filter. The search range is smaller so the adjusted motion vector will not be very far away from original motion vector.

As in Figure 19, if the motion vector obtained in step one motion estimation is (X_1, Y_1) and the center of macroblock in WZ frame is (P_x, P_y), then when performing bi-directional motion adjustment, the center is fixed and motion vectors (X_1+d_x, Y_1+d_y) with d_x and d_y within range ±10 are searched. The new motion vector (X', Y') will make SAD (sum of absolute difference) value of macroblock pair in neighboring key frames smallest. Every motion vectors obtained in step one will be modified in this step.

In step three, median filter is applied in order to smooth the estimated motion field. This step is also suggested by the DISCOVER DVC codec [16]. After motion estimation for neighboring key frames and bi-directional motion adjustment, for each macroblock, its motion vector and the motion vectors of eight-connected macroblock neighbors are listed. Then a median motion vector is obtained among these motion vectors. When deciding which one of them is the median, a weight for each motion vector is used.

There are eight neighbors with motion vectors m_1 to m_8, and motion vector of the center macroblock is m_0. For the center macroblock, the motion vector m_0 will point to two macroblocks in neighboring key frames, and SAD of these two macroblocks is s_0. When m_0 is replaced by m_1 to m_8, the SAD of neighboring two macroblocks will be s_1 to s_8. The weight value w_i of the neighbor motion vector m_i is defined as s_0/s_i. So, if m_i makes neighboring macroblocks similar, then its weight is larger. After median motion vector is obtained, the motion vector will be replaced with this median motion vector. Now the motion field of WZ frame is obtained and side information will be interpolated based on this motion field.

Average of macroblocks in neighboring frames is interpolated to generate side information. The SAD values of macroblock in neighboring key frames are recorded for macroblock classification later. Now, there are some pixels which have more than one projections from neighboring key frames. For such pixels, the average of interpolated values of all projections is used as the side information. Some pixels do not have projections at all, and these pixels remain unfilled as holes in the side information..

Two hole-filling procedures are applied to complete the side information. For the first procedure, if the Manhattan distance between the hole and nearest filled pixel is within 25 pixels, then the motion vector of this filled pixel is used by the hole for motion compensation from neighboring key frames. Otherwise, another hole-filling procedure is applied. Distance upper bound is chosen as 25 pixels because we do not want to use motion vectors of pixels too far away.

The remaining holes will be filled by the second hole-filling procedure. Now for each macroblock in side information, calculate the percentage of holes. If the percentage of holes is less than 40 percent, then motion estimation for this macroblock and previous key frame in display order will be performed. Only filled pixels are used to calculate SAD, and we achieve this by using a mask to ignore difference values at holes when calculating SAD. The macroblock size is 16 by 16 and the search range is ±32. The macroblock with smallest SAD in previous key frame is located and the corresponding pixels in this macroblock will be used to fill the holes in side information.

If the percentage of hole is larger than 40 percent, then the size of macroblock will be enlarged by 2 each time until it reaches 32 by 32. The percentage of holes is 40 percent at most because when there are too many holes in one macroblock, then the valid pixels used to find motion will be minority. Thus an incorrect motion vector will be obtained. The size of macroblock used to find motion vector for holes can not be too large, too. When the macroblock size is too large, the results of motion estimation will be bad because pixels within one macroblock in practice have different motions.

4.3 Macroblock Classification and Grouping

Given the generated side information, the decoder will classify macroblocks into groups S_A and S_B. S_A contains 25% macroblocks with worse side information quality and S_B contains 75% macroblocks with less side information error. For QCIF resolution sequences, there are 24 macroblocks in group S_A and there are total 99 macroblocks. And this makes each bit-plane of each coefficient band of group S_A contains 384 bits (24 macroblocks are equal to 384 4 by 4 blocks,) and the LDPCA block size we use when only group S_A is decoded is 396. If we pick up 25 macroblocks for group S_A, then each bit-plane of each coefficient band of group S_A will contain 400 bits and this is larger than 396.

Because the decoder does not have the original WZ frames, therefore it must estimate the quality of side information in order to classify macroblocks. Many features of the side information image, such as the motion field variance, the number of edges or corner points, and the SAD values of macroblocks in neighboring key frames, can be used as estimates for classification.

To obtain motion field variance, we need dense motion field. When generating side information, motion vectors of macroblock size 16 by 16 is generated. And then a 16 by 16 macroblocks is divided to four 8 by 8 blocks. Start from motion field already obtained, motion estimation similar to bi-direction adjustment above is performed and the search range is ±10. So, for each macroblock, there will be four motion vectors, and we get a dense motion field. We can further divide each 8 by 8 macroblock to four 4 by 4 macroblocks and get a more dense motion field. After dense motion field is obtained, motion vector length variance of every macroblock is calculated and this variance is used to classify macroblocks. When the variance is larger, we think the side information quality of this macroblock is worse.

To obtain number of edges and corner points, we use Sobel filter to get edges and use Harris corner detector to get corner points. The Sobel filter size is 3 by 3, and the setting of Harris corner detector is radius 3, sigma 1, and threshold 10 for all test sequences. The values of radius and sigma are recommended by the author of Harris corner detector and the threshold is set as 10 so there will not be too many annoying corner points. When there are more corner points or edges in a macroblock, we think the side information quality is worse. We use these features as cues because we have observed that for Foreman sequence, macroblocks containing these features have worse side information quality.

To obtain SAD of macroblocks in neighboring key frames, we do not do additional works because we obtain these values when generating side information. When SAD of macroblocks in neighboring key is larger, we think the quality of side information is worse. And this is because side information quality is proportioned to SAD of macroblocks for many pixels statistically, especially when motion is well guessed.

After trying these cues, we have discovered empirically that SAD of macroblocks in neighboring key frames is a good cue for the decoder to pick up worse macroblocks, as shown in Figure 21. Several sequences are tested in pixel domain with lossless key frames, and the optimal R-D curve is the case where we take a peek at the original WZ frame and choose macroblocks that have the worst side information.

The decoder will use 99 bits to descript classifying result and the result will be sent to the encoder. Bit ‘0’ represents group S_A and bit ‘1’ represents group S_B. The decoder will rearrange macroblocks in the side information and macroblocks of same group will be gathered together. The encoder will also rearrange macroblocks in WZ frame in the same way.

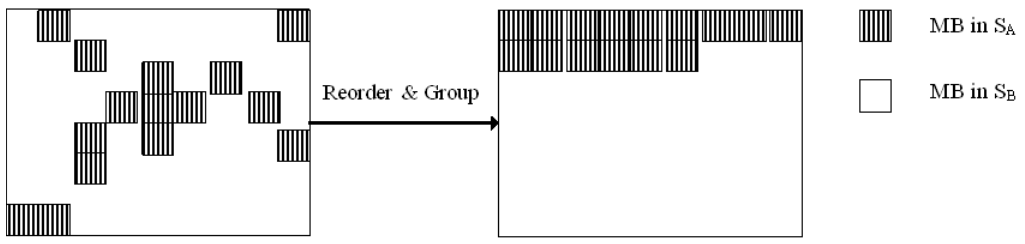
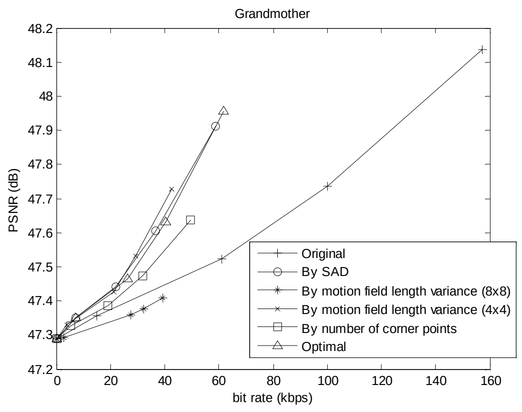
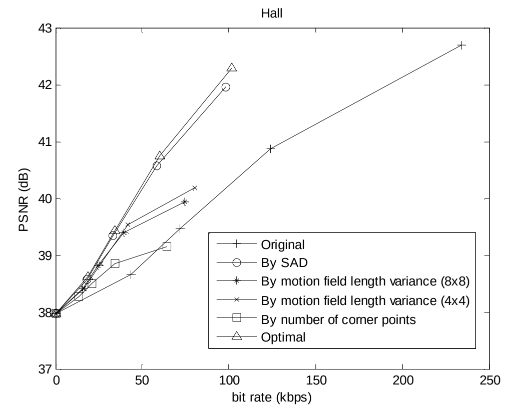
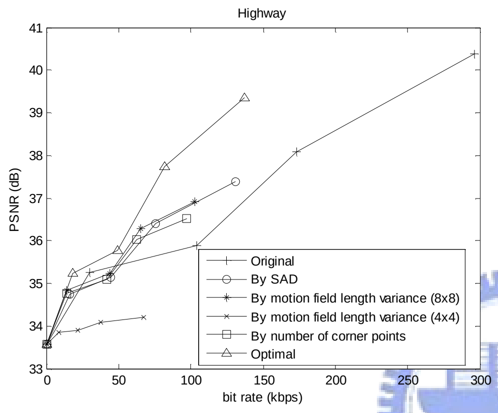


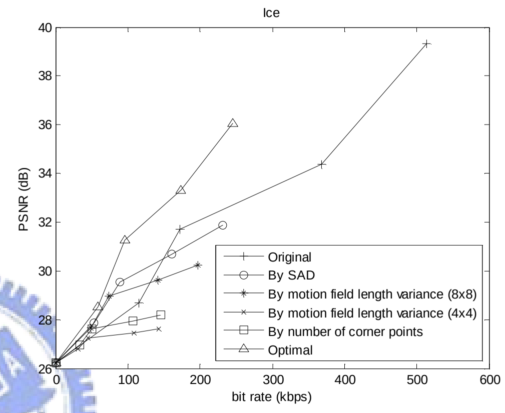
Figure 20. Macroblock rearrange

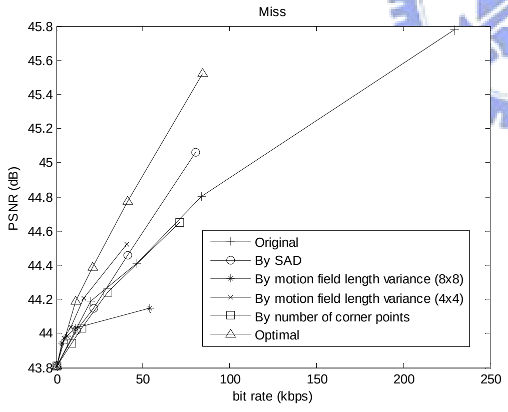
The macroblocks of group S_A are placed at top of the frame, and macroblocks of group S_B are placed at bottom of the frame, as shown in Figure 20. The order of macroblocks of same group is maintained in scan-line order. So if there are 2 macroblocks of group S_A, m_1 and m_2, and m_1 precedes m_2 before the rearrangement. Then m_1 will precede m_2 after the rearrangement. After macroblock reordering and grouping, macroblocks of the same group will be packed in the same LDPCA block for W-Z frame coding.

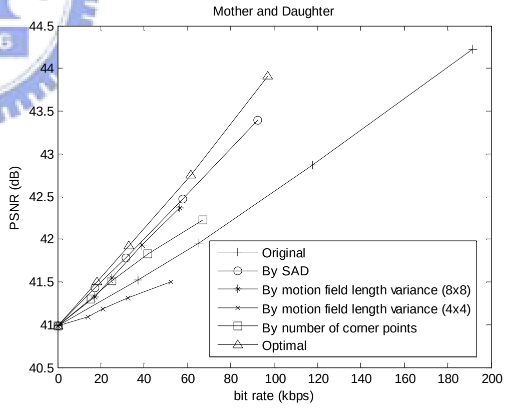

Grandmother
PSNR (dB)
bit rate (kbps)
48.2 48.1 48 47.9 47.8 47.7 47.6 47.5 47.4 47.3 47.2
0 20 40 60 80 100 120 140 160
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

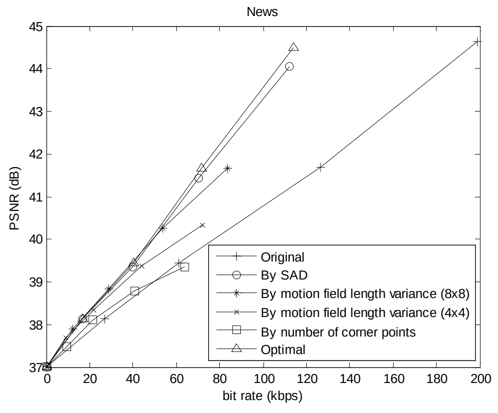

Hall
PSNR (dB)
bit rate (kbps)
43 42 41 40 39 38 37
0 50 100 150 200 250
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

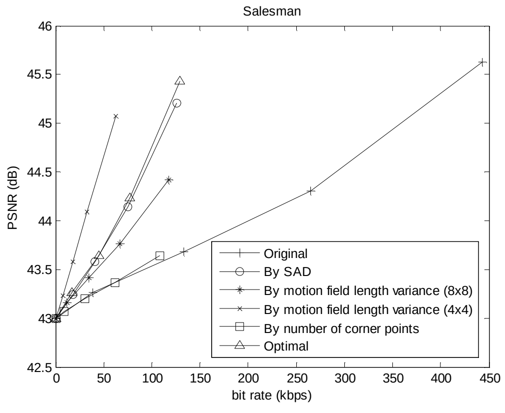

Highway
PSNR (dB)
bit rate (kbps)
41 40 39 38 37 36 35 34 33
0 50 100 150 200 250 300
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

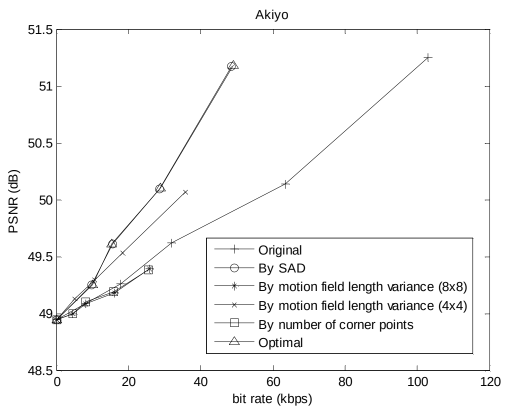
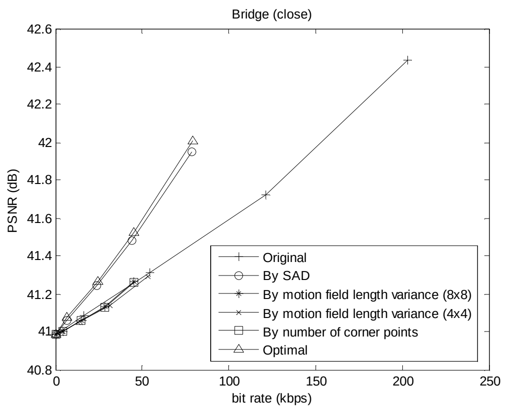
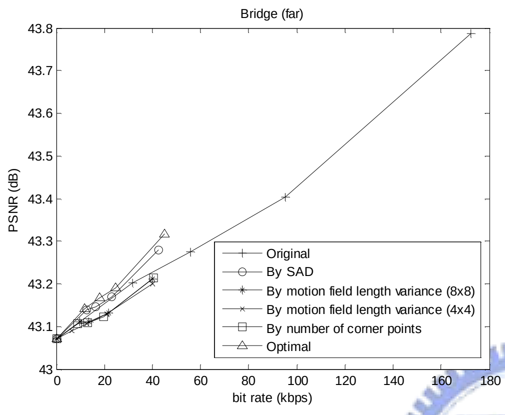
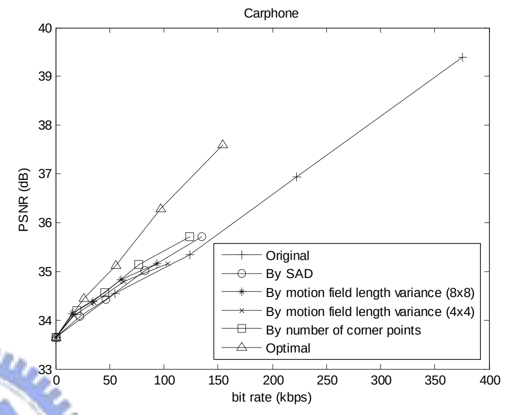
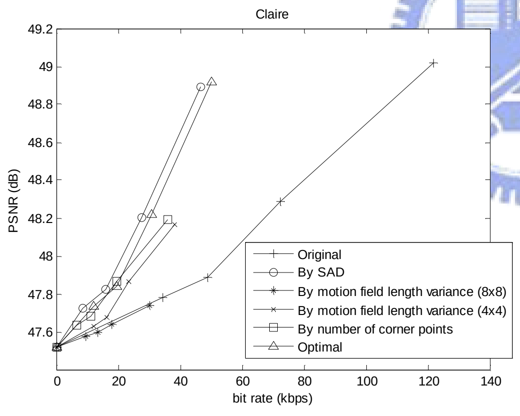
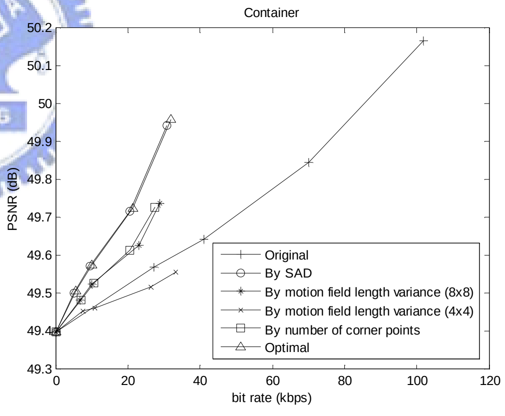
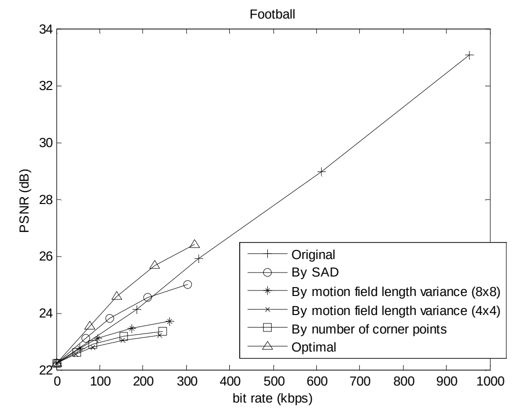
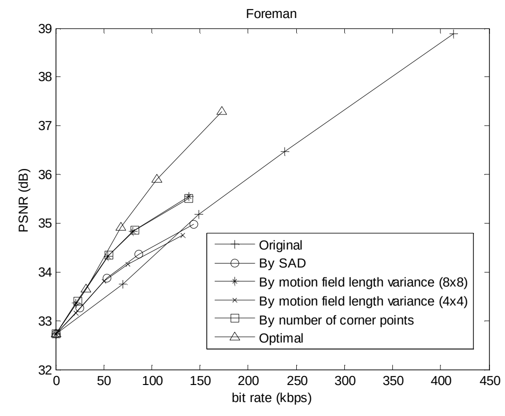

Ice
PSNR (dB)
bit rate (kbps)
40 38 36 34 32 30 28 26
0 100 200 300 400 500 600
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal


Miss
PSNR (dB)
bit rate (kbps)
45.8 45.6 45.4 45.2 45 44.8 44.6 44.4 44.2 44 43.8
0 50 100 150 200 250
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal


Mother and Daughter
PSNR (dB)
bit rate (kbps)
44.5 44 43.5 43 42.5 42 41.5 41 40.5
0 20 40 60 80 100 120 140 160 180 200
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal


News
PSNR (dB)
bit rate (kbps)
45 44 43 42 41 40 39 38 37
0 20 40 60 80 100 120 140 160 180 200
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal


Salesman
PSNR (dB)
bit rate (kbps)
46 45.5 45 44.5 44 43.5 43 42.5
0 50 100 150 200 250 300 350 400 450
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal


Akiyo
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Bridge (close)
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Bridge (far)
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Carphone
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Claire
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Container
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Football
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

Foreman
PSNR (dB)
bit rate (kbps)
Original
By SAD
By motion field length variance (8x8)
By motion field length variance (4x4)
By number of corner points
Optimal

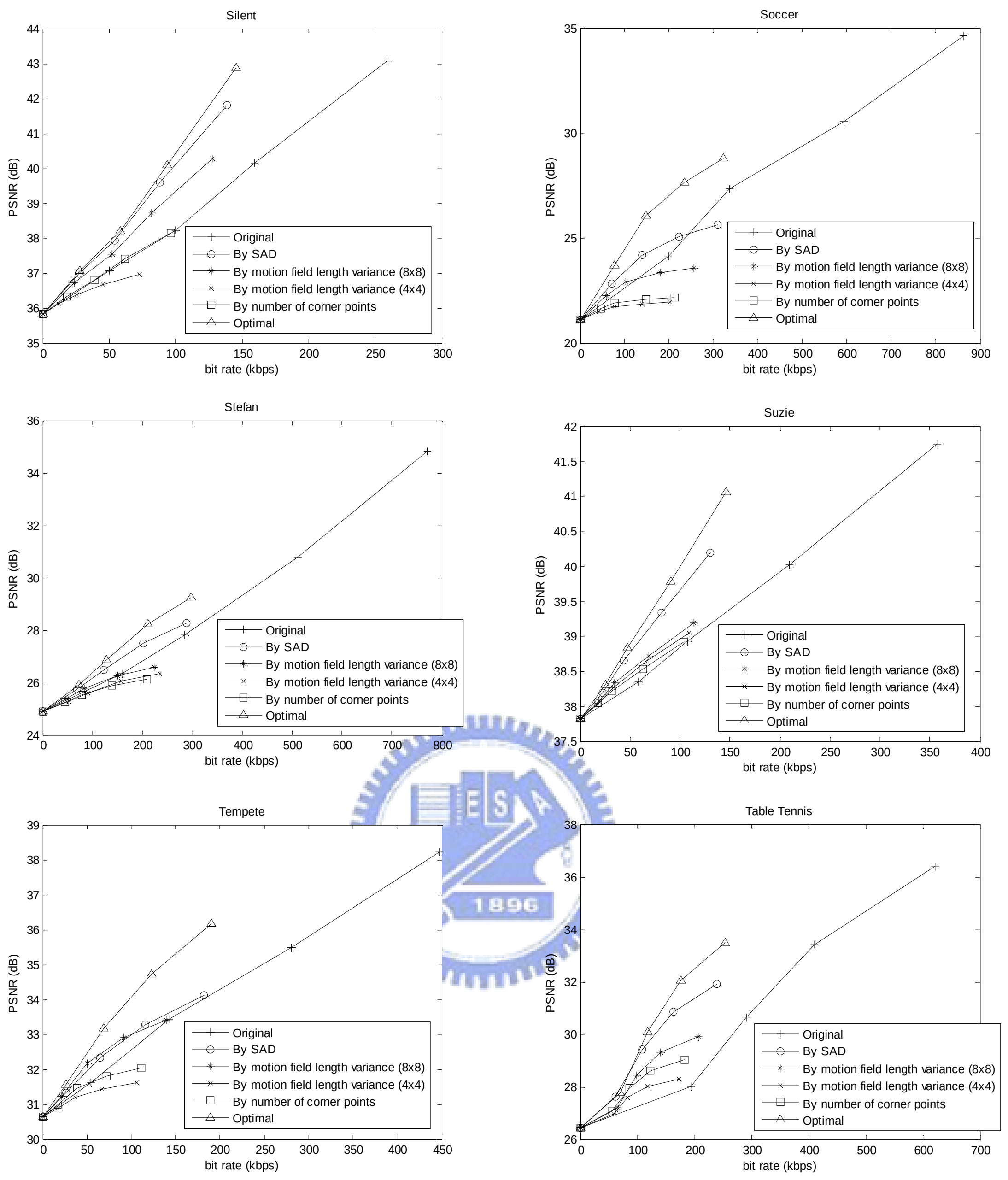
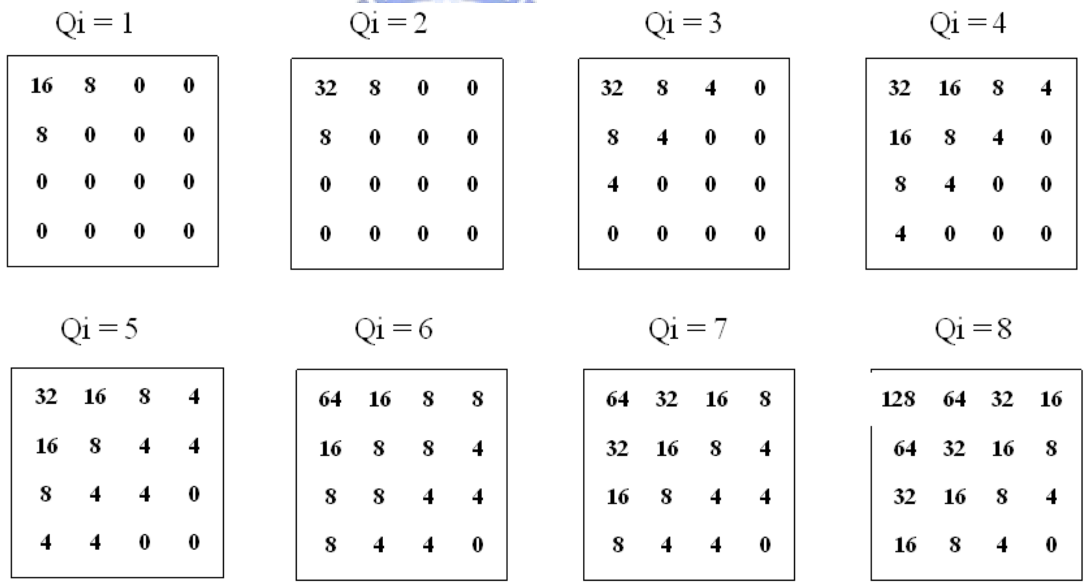


Figure 21. R-D performance when different cues are used by the decoder to pick up worse macroblocks

4.4 Transform and Quantize

At the encoder side, after macroblock rearrangement, each WZ frame will be transformed to frequency domain. At the decoder side, side information is also transformed too. The integer transformation of H.264 is used. The block size for the transformation is 4 by 4. For each 4 by 4 block, after transformation, there will be 16 coefficients, b_1 to b_{16}. B_1 is the DC band and b_2 to b_{16} are AC bands.

The coefficients of each band will then be quantized. There are 8 quantization matrices, Q_1 to Q_8, which are proposed by the Stanford DVC team except Q_7 by DISCOVER. These 8 quantization matrices are listed in Figure 22. Numbers in these matrices represent the number of quantization levels for coefficients of corresponding band. Matrix Q_k with larger k will introduce less quantization error because more bits are used to represent coefficient values. QP values for encoding key frames are chosen according to quantization matrices used. The DISCOVER team has proposed a set of QP values which make the decoded WZ frames and key frames have almost same quality. The QP values for Foreman, Hall, Coastguard, and Soccer sequences are listed in Table 3.



$Qi = 1$: $\begin{bmatrix} 16 & 8 & 0 & 0 \\ 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ $Qi = 2$: $\begin{bmatrix} 32 & 8 & 0 & 0 \\ 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ $Qi = 3$: $\begin{bmatrix} 32 & 8 & 4 & 0 \\ 8 & 4 & 0 & 0 \\ 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ $Qi = 4$: $\begin{bmatrix} 32 & 16 & 8 & 4 \\ 16 & 8 & 4 & 0 \\ 8 & 4 & 0 & 0 \\ 4 & 0 & 0 & 0 \end{bmatrix}$

$Qi = 5$: $\begin{bmatrix} 32 & 16 & 8 & 4 \\ 16 & 8 & 4 & 4 \\ 8 & 4 & 4 & 0 \\ 4 & 4 & 0 & 0 \end{bmatrix}$ $Qi = 6$: $\begin{bmatrix} 64 & 16 & 8 & 8 \\ 16 & 8 & 8 & 4 \\ 8 & 8 & 4 & 4 \\ 8 & 4 & 4 & 0 \end{bmatrix}$ $Qi = 7$: $\begin{bmatrix} 64 & 32 & 16 & 8 \\ 32 & 16 & 8 & 4 \\ 16 & 8 & 4 & 4 \\ 8 & 4 & 4 & 0 \end{bmatrix}$ $Qi = 8$: $\begin{bmatrix} 128 & 64 & 32 & 16 \\ 64 & 32 & 16 & 8 \\ 32 & 16 & 8 & 4 \\ 16 & 8 & 4 & 0 \end{bmatrix}$

Figure 22. Quantization matrices

Table 3. QP Values for encoding key frames

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
Foreman	40	39	38	34	34	32	29	25
Hall Monitor	37	36	36	33	33	31	29	24
Coastguard	38	37	37	34	33	31	30	26
Soccer	44	43	41	36	36	34	31	25

The quantization we use is uniform quantization. The number of quantization levels is already determined by quantization matrices. But the quantization interval of each band is determined according to the range of coefficients. Before applying quantization, the quantization interval I_2 to I_{16} will be calculated by the encoder. If the coefficients of band b_k is within range $\pm R_k$ and number of quantization level is N_k, then the quantization interval for this band will be $2R_k/N_k$. The range of DC band is fixed as 0 to 1024, so the quantization interval of DC band is also fixed. So, fixing number of quantization interval, when the range is smaller, the quantization interval will be smaller too, and this will reduce quantization error. The calculated quantization interval for each band will be sent to the decoder. The decoder will transform and quantize side information using the same quantization interval.

After quantization, the coefficients will be split into bit-planes. The DC coefficients are all positive while AC coefficients may be positive or negative. When AC coefficients are split into bit-planes, the most significant bit is a sign bit. ’0’ represents positive and ‘1’ represents negative. The remaining bits are magnitude bits. The relationship between AC coefficients and quantized values are shown in Figure 23. A double dead zone quantization is used. That is, coefficients with small absolute values are quantized to positive zero whether they are positive or negative. From experimental result, we discover that double dead zone will make R-D performance better. This is because for coefficients with small absolute values, the error of sign does not reduce PSNR a lot but it cost many correction bits to recover.

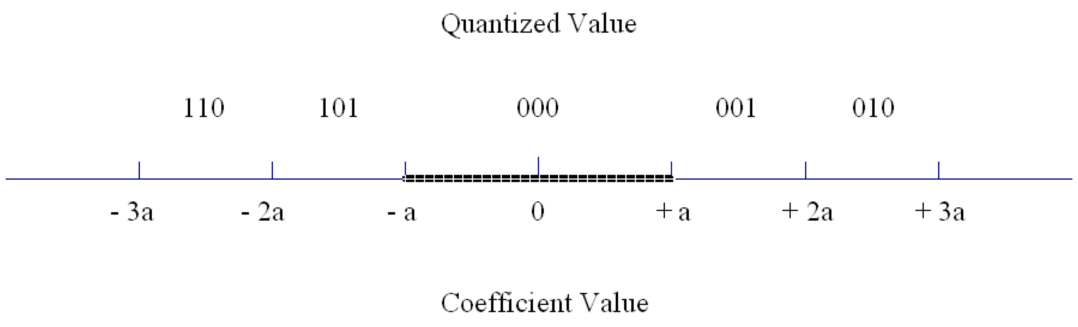


Figure 23. The relationship between AC coefficients and quantized values

4.5 LDPCA Encoding

Each coefficient band is represented in different number of bits as defined by the quantization matrices, so each band will be split into different number of bit-planes. Regular LDPCA codes with degree 3 will be used for encoding and decoding these bit-planes. For a QCIF resolution frame, each bit-plane contains 1584 bits, and this is the longest length of channel coding block we can choose. So we use length 1584 LDPCA code to encode and decode these bit-planes if all macroblocks are encoded and decoded. If only chosen macroblocks are encoded and decoded, then length 396 LDPCA code is used. The LDPCA encoding and decoding procedure of our DVC codec is rewritten from Stanford's MATLAB implementation. After encoding, the WZ bits are stored in a buffer.

4.6 LDPCA Decoding and Reconstruction

The model used to describe the error of side information is the Laplace distribution with scale parameter and zero mean. The scale parameter for each coefficient band is estimated offline for the entire sequence. So, there will be 16 scale parameters for 16 coefficient bands.

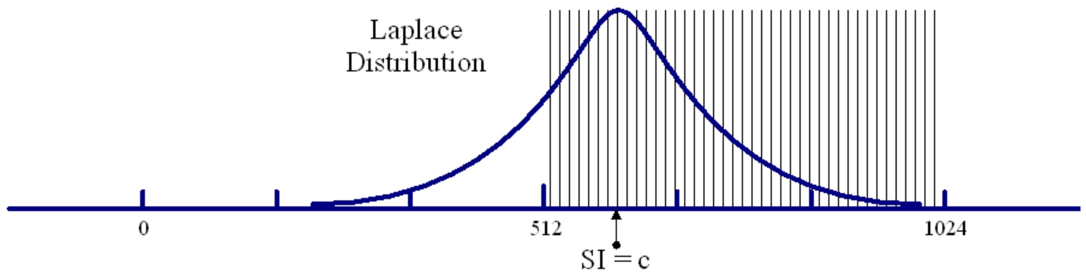


Figure 24. Probability calculation of second significant bit when most significant bit is decoded as 1

After side information is transformed and quantized. The quantized coefficients are split into bit-planes and then LDPCA decoded. For each bit-plane, LLR values of each bit are calculated according to error model and bit-planes decoded. To calculate LLR value of each bit b_i of one coefficient, probability P (b_i = 1) and P (b_i = 0) must be calculated first. Taking DC band for example, in Figure 24, the coefficient value of side information is c. When calculating LLR of most significant bit of c, the probabilities are calculated within the range 0 to 1024. But when calculating LLR of second significant bit of c, the probabilities are calculated within the range with stripe pattern if the most significant bit is decoded as 1.

When decoder performs LDPCA decoding, the rate will increase as more W-Z bits are requested by the decoder if the corrected side information still has errors. The rate can be from 2/66 to 66/66, and when rate equal to 66/66, all bits will be decoded without error. Each time W-Z bits are requested, iterative belief propagation with at most 1000 iterations is performed in order to decode bits. We choose 1000 iterations as upper bound because for most test sequences, 1000 iterations are close enough to convergence. For each iteration, the decoded bits are syndrome-checked. If syndrome-check is not satisfied, next iteration continues. When the belief propagation iterations converge, the values decoded must be checked. The decoder checks correctness of decoded values by peeking at the original values of WZ frame. If any of decoded values is wrong, then higher rate will be tried by requesting more W-Z bits, otherwise the LDPCA decoding procedure ends.

For coefficient c of side information, after one bit is decoded, if the bit is correct, then the value of c does not needs to be changed. But if the bit is wrong, then value of c will be reconstructed. As Figure 25 shows, when the bit is wrong, the value of c will not be in the correct range. So, the value of c will be reconstructed to the nearest value which is in the correct range. For example, when decoding most significant bit of coefficients in DC band, if the decoded value is 0, then the correct range will be from 0 to 511. If coefficient of the side information is outside the range, then it will be reconstructed to 511.

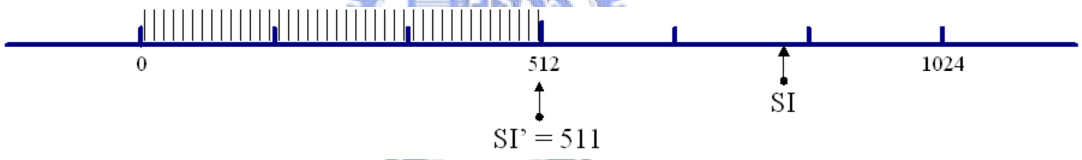


Figure 25. Side information is reconstructed to the stripe pattern area when the decoded value is 0

Because the macroblocks of group S_A are all in one LDPCA block, so we can only request WZ bits for these macroblocks and only decode them. For macroblocks of group S_B, no WZ bits are requested. So, prioritized channel decoding is achieved.

After LDPCA decoding, the macroblocks will be inverse transformed and rearranged to original order. Then WZ frame decoding is finished.

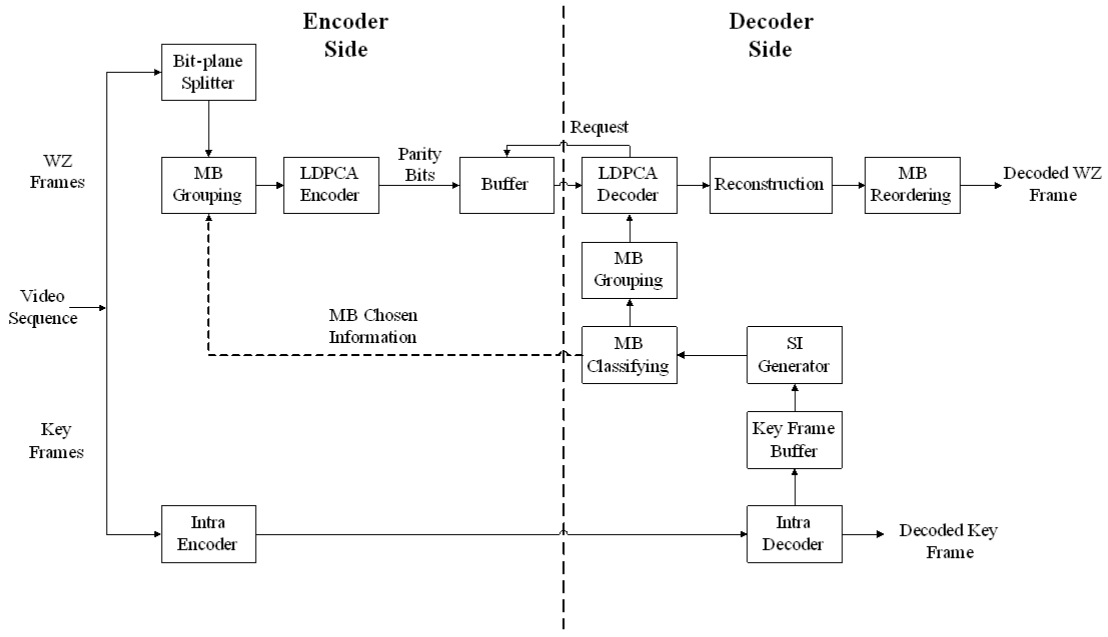


Figure 26. System flow of our pixel domain DVC codec

4.7 Pixel Domain DVC Codec

The block diagram of pixel domain DVC codec is shown in Figure 26. The differences between pixel domain DVC codec and frequency domain DVC codec are described as follows.

The pixel domain DVC codec works at pixel domain, so there is no transformation. The quantization is achieved while splitting pixels into bit-planes. At the encoder side, each WZ frame is split into 8 bit-planes. Discarding less significant bit-planes is equal to quantization. We discard 4 less significant bit-planes. At the decoder side, side information is generated using same method used in frequency domain DVC codec. The side information is also split into 8 bit-planes and 4 bit-planes are discarded.

For the frequency domain DVC codec, there are Laplace parameters for each coefficient band. For pixel domain DVC codec, one Laplace parameter for all pixels is estimated offline for entire sequence.

Each bit-plane is LDPCA encoded and decoded. For a QCIF sequence, each bit-plane contains 25344 bits. We use LDPCA codes with block length 6336, so each bit-plane will be packed into 4 LDPCA blocks. When only chosen macroblocks are encoded and decoded, each bit-plane will be packed into 1 LDPCA block. When the decoder performing LDPCA decoding, LLR values are calculated in the same way we have described in frequency domain DVC codec. Reconstruction is the same, too.

Chapter 5: Experimental Results

In this chapter, experimental results and performance of our frequency domain DVC with macroblock rearrangement are presented. The performance of DISCOVER frequency domain DVC codec is used for comparison because the DISCOVER codec has the best performance among published DVC codecs today. The testing conditions used in these experiments is the same as those used in DISCOVER, as shown in Table 4. The QP values for encoding key frames are listed in Table 3.

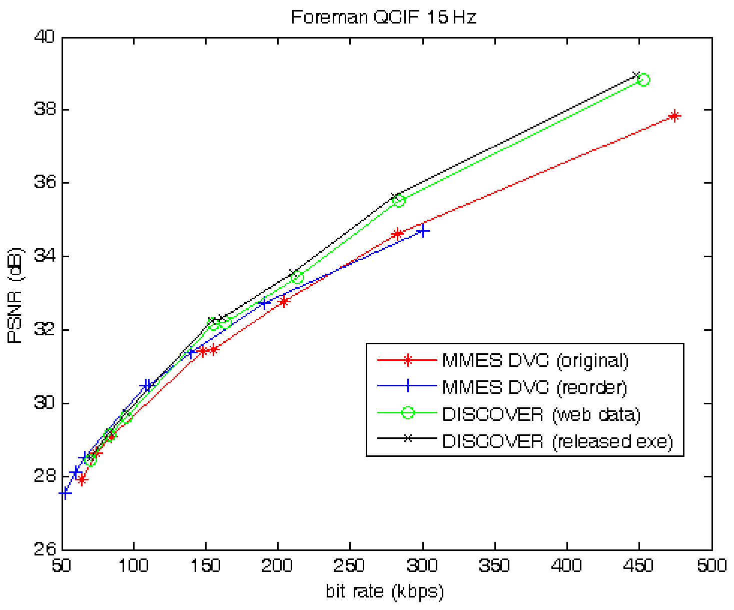
In Figure 27 there are R-D curves of our frequency domain DVC codec and DISCOVER DVC codec. For our DVC codec, we decode all macroblocks within a frame and plot one R-D curve. This curve represents original performance of our DVC codec. And then we apply our prioritized channel decoding method and plot another R-D curve. The two R-D curves of DISCOVER codec are plotted according to the performance data they present in their website and the data obtained by executing their released program.

Without prioritized channel decoding, performance of the proposed DVC codec is worse than DISCOVER. And it is mainly because the DISCOVER codec generates much better side information. In particular, when less significant bits of coefficient bands are LDPCA decoded, the simple frequency-domain DVC codec without prioritized coding requests more bits than the DISCOVER codec and the R-D performance is worse. The side information quality of our implementation when quantization matrices are Q_4 and Q_8 is listed in Table 5. The side information quality of DISCOVER's is not listed because they do not release the side information quality data.

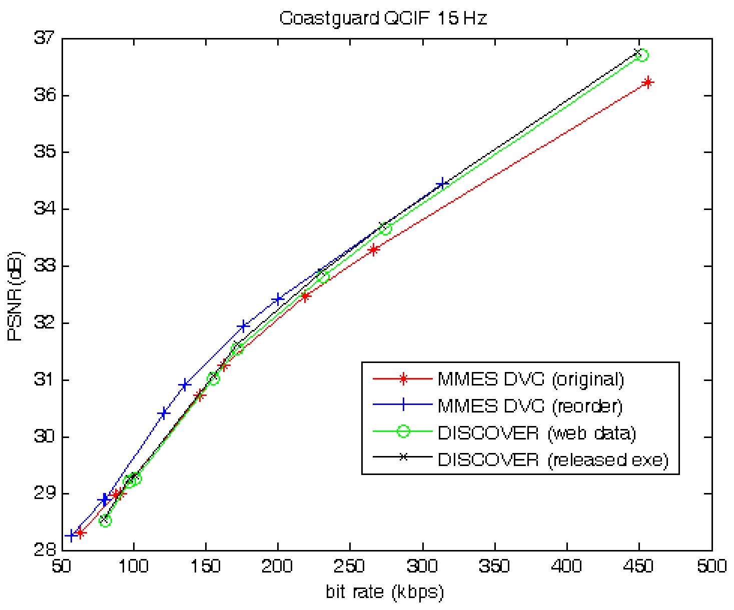
For Foreman, Hall Monitor, and Coast Guard sequences, when prioritized channel decoding is applied, the R-D performance will increase at most 0.5 dB when bit-rate is smaller than 200 kbps, which is a reasonable bit-rate range for QCIF sequences. However, for the Soccer sequence, the performance is worse when macroblock rearrangement and partial decoding is applied. This is because we choose SAD of macroblocks in neighboring key frames as a measure for deciding which macroblocks are worst and need more parity bits. The motion field of Soccer sequence is not regular and motion field prediction error affects side information quality a lot. The really worst macroblocks are not detected and parity bits are wasted on macroblocks not really bad and PSNR of side information increase a little. Although the proposed prioritized channel decoding DVC codec is not obviously better than the DISCOVER DVC codec, it is most likely due to worse side information quality. And side information quality improvement is not the key points in this thesis. If the side information quality of the proposed scheme is improved, the R-D performance of the proposed scheme should be better than that of the DISCOVER codec.

Table 4. DISCOVER DVC codec testing setting

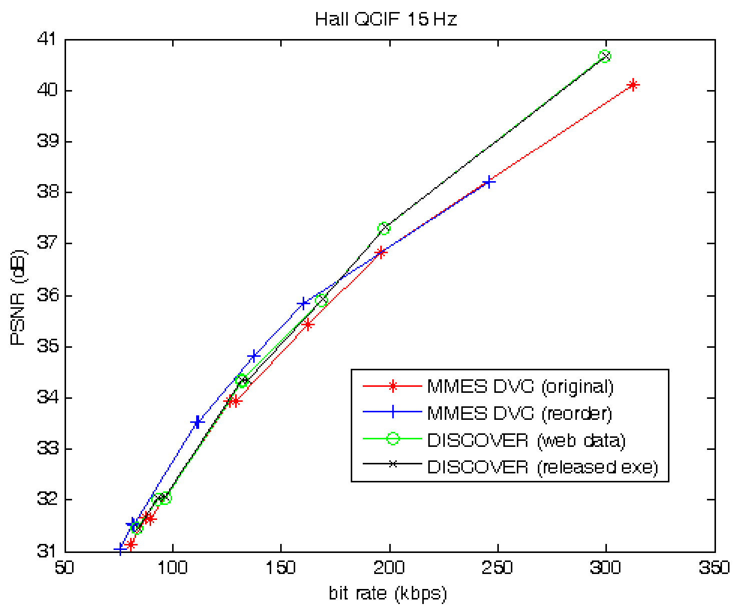
DISCOVER DVC Codec Testing Setting	
Test Sequence	Foreman, Hall Monitor, Coast Guard, and Soccer
Test Frames	All frames
Spatial Resolution	QCIF
Temporal Resolution	15 Hz, so when GOP length is 2, WZ frames are 7.5 Hz
Key Frames	H.264 intra (main profile)
GOP Length	2
Channel Code	LDPCA



Foreman QCIF 15 Hz
PSNR (dB)
bit rate (kbps)
MMES DVC (original)
MMES DVC (reorder)
DISCOVER (web data)
DISCOVER (released exe)



Coastguard QCIF 15 Hz
PSNR (dB)
bit rate (kbps)
MMES DVC (original)
MMES DVC (reorder)
DISCOVER (web data)
DISCOVER (released exe)



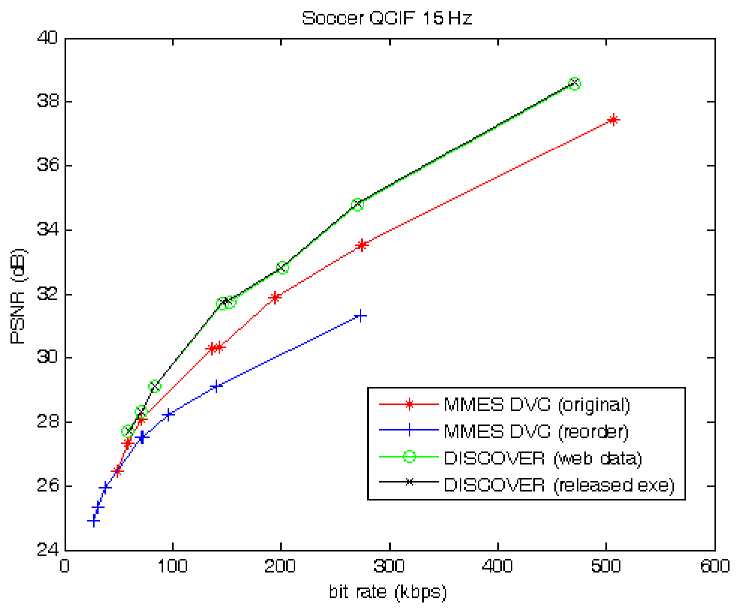


Figure 27. R-D performance of our frequency domain DVC codec

Table 5. Side information quality

	Key Frame QP = Q4	Key Frame QP = Q8
Foreman	27.9104	29.3168
Hall Monitor	32.7581	35.5093
Coast Guard	29.3375	31.0476
Soccer	22.1290	22.4782

Chapter 6: Conclusion and Future Work

In this thesis we present a prioritized channel decoding DVC scheme to improve the coding efficiency. After side information is generated, the decoder classifies macroblocks according to side information quality. And more WZ bits can be used to correct macroblocks whose side information quality is worse while less WZ bits are requested for macroblocks with less side information error. The WZ bits can be requested more efficiently.

From experimental results, we tried several cues and discover that SAD value of motion-matching macroblocks in neighboring key frames is a good cue to estimate the errors in side infromation. When the SAD value is larger, the macroblock is considered worse. Although the classification is not very accurate, for some test sequences, the R-D performance will increase especially for lower bit-rate. Compare to DISCOVER's DVC codec, our improvement is little. But it is because our side information quality is not as good as that of the DISCOVER codec. Our prioritized channel decoding DVC codec should be better than DISCOVER's if our side information quality is improved. Although side information quality improvement is not a key point in this thesis, it is one of the future work items.

In this thesis we try to use motion field variance, number of edges and corner points, and SAD of motion-matching macroblocks in neighboring key frames as cues for classifying macroblocks. In the end, only SAD of motion-matching macroblocks in neighboring key frames is used to classify macroblocks. But this cue is bad when motion is irregular and the performance will become very poor because macroblocks with really worse side information are not recognized. So, in the future we can combine several different cues for the decoder to classify macroblocks more correctly. For example, we can use motion field variance to decide whether we should use SAD or other cues to classify macroblocks. Only when motion field variance is not small, SAD can be used to classify macroblocks. If motion field variance is large, we will try to use other cues to classify macroblocks.

The decoder classifies macroblocks and sends the classifying result to the encoder. The encoder waits for the decoder's instruction before coding of a W-Z frame and delay occurs. If the macroblock classification is done by the encoder or the classification result can be guessed by the encoder, it will not be delayed. However, after observing classification result, we discover the classification is not regular enough for encoder to guess. And the classification at decoder side is not very good even the decoder has the side information. The encoder can not do better than decoder so we can not let encoder do this job.

In this experiment, we classify macroblocks to two groups $_A$ and S_B and prioritized channel decode them. In the future we can increase the number of groups and rate distribution can be more flexible. For example, we can classify macroblocks to three groups S_A, S_B and S_C. S_A is group with worse side information quality and S_C is group with best side information quality. The quantization matrix for group S_A, S_B and S_C will be Q_8, Q_4 and Q_1. So more bits are requested for this S_A and fewer bits are requested for group S_C. The percentage of S_A, S_B and S_C can also adaptive for different test sequences.

Reference:

[1] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, “Distributed video coding,” Proceedings of the IEEE, pp. 71–83, 2005.

[2] D. Slepian and J. K. Wolf, “Noiseless coding of correlated information sources,” IEEE Trans. Inform. Theory, 1973.

[3] A. D. Wyner and J. Ziv., “The rate distortion function for source coding with side information at the decoder,” IEEE Trans. Inform. Theory, 1976.

[4] L. Lu, D. He, and A. Jagmohan, “Side information generation for distributed video coding,” Proc.of IEEE ICIP, 2007.

[5] S. Klomp, Y. Vatis, and J. Ostermann, “Side information interpolation with sub-pel motion compensation for Wyner-Ziv decoder,” In Nordic Signal Processing Symposium, Reykjavik, Iceland, June 2006.

[6] R. Puri and K. Ramchandran, “PRISM: A “reversed” multimedia coding paradigm,” Proc. of IEEE ICIP, 2003.

[7] A. Aaron, R. Zhang, and B. Girod, “Wyner-Ziv coding of motion video,” Proc. Asilomar Conference on Signals and Systems, pp. 240–244, vol.1, 2002.

[8] A. Aaron and B. Girod, “Wyner-Ziv video coding with low-encoder complexity,” Proc. Of 2004 Picture Coding Symposium (PCS-2004), San Francisco, CA, Dec. 2004.

[9] J. Ascenso and F. Pereira, “Adaptive hash-based side information exploitation for efficient Wyner-Ziv video coding,” Proc. of IEEE. ICIP, 2007.

[10] M. Tagliasacchi, L. Frigerio and S. Tubaro, “Analysis of coding efficiency of motion-compensated interpolation at the decoder in distributed video coding,” IEEE Proc. of ICIP, 2007.

[11] 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, Sept. 2003.

[12] A. Aaron and B. Girod, “Compression with side information using turbo codes,” Proc. Of IEEE Data Compression Conf., pp. 252–261, Snowbird, Utah, USA, 2002.

[13] A. Aaron, S. Rane, and B. Girod, “Wyner-Ziv video coding with hash-based motion compensation at the receiver,” Proc. Of 2004 International Conference on Image Processing, vol. 5, pp. 3097–3100, 2004.

[14] A. Aaron, S. Rane, E. Setton, and B. Girod, “Transform-domain Wyner-Ziv codec for video,” Proc. Of Visual Communications and Image Processing, Jan. 2004.

[15] D. Varodayan, A. Aaron, and B. Girod, “Rate-adaptive distributed source coding using low-density parity-check codes,” Conference Record of the Thirty-Ninth Asilomar Conference, pp. 1203–1207, 2005.

[16] C. Brites, J. Ascenso, and F. Pereira, “Improving transform domain Wyner-Ziv video coding performance,” Proc. Of 2006 IEEE International Conference In Acoustics, Speech and Signal Processing, vol. 2, 2006.

[17] C. Brites, J. Ascenso, and F. Pereira, “Feedback channel in pixel domain Wyner-Ziv video coding: Myths and realities,” 14th European Signal Processing Conference (EUSIPCO2006), Sep. 2006.

[18] J. Ascenso, C. Brites, and F. Pereira, “Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding,” In 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services, July 2005.

[19] R. P. Westerlaken, S. Borchert, R. K. Gunnewiek, and R. I. L. Lagendijk, "On the comparison of distributed video coding using ldpc codes on bit-plane and symbol level," Proc. Of the Thirteenth Annual Conference Of The Advanced School For Computing And Imaging, 2007.

[20] Zhen Li and E. J. Delp, "Wyner-Ziv video side estimator: conventional motion search methods revisited," Proc. Of IEEE International Conference on Image Processing, vol. 1, 2005.

[21] S. Borchert, R. P. Westerlaken, R. Klein Gunnewiek, and R. L. Lagendijk, "On the generation of side information for DVC," In Twenty-eigth Sympoisum on Information Theory in the Benelux, pp. 1141–148, May 2007.

[22] R. P. Westerlaken, S. Borchert, R. K. Gunnewiek, and R. L. Lagendijk, "Dependency channel modeling for a ldpc-based Wyner-Ziv video compression scheme," In IEEE International Conference on Image Processing, pp. 277–280, Oct 2006.

[23] J. Garcia-Frias and Y. Zhao, "Compression of correlated binary sources using turbo codes," In IEEE Communication Letters, Oct 2001.

[24] M. Tagliasacchi, J. Pedro, F. Pereira, and S. Tubaro, "An efficient request stopping method at the turbo decoder in distributed video coding," In 15th EURASIP European Signal Processing Conference, 2007.

[25] A. Liveris, Z. Xiong, and C. Georghiades, "Compression of correlated binary sources using turbo codes," In IEEE Communication Letters, 2002.

[26] Li-Wei Kang and Chun-Shien Lu, "Low-complexity Wyner-Ziv video coding based on robust media hashing," In IEEE 8th Workshop on Multimedia Signal Processing, pp. 267–272, 2006.

[27] H. Kondo, K. Yamahara, and Jun Liao, “Identification of DCT signs for sub-block coding,” In Sixth International Symposium on Signal Processing and its Applications, vol. 2, pp. 569–572, 2001.

[28] Yixuan Zhang, and Ce Zhu, “Full Search of side-Information in distributed video coding,” Fourth International Conference on Image and Graphics, pp. 246-249, 2007

[29] L. Liu, Z. Li, and E. J. Delp, “Backward channel aware Wyner-Ziv video coding,” Proc. On IEEE International Conference on Image Processing, pp. 1677-1680, 2006

[30] C. S. Liu and H. Y. M. Liao, “Structural digital signature for image authentication: an incidental distortion resistant scheme,” IEEE Trans. On Multimedia, vol.5, no. 2, pp. 161-173, June 2003.

[31] J. Ascenso, C. Brites, and F. Pereira, “Motion compensated refinement for low complexity pixel based distributed video coding,” IEEE Conference on Advanced Video and Signal Based Surveillance, pp. 593-598, 2005

[32] M. Dalai, R. Leonardi, and F. Pereira, “Improving turbo codec integration in pixel-domain distributed video coding,” IEEE International Conference on Acoustics, Speech and Signal Processing, 2006.

[33] M. Tagliasacchi, A. Trapanese, S.Tubaro, J. Ascenso, C. Brites, and F. Pereira, “Exploiting spatial redundancy in pixel domain Wyner-Ziv video coding,” IEEE International Conference on Image Processing, pp. 53-256, 2006

[34] M. Schneider and S. F. Chang, “A robust content based digital signature for image authentication,” IEEE International Conference on Image Processing, Vol. 3, pp. 227-230, 1996

[35] F. Lefebvre, J. Czyz, and B. Macq, “A robust soft hash algorithm for digital image signature,” IEEE International Conference on Image Processing, 2003

[36] V. Monga, D. Vats and, B. L. Evans, “Image authentication under geometric attacks via structure matching,” IEEE International Conference on Multimedia and Expo, pp. 229-232, July 2005

[37] L. Yu and S. Sun, “Image robust hashing based on DCT sign,” IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 131-134, 2006

[38] C. Brites, “Advances on distributed video coding,” Instituto Superior Tecnico, Technical University of Lisbon, Portugal, Dec 2005.

[39] C. Harris and M. Stephens, “A combined corner and edge detector,” Proceedings of The Fourth Alvey Vision Conference, Manchester, pp 147-151. 1988

[40] M. Tagliasacchi and S. Tubaro, “Hash-based motion modeling in Wyner-Ziv video coding,” IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. I-5-9-I-512, 2007