Wyner-Ziv Based Bidirectionally Decodable Video Coding

Xiaopeng Fan, Oscar C. Au, Yan Chen, Jiantao Zhou, Mengyao Ma, Peter H.W. Wong

Abstract

In this paper, we propose a novel Wyner-Ziv based video compression scheme which supports encoding a new type of inter frame called ‘M-frame’. Different from traditional multi-hypothesis inter frames, the M-frame is specially compressed with its two neighbor frames as reference at the encoder, but can be identically reconstructed by using any one of them as prediction at the decoder. Based on this, the proposed Wyner-Ziv based bidirectionally decodable video compression scheme supports decoding the frames in a video stream in both temporal order and reverse order. Unlike the other schemes which support reverse playback, our scheme achieves the reversibility with low extra cost of storage and bandwidth. In error resilient test, our scheme outperforms H.264 based schemes up to 3.5dB at same bit rate. The proposed scheme also provides more flexibility for stream switching.

Key words: Video streaming, reverse playback, error resilience, Wyner-Ziv, M-frame
1. Introduction

Due to the rapid growth of the Internet and the increasing demand for video contents, video streaming services over communication network has received tremendous attention from academia and industry. For fast and user friendly browsing, it is desirable that video streaming systems should have the capability of providing effective video cassette recorder (VCR) functionality such as forward, backward (reverse playback), stop, pause, fast forward, fast backward, and random access. However, almost all the ISO and ITU video coding standards were mainly designed for forward play operations, and the hybrid video coding scheme employed in those standards severely complicates the backward play operations. With simple ‘IPPP...’ GOP (group of picture) structure of video sequence, although the last P-frame in a GOP is the first frame to be displayed in reverse playback, it can only be decoded and displayed after all the other frames in the GOP are transmitted and decoded. This requires a significant storage buffer in the decoder to store all the decoded frames in the GOP. Otherwise those frames need to be transmitted and decoded again when decoding the second last P-frame (and also the third last and so on) in the GOP.

To reduce the buffer size and the bandwidth requirement of reverse playback, it was proposed in [1] and [2] to transcode those original P frames into I frames or new P frames using future frames as reference. This transcoding approach breaks the interframe dependency from the requested frame to its previous frame (in temporal order). However it still requires large buffer size and requires extra complexity of the decoder, and is not applicable if the requested frame has not been decoded before. The dual-bitstream approach in [3] [4] [5] can be considered as performing a prior transcoding at the encoder, in which the server stores both the forward bitstream and the backward bitstream. It enables reverse playback with no extra bandwidth requirement, but approximately doubles the storage requirement of the server. In [6], a low-complexity online transcoding was performed at the encoder for stationary macroblocks (with zero motion) based on a macroblock classification rule so as to reduce the number of MBs to be decoded in the reverse playback operation. The idea is further extended in [7] where a newly mixed VLC/DCT-domain technique was proposed to handle the remaining macroblocks in order to further reduce the required bandwidth. In [8], this scheme was combined with the dual bitstream scheme. This combination requires a bandwidth as low as the dual bitstream scheme while reducing the storage requirement of the server. To deal with the drift problem in those transcoding schemes, the SP frames in H.264 [9] were adopted to replace some of the P frames in [10]. Another novel scheme [11] encoded the second half of frames in each GOP in reverse order. It reduces the required bandwidth of reverse playback, but also increases the bandwidth of forward play.

Although there are already so many schemes for reverse playback, none of them achieve reversibility with low extra storage and low extra bandwidth simultaneously.

In addition to the VCR functionalities, another interesting research topic is to improve the error resilience performance of the video streaming system when the video is transmitted over error-prone channels [12] [13]. At the network level, traditional error control and recovery schemes such as forward error correction (FEC) and automatic retransmission request (ARQ) have been extended for video transmission [14] [15]. However, for real-time video streaming, the transmission errors cannot be avoided even when FEC
and ARQ are combined [16]. Thus the video streaming systems are expected to provide sufficient build-in robustness (against transmission loss) at the source coding level. The existing source-coding-level approaches include intra block refreshment [17], long-term prediction [18], multiple description coding (MDC) [19], redundant picture [20], and SP/SI frames [21], etc. They mitigate or stop error propagation by reducing or modifying the temporal prediction dependency, and therefore achieve error resilience.

In this paper, we propose a novel Wyner-Ziv [22] [23] based bidirectionally decodable video coding (WZ-BID) scheme which supports both forward decoding and backward decoding, based on the flexibility of Wyner-Ziv coding [24] [25] [26]. Different from traditional B-frame and multi-hypothesis P-frame, the inter frame (called M-frame) in the proposed scheme can reference its two neighbor frames at the encoder but can be identically reconstructed with any one of the two neighbor frames available at the decoder. Fig.1 gives a comparison of the reference relationship among B-frames, P-frames and M-frames.

![Diagram of reference relationship among B-frames, P-frames, and M-frames](image_url)

The proposed WZ-BID scheme has following functionalities and advantages: Firstly, unlike the other schemes for reverse playback, WZ-BID can achieve reversibility at low extra costs of storage and bandwidth. Secondly, when a frame is lost during transmission, the decoder can decode the subsequent M-frames in the same GOP in the reverse order, thus avoiding error propagation efficiently. Thirdly, WZ-BID provides more flexibility for stream switching. It allows the users to switch at not only the predetermined positions (I-frames, SP-frames or SI-frames) but also M-frames. Note that the decoder needs enough buffer and delay to fully enable the functionalities of error resilience and stream switching. Otherwise, it will take only partial or even no advantage from M-frames for these two functionalities.

The rest of this paper is organized as follows: In section 2, we present the analysis and the implementation of the Wyner-Ziv based M-frame. In section 3, we propose our Wyner-Ziv based bidirectionally decodable video compression scheme. In section 4, we explain the three major functionalities of the proposed scheme. We show the experimental results in section 5 and conclude the paper in section 6.

2. Analysis and implementation of the Wyner-Ziv based M-frame

In this section, we present the analysis and the implementation of the M-frame which can be identically reconstructed as long as any one of its reference frames is available. The M-frame is encoded based on Wyner-Ziv code, i.e. error correction code such as LDPC code. The decoder has two possible prediction frames $Y_1$ and $Y_2$. The decoder
regards each prediction frame as a noisy version of the quantized current frame $\hat{X}$ and use the error correction code to recover $\hat{X}$. The error correction code $w$ is chosen such that it has enough error correction capability to fully recover $\hat{X}$ with any one of the two prediction frames. This guarantees an identical reconstruction $\hat{X}$ regardless of whether $Y_1$ or $Y_2$ is available at the decoder, i.e.:

$$\hat{X} = WZ^{-1}(w, Y_1) = WZ^{-1}(w, Y_2)$$  \hspace{1cm} (1)

where $WZ^{-1}(.)$ means Wyner-Ziv decoding.

Although in this paper we present our scheme by assuming two predictions $Y_1$ and $Y_2$, the results can be generalized easily to more than two predictions. Since our M-frame is implemented based on Wyner-Ziv code, we will first give a brief introduction of Wyner-Ziv coding and then present our theoretical analysis and practical implementation of the M-frame.

2.1. Wyner-Ziv coding and its application in video coding

![Wyner-Ziv coding](image)

Wyner-Ziv coding (WZC) is to compress a source $X$ when its side information $Y$ ($Y$ gives us some information of $X$, e.g. $Y$ is a prediction of $X$) is only available at the decoder. The basic principle behind WZC is that, efficient compression can be achieved when the source statistics are exploited—partially or wholly—at the decoder only [22] [23]. In a special lossless case of WZC called Slepian-Wolf coding (SWC) (i.e. $\hat{X} = X$ in Fig. 2), the minimum achievable rate has been proved to be $H(X|Y)$ in [22], i.e. the absence of $Y$ at the encoder does not affect the compression efficiency. In a general lossy case, the absence of $Y$ at the encoder causes rate increase. Nevertheless, the rate increase caused by the absence of $Y$ has been proved to be zero in the quadratic Gaussian case and the relaxed quadratic Gaussian case [27].

Often, WZC is based on some channel-coding schemes [28] [29] [30] [31]. Particularly for video signals, the modelling of the correlation between the current frame $X$ and the prediction frame $Y$ is through a virtual noisy channel: $X$ is the channel input and $Y$ is the channel output. The encoder encodes $X$ by error correction code, and the decoder reconstructs it by using the code and the channel output $Y$. Since video sources approximately satisfy the conditions of the relaxed quadratic Gaussian case, efficient video compression can be achieved by WZC when the reference frames are only available at the decoder [32] [33]. Although Wyner-Ziv coding appears only very recently, it has found considerable applications in video coding, including low-complexity video coding [32], distributed multiview video coding [34], joint source-channel coding [35], and robust scalable video coding [36].
2.2. Source coding with two alternative predictions

![Diagram of source coding with two alternative predictions]

Fig. 3. Source coding with two alternative prediction

The problem addressed in this paper is the compression of a source with two alternative predictions at the decoder as shown in Fig. 3. Suppose a source $X$ is to be compressed such that two possible predictions $Y_1$ and $Y_2$ are available at the encoder, but only one of them is available at the decoder. Exactly which one is available at the decoder is unknown to the encoder. In this new problem, as the decoder has the flexibility to decode with any one of the predictions, it should have considerable robustness when one of the predictions is lost as explained later. In [24], this problem is firstly addressed and the minimum rate has been presented. In this section, we will first prove this minimum rate and then presents the practical implementation on approaching this minimum rate.

2.2.1. Theoretical analysis

Let $r^*$ be the optimal rate of this system. Firstly it must satisfy $r^* \geq H(X|Y_1)$, because otherwise the rate will be lower than the Slepian-Wolf bound when the switch connects $Y_1$. And for the same reason the optimal rate $r^*$ also must satisfy $r^* \geq H(X|Y_2)$.

Therefore, we have:

$$r^* \geq \max(r^*_1, r^*_2)$$  \hspace{1cm} (2)

where $r^*_1 = H(X|Y_1)$ and $r^*_2 = H(X|Y_2)$.

This bound is actually tight for the jointly binary sources modelled by binary symmetric channel (BSC): $(X, Y_1, Y_2)$ are jointly binary sources, with the relationship between $X$ and $Y_1$ is modelled by a BSC with crossover probability $p_1$, and the relationship between $X$ and $Y_2$ is modelled by a BSC with crossover probability $p_2$. To show that the bound in Eq.2 is achievable, we assume without loss of generality that $0 \leq p_1 \leq p_2 \leq 0.5$. This implies that $r^*_1 \leq r^*_2$. We encode $X$ by a SWC code at rate $r^*_2$ by assuming the crossover probability between the source and the side information is $p_2$. When only $Y_2$ is available at the decoder, the codeword is decodable because the crossover probability between $X$ and $Y_2$ is $p_2$. When only $Y_1$ is available at the decoder, we can put $Y_1$ into another BSC channel with crossover probability $\alpha$ and get output $Y'_1$. Then the crossover probability between $X$ and $Y'_1$ is $\alpha (1-p_1) + (1-\alpha)p_1$. We choose $\alpha$ appropriately such that $p_2 = \alpha (1-p_1) + (1-\alpha)p_1$ and the relationship between $X$ and $Y'_1$ can also be modelled by a BSC channel with crossover probability $p_2$. This means that we can still decode $X$ by using $Y'_1$ as side information. Therefore we can decode $X$ when only $Y_1$ is available, because $Y'_1$ is directly obtained from $Y_1$. It follows that we can encode $X$ at rate $\max(r^*_1, r^*_2)$ such that it can be decoded regardless of either $Y_1$ or $Y_2$ is available at the decoder. This, together with Eq.2, establishes the following minimum achievable rate:

5
2.2.2. Practical implementation

The rate in Eq. 3 can be approached by scalable SWC code, such as the punctured Turbo codes [29] [30] and the rate adaptive LDPC codes [37]. Suppose $X$ is the binary source we want to encode, and $Y_1$ and $Y_2$ are the two alternative side information. Firstly we estimate the minimum rate by Eq. 3. Then we encode $X$ by the scalable SWC code and increase the rate until the codeword $w$ is enough to reconstruct $X$ losslessly when either $Y_1$ or $Y_2$ is the decoder side information. This $w$ will be the codeword we send to the decoder.

To evaluate the compression efficiency of this scheme, we also find the shortest codeword $w_1$ which can only be decoded with side information $Y_1$, and the shortest codeword $w_2$ which can only be decoded with side information $Y_2$. Since the code we use is a scalable SWC code, both $w_1$ and $w_2$ are prefixes of $w$. Furthermore, in most cases (> 99.8% in our simulation), the $w$ is the longer one of $w_1$ and $w_2$. And in the remaining cases, the $w$ is only slightly longer than the longer one of $w_1$ and $w_2$. This is because that, a longer codeword contains all the check node of a shorter codeword such that it has stronger error correction capability and decoding capability. For example, if $w_1$ is longer than $w_2$, then $w_1$ can also decode $X$ with side information $Y_2$ with very high probability.

Suppose the length of $w_1$ is $r_1 = r_1^* + \varepsilon_1$, and the length of $w_2$ is $r_2 = r_2^* + \varepsilon_2$, where the $\varepsilon_1 \geq 0$ and $\varepsilon_2 \geq 0$ are the rate redundancies of the scalable SWC code. Then the rate of the proposed method is:

$$r = \max(r_1, r_2) \leq r^* + \max(\varepsilon_1, \varepsilon_2)$$

This means the usage of the SWC code is reasonable for the problem, because it does not introduce extra rate redundancy.

In this paper we use the rate adaptive LDPC code in [37] as the scalable SWC code, because its performance is better than Turbo code at most bit rate. It is efficient at high bit rate but not very efficient at low bit rate. The performance comparison and the rate redundancy at different bit rate were given in [37].

2.3. The implementation of Wyner-Ziv M-frame

Here we apply our method in section B to encode the M-frames, which can be decoded as long as any one of its reference frames is available at the decoder. The proposed
framework is given in Fig. 5. The compression process is as follows:

We do motion estimation in spatial domain in the two reference frames, and generate two prediction frames. Then both the two prediction frames and the input source frame are transformed into frequency domain. The DCT coefficients in the source frame are then quantized into bitplanes. The encoding of each bitplanes are as follows:

Denote $X(i)$ as the i-th frequency band of the (transformed) source frame. Similarly, $Y_1(i)$ and $Y_2(i)$ are the i-th frequency bands of the two prediction frames respectively. $B(i)$ is the quantized $X(i)$, and $B(i, k)$ is the k-th bitplane of $B(i)$. Here $i$ is the order of reverse zigzag scan, and $k$ is the bitplane index. Note that $i = 0$ corresponds to the highest frequency AC component and $k = 0$ corresponds to the most significant bitplane (MSB).

From MSB to LSB (least significant bitplane), we encode each bitplane $B(i, k)$ by a scalable SWC encoder using the method presented in last subsection, while $Y_1(i)$ and $Y_2(i)$ are the two encoder predictions.

At the decoder, by transforming the available prediction frame we first get $Y_1(i)$ or $Y_2(i)$ (this depends on which reference frame is available). Then from MSB to LSB, we decode each bitplane $B(i, k)$ by using $Y_1(i)$ or $Y_2(i)$ as the decoder side information. As explained in last subsection, we can get exactly $B(i, k)$ at the decoder regardless of either $Y_1(i)$ or $Y_2(i)$ is used as the decoder side information. Therefore we can reconstruct the coefficients and the frame even if one reference frame is unavailable at the decoder.
3. Proposed Wyner-Ziv based bidirectionally decodable video coding

In this section, we propose a Wyner-Ziv based bidirectionally decodable video coding scheme (WZ-BID) with the GOP structure in Fig. 1 using the M-frames in the last section. In WZ-BID, each M-frame is encoded by the scheme in Fig. 5 using the previous frame and the next frame as the two reference frames such that the M-frame can be correctly decoded as long as any one of the two reference frames is available at the decoder. Since the compression scheme of a general M-frame has been given in last section, we will focus on several special problems and compression details of the bidirectionally decodable M-frame in this section.

3.1. Precalculation of the reconstructed next frame

Since the current frame (frame n) requires the next frame (frame n+1) as a reference frame, we need to precalculate the reconstructed next frame in order to encode the current frame, and this introduces a 1-frame delay at the encoder.

In the case that the frame n+1 is an I-frame, we simply encode and reconstruct it as the backward reference of the current frame n.

In the case that frame n+1 is also an M-frame, it seems we have a problem because both frame n and frame n+1 require each other as reference. However, notice that in Fig. 5 we have $\hat{B} = B$ such that

$$\hat{X} = Q^{-1}(\hat{B}) = Q^{-1}(B) = Q^{-1}(Q(X))$$

Thus the reconstruction of frame n+1 can be exactly precalculated as $T^{-1}(Q^{-1}(Q(X^{(n+1)})))$ without the $SWC$ and $SWC^{-1}$ in Fig. 5. The Fig. 7 shows the precalculation process where T means transform and Q means quantization.

![Fig. 7. Precalculation of the reconstructed next M-frame](image)

This precalculation arrangement does not increase encoder complexity. When encoding frame n+1 as shown in Fig. 5, the encoder will not need to calculate the $B$ and the reconstructed frame, because they have been precalculated when encoding frame n. Therefore, the increase (in complexity) due to the process in Fig. 7 is offset by the corresponding decrease in complexity in Fig. 5.

3.2. Motion estimation and motion vector compression

For each M-frame, we need to estimate and encode both forward motion vectors (FMV) and backward motion vectors (BMV) to generate prediction frames. Each FMV is predicted by the FMVs of the neighboring blocks as in H.264. If the next frame is not an M-frame, each BMV is predicted by the neighboring BMVs similarly. Otherwise, the BMVs of the current frame are predicted by the FMVs of the next frame, because they are highly correlated.
At the encoder, to predict the BMVs of frame n, the FMVs of frame n+1 are precalculated. The FMVs of frame n+1 are estimated by using the input frame n+1 and the reconstructed frame n (which has been precalculated when encoding frame n-1). Then the BMVs of frame n are predicted by the inverse of the FMVs of the frame n+1. The inverse of the FMVs is based on the motion field reversing algorithm in [38].

At the decoder, we only need to decode FMVs in forward decoding, and BMVs in backward decoding. In forward decoding, we can decode the FMVs of each frame directly. In backward decoding, since the BMVs are predicted by the FMVs, for each frame we decode both the FMVs and the BMVs. We can predict the BMVs of the current frame by the FMVs of the next frame because the next frame is decoded before the current frame in backward decoding.

3.3. DCT coefficients compression

The DCT coefficients are compressed as proposed in section 2.3. We use $Y_1$ and $Y_2$ to represent the DCT coefficients of the forward prediction frame and the backward prediction frame, respectively. We encode each bitplane $B(i,k)$ of DCT coefficients into codeword $w(i,k)$ as proposed in section 2.3 (by considering $B(i,k)$ as the source $X$ in section 2.2).

To evaluate the compression efficiency, suppose we also find the shortest codeword $w_1(i,k)$ needed to decode $B(i,k)$ with side information $Y_1(i)$, and find the shortest codeword $w_2(i,k)$ needed to decode $B(i,k)$ with side information $Y_2(i)$. Denote $r_1(i,k)$ and $r_2(i,k)$ as the length of $w_1(i,k)$ and $w_2(i,k)$. Then the length of $w(i,k)$ is $\max(r_1(i,k), r_2(i,k))$. Denote $R_0$ as the total rate of all bitplanes, $R_1$ as the minimum total rate of all bitplanes for forward decoding, and $R_2$ as the minimum total rate of all bitplanes for backward decoding. We have:

$$R_1 = \sum_{i,k} r_1(i,k), \quad R_2 = \sum_{i,k} r_2(i,k), \quad R_0 = \sum_{i,k} \max(r_1(i,k), r_2(i,k))$$

(6)

3.4. Optimal Lagrangian multiplier for motion estimation

In motion estimation, we minimize the Lagrangian cost function:

$$J_{ME} = D_{DFD} + \lambda R_{MV}$$

(7)

Here the prediction distortion $D_{DFD}$ is weighted against the MV cost $R_{MV}$ using a Lagrangian multiplier $\lambda$ which will be derived below. We assume that sum of square difference (SSD) is the distortion measure used in motion estimation. If sum of absolute difference (SAD) is used instead, we need to use $\sqrt{\lambda}$ as the Lagrangian multiplier in Eq.7. However this will not affect the following derivation of optimal $\lambda$.

In forward motion estimation, the optimal $\lambda$ follows:

$$\frac{1}{\lambda} = -\frac{dR_0}{dD}$$

(8)

and it has been found in previous researches that $\lambda \approx 0.85Q^2$ where $Q$ is quantization step size.
In backward motion estimation, all \( r_1(i, k) \) are fixed and we have:
\[
\frac{1}{X} = -\frac{dR_0}{dD} |_{r_1} = -\alpha \frac{dR_2}{dD} \approx \frac{\alpha}{X} \quad \Rightarrow \quad \lambda' \approx \frac{\lambda}{\alpha} \quad \text{(9)}
\]
where
\[
\alpha = \frac{dR_0}{dD} |_{r_1} / \frac{dR_2}{dD} = \left( \sum_{r_2(i, k) > r_1(i, k)} \frac{dr_2(i, k)}{dD} \right) / \left( \sum_{i, k} \frac{dr_2(i, k)}{dD} \right) \quad \text{(10)}
\]
and \( \alpha \) depends on the percentage of bitplanes which satisfy \( r_2(i, k) > r_1(i, k) \), and obviously we have \( 0 < \alpha < 1 \).

Therefore, we have \( \lambda' \approx 0.85\alpha^{-1}Q^2 > \lambda \). This means we decrease the weight of the prediction error in Eq.7 in the backward motion estimation. An intuitive explanation is that in the backward motion estimation a larger prediction error may and may not increase the total rate \( R_0 \). For example if we have \( r_1 > r_2 \) in Eq.4, then a larger prediction error increase \( r_2 \) but may not affect the total rate \( R_0 = \max(r_1, r_2) \).

Notice that we need to calculate \( \lambda' \) by \( \alpha \) before motion estimation, and at that time it is unknown whether \( r_2(i, k) > r_1(i, k) \) or not. Therefore, we assume \( \alpha = 0.5 \) in our experiments, because statistically the forward bit rate \( r_1(i, k) \) and backward bit rate \( r_2(i, k) \) have equal chance to be larger than each other. This Lagrangian optimized motion estimation improves the performance of our WZ-BID scheme by about 0.2dB in average.

3.5. Slepian-Wolf coding

We use the rate adaptive LDPC code in [37] as the scalable SWC code. We encode the transform coefficients in reverse zigzag order, starting from the highest frequency AC coefficients and ending up with the DC coefficients. For each frequency coefficient, from MSB to LSB, each bitplane is encoded as presented in subsection 2.2.

At the decoder, the input of the LDPC decoder is the LDPC code and the intrinsic LLR (log likely-hood ratio) of each input bit. The intrinsic LLR is:
\[
\text{inLLR}(i, k) = \log(P(B(i, k) = 0|Y(i), B^{-}(i, k))) - \log(f_{X|Y}(-x')dx') = \log(f_{X|Y}(x')dx') - \log(f_{X|Y}(x)dx) \quad \text{(11)}
\]
where \( B^{-}(i, k) = \{B(i, k - 1), B(i, k - 2), ..., B(i, 0)\} \) are bitplanes reconstructed prior to \( B(i, k) \);
\( Y(i) = Y_1(i) \) in forward decoding and \( Y(i) = Y_2(i) \) in backward decoding;
\( f_{X|Y}(x) \) is the conditional probability density function (PDF);
\( [x^-, x^+] \) is the possible range of \( X(i) \) determined by \( B^{-}(i, k) \);
and \( t \) is the threshold of current bitplane (for scalar quantization, \( t \) is the middle point of \( x^- \) and \( x^+ \)). We have Eq. (11) because of the fact that \( B(i, k) = 0 \) means \( X(i) \in [x^-, t] \) and \( B(i, k) = 1 \) means \( X(i) \in [t, x^+] \).

We assume the PDF \( f_{X|Y}(x) \) is Laplacian and \( f_{X|Y}(x) = \frac{1}{\lambda} e^{-\lambda|x-y|} \), where \( \lambda \) is calculated by the residue variance \( \sigma^2 \) through \( \lambda = \frac{2}{\sigma^2} \). And \( \sigma^2 = \mathbb{E}(X-Y)^2 \) is estimated for each coefficient in each block. For the \( i \)-th frequency coefficient \( X(i) \), \( \sigma^2 \) is estimated by those reconstructed higher frequency coefficients \( \hat{X}(0), \hat{X}(1), ..., \hat{X}(i-1) \) in the same
block. We scan the transform coefficients in reverse zigzag. This benefits the variance estimation of the low frequency coefficients. In our scheme the low frequency coefficients usually cost many more bits (than the high frequency AC coefficients) because low frequency coefficients tend to have larger variance \[39\]. Therefore, with reverse zigzag scan, the proposed scheme achieves better compression for low frequency coefficients and hence higher compression efficiency for the whole video.

Denote $\Delta = t - x^- = x^+ - t$ as the quantization step of the $k$-th bitplane. The intrinsic LLR is calculated by:

$$
\text{inLLR} = \begin{cases} 
+\lambda \Delta, & Y \leq x^-; \\
-\lambda \Delta, & Y \geq x^+; \\
\log \left( \frac{1 - e^{+\lambda(x^- - Y)}}{1 - e^{-\lambda(x^+ - Y)}} + \text{sgn}(t - Y)(1 - e^{-\lambda|t-Y|}) \right), & Y \in (x^-, x^+).
\end{cases}
$$

where $\text{sgn}(t - Y) = 1$ when $t - Y \geq 0$ and $\text{sgn}(t - Y) = -1$ when $t - Y < 0$.

The LDPC decoder performs belief propagation over the codewords and the intrinsic LLR, and outputs the extrinsic LLR. Then the decoder makes a hard decision on the extrinsic LLR to get each $B(i, k)$.

3.6. MMSE reconstruction

At the encoder, the reconstruction frames are calculated as shown in Fig. 7. Based on the quantized transform coefficients $B$, the dequantization of the transform coefficients is calculated by minimum mean square error (MMSE) estimation:

$$
\hat{X} = Q^{-1}(B) = \mathbb{E}(X|B) = \frac{\int_{x^-}^{x^+} x f_X(x)dx}{\int_{x^-}^{x^+} f_X(x)dx}
$$

where $[x^-, x^+]$ is the possible range of $X$ determined by $B$.

At the decoder, the reconstruction is also calculated through Eq.13. However, this reconstruction is not used for display, but only used as the reference of the next frame (in decoding order).

![Fig. 8. Refinement of the reconstruction](image)

For display, we calculate a refined reconstruction $\hat{X}^*$ by another MMSE based on not only $B$ but also $Y^*$, the prediction from the reference frame. This refined reconstruction $\hat{X}^*$ is calculated by:
\[
\hat{X}^* = \mathbb{E}(X|Y, B) = \frac{\int_{x^-}^{x^+} x f_{X|Y}(x) dx}{\int_{x^-}^{x^+} f_{X|Y}(x) dx}
\]
\[
= Y + \frac{(1 + \lambda|z^-|)e^{-\lambda|z^-|} - (1 + \lambda|z^+|)e^{-\lambda|z^+|}}{\text{sgn}(z^+)\lambda(1 - e^{-\lambda|z^+|}) - \text{sgn}(z^-)\lambda(1 - e^{-\lambda|z^-|})}
\]

where

- \(Y\) comes from last frame in forward decoding, and from next frame in backward decoding;
- \([x^-, x^+]\) is the possible range of \(X\) determined by the bitplanes \(B\);
- \(z^- = x^- - Y^*\) and \(z^+ = x^+ - Y^*\).

This refinement is recursive, i.e. the refined reconstruction is used during the refinement of the next frame (in decoding order), as the reference.

4. Functionalities of the proposed WZ-BID scheme

In this section, we explain the three main functionalities of the proposed WZ-BID scheme as video streaming systems. For the VCR application and the stream switching application, we propose to extract the simplified M-frames from the original M-frames to save bit rate. For the error resilient application, we propose a hybrid scheme to improve the RD performance.

4.1. On supporting VCR functionality

By replacing the P-frames in a traditional streaming system by the proposed M-frames, our WZ-BID can readily support reverse play operation. The client can receive and decode the M-frames in normal order, and can also receive and decode the M-frames in reverse order.

However, to reduce network traffic, during forward play we propose not to send a whole M-frame to the decoder, but send a simplified M-frame which only supports forward decoding: This simplified M-frame is extracted from the original M-frame. It contains the FMVs but no BMVs. And its total bit rate for transform coefficients is \(R_1\) (rather than \(R_0\) as explained in section 3.3). For each frequency band \(i\) and each bitplane \(k\), the codeword is \(w_1(i, k)\) (rather than \(w(i, k)\) in section 3.3).

The codeword \(w_1(i, k)\) is extracted from \(w(i, k)\) as follows: As we explained in section 2.2, both \(w_1(i, k)\) and \(w_2(i, k)\) are prefix of the codeword \(w(i, k)\), and the longer one of them is equal to \(w(i, k)\) in 99.8% cases. In those 99.8% cases, we encode the rate difference \(d(i, k) = r_1(i, k) - r_2(i, k)\). Otherwise we encode \(d(i, k) = 0\). We store \(d\) in the server, and the cost is about 1 ~ 4kb/s depending on the frame rate. When we extract \(w_1(i, k)\), if \(d(i, k) \geq 0\) then we let \(w_1(i, k) = w(i, k)\). Otherwise we extract \(w_1(i, k)\) from \(w(i, k)\) by removing the last \(d(i, k)\) bits. It can be verified that, the extracted \(w_1(i, k)\) is always forward decodable, and the total rate of all \(w_1(i, k)\) is approximately \(R_1\).

We extract both FMVs and forward codewords \(w_1(i, k)\) from the original M-frame to form the simplified M-frame. Although this simplified M-frame has fewer bits and is not backward decodable, its reconstruction frame is exactly the same as the original M-frame in forward decoding.
Similarly, during backward play we only need to send BMVs and the Wyner-Ziv coded transform coefficients at rate $R_2$. In backward play, the extraction of the simplified M-frame is similar to the extraction in forward play. Since the BMVs are predicted by the FMVs in the original M-frames, the server needs to decode the BMVs, re-encode them and send them to the client.

4.2. Stream switching

Stream switching is difficult for the conventional P-frames. When switching from one stream to another stream at a P-frame, normally we need to transcode it into an I-frame to make it decodable. This causes error propagation starting from the switching time point. H.264 overcame this error propagation by encoding the SP-frames which exactly synchronizes two streams. However, the switching can only happen at specific frames (the switching points) and each switching point needs one extra SP or SI-frame which increases the bit rate considerably.

![Diagram of video stream switching with proposed M-frames](image)

WZ-BID provides more flexibility for stream switching. It allows the users to switch at not only the predetermined positions (I-frames, SP-frames or SI-frames) but also M-frames. The M-frames after the switching point (before the next I-frame) can be decoded in reverse order. This switching can also happen between two streams of different video content, without extra I-frame inserted.

Similar to the application for VCR functionality, we can transmit the simplified M-frames instead of the full M-frames to save bits. Fig. 9 gives an example: When the user requests to switch from Stream 2 to Stream 1 at the fourth frame, the server can send the simplified M-frames to the user. The first 3 simplified M-frames from Stream 2 only need to support forward decoding, and the following 5 simplified M-frames from Stream 1 only need to support backward decoding. The simplified M-frames are extracted from the original M-frame as described in the last subsection. Only the original M-frames are stored in the server.

A shortcoming of our switching scheme is the decoding delay due to backward decoding. After a switching, the decoder can start decoding only after it has received the next I-frame. The target application of this paper is the video streaming system with client-server network structure. Normally in these applications the clients need to buffer tens of seconds (or more) of video data before decoding and display for stable playback. To fully enable the flexible stream switching functionality, the decoder needs an extra delay of one GOP (several seconds). Otherwise, it will take only partial or even no advantage from M-frames in stream switching.
4.3. Error resilience

4.3.1. M-frame only

Our M-frame is error resilient when transmitted over error prone channel, because the M-frames after a lost frame can be decoded in reverse order starting from the next I-frame. Unlike the application of VCR functionality and the application of stream switching, to achieve this robustness we need to send the whole M-frames rather than the simplified M-frames. Let $n$ be the GOP length. Table 1 gives the analytical results of average error propagation length (number of frames with error) when one or two frames are lost. We can see that our scheme has shorter error propagation length than the conventional scheme.

Table 1

<table>
<thead>
<tr>
<th>Error propagation length</th>
<th>Conventional scheme with GOP ‘IPPP...’</th>
<th>Proposed scheme with GOP ‘IMMM...’</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 frame lost</td>
<td>$\frac{n+1}{2}$</td>
<td>1</td>
</tr>
<tr>
<td>2 frames lost</td>
<td>$\frac{3}{2}(n+1)$</td>
<td>$\frac{3}{4}(n+4)$</td>
</tr>
</tbody>
</table>

However, our M-frame has lower RD performance than the H.264 P-frame when transmitted over a lossless channel. This suggests that, in an erroneous case, P-frame may be more efficient than M-frame when the decoder can obtain the forward prediction frame with high probability.

4.3.2. Hybrid M-frame/P-frame scheme

To achieve better RD performance for both error prone and lossless channels, we propose a hybrid GOP structure which uses both P-frames and M-frames as shown in Fig.11. We encode the first $m$ frames after the I-frame as P-frames, and the remaining $n-m+1$ frames as M-frames. There are two major reasons for this GOP structure. Firstly, the probability that the $t$-th inter frame in a GOP can be forward decoded correctly is $(1-p)^{t+1}$, where $p$ is the frame loss probability. In particular, the frames at the beginning of a GOP can have very high probability to be forward decoded correctly. Therefore, the backward decodable capability of the M-frames contributes so little to those frames that the additional bits of M-frames over P-frames cannot be justified. Secondly, P-frames should not be used after the first M-frame because otherwise the first M-frame cannot be backward decoded.

The optimal $m$ is derived as follows: Let $g_1$ be the distortion reduction of using P-frame (compared with M-frame) for the current frame when the forward prediction and the
current frame are available. Let $g_2$ be the distortion increase of using P-frame (compared with M-frame) when the forward prediction is not available but the backward prediction and the current frame are. For frame $m$, the expected distortion reduction of encoding it as P-frame (compared with M-frame) is:

$$G(m) = g_1 \times (1 - p)^{m+1} - g_2 \times (1 - (1 - p)^m)(1 - p)^{n-m+1}$$

An optimal $m$ should follow $G(m) \geq 0$, for otherwise it will be better to encode frame $m$ as an M-frame, and $G(m+1) < 0$, for otherwise it will be better to encode frame $m + 1$ as a P-frame. Since $G(m)$ is a decreasing function of $m$, the optimal $m$ is unique and is the largest integer number which satisfies $G(m) \geq 0$. Notice that $m < 1$ means all frames are encoded as M-frame, and $m \geq n - 1$ means all frames are encoded as P-frame.

In our experiments $g_1$ is estimated by subtracting the mean square error (MSE) of the M-frames and the P-frames by assuming both are correctly forward decoded. Similarly, $g_2$ is estimated by subtracting the MSE of the M-frames and the P-frames when the frames are backward decodable but not forward decodable (by assuming the previous frame is lost).

5. Experimental results

In our experiments, I-frames and P-frames are encoded by H.264 encoder (JM13.2), and M-frames are encoded by our WZ-BID scheme. Both P-frames and our M-frames use variable block size (from $4 \times 4$ to $16 \times 16$) in motion estimation. In this section, we will first compare the proposed bidirectionally decodable M-frame with the conventional P-frame in terms of the compression efficiency. Then we will evaluate the three applications of M-frame in video streaming system, including the support of the VCR functionalities, the improvement of the error robustness, and the support of more flexible channel switching.

5.1. Compression efficiency

The comparison of the compression efficiency is mainly between the H.264 baseline profile (JM 13.2) and the proposed WZ-BID scheme. Besides, we also implement an H.264 based bidirectionally decodable scheme and the proposed simplified M-frame. The H.264 based bidirectionally decodable scheme encodes both forward prediction residue and backward prediction residue for each inter frame. It has the same functionalities as the proposed scheme and its bitrate is basically the storage cost of [3]. The simplified M-frame is extracted from the original M-frame as explained in section 4.1, and it only supports forward decoding like H.264 P frame does.

According to our experiment, in the proposed bidirectionally decodable M-frame, about 80% ~ 90% of the total bits are the Wyner-Ziv bits and the remain bits are for the MVs. The simplified M-frame saves about 6% ~ 11% Wyner-Ziv bits and about 30% motion bits. As a result the simplified M-frames are about 10% smaller than the bidirectionally decodable M-frames.

The R-D curves are given in Fig.12. The H.264 scheme has the highest compression efficiency. The simplified M-frame curve is lower than H.264. This gap mainly comes from the difference between the residue coding in H.264 and the Wyner-Ziv coding in our scheme. Comparing the proposed M-frame and simplified M-frame curves, we can see
that M-frame achieves reversibility with a cost of less than 1dB loss in PSNR. Compared with the H.264 based bidirectional scheme, our M-frame scheme gains up to more than 3dB in PSNR for the same bit rate.

Similar to P-frames, the proposed bidirectionally decodable M-frames can be combined with B-frames to form GOP structures like ‘IBBBMBBBM...’. Such GOP structures also support both forward decoding and backward decoding. In Fig.12 (d), we can see the compression efficiency of ‘IBBBMBBBM...’ is very close to the traditional ‘IBBBPBBBP...’ GOP structure.

The PSNR loss of the proposed WZ-BID compared with H.264 tends to be larger at lower bit rate than at higher bit rate. This is mainly because the rate adaptive LDPC code we used has larger rate redundancy at low bit rate [37]. For the same reason, WZ-BID tends to perform better for fast motion video such as ‘football’ than for slow motion video such as ‘news’ (Fig.12 (e) and (f)) because slow motion video tend to require lower bit rate. We also observe that a large GOP length does not degrade the compression efficiency test

Fig. 12. Compression efficiency test
efficiency (Fig. 12(c)).

Although the coding efficiency is lower than H.264 P-frame, the proposed M-frame is bidirectionally decodable. In the following experiments we will show that we can achieve better performance in some applications by replacing the P-frames by the proposed M-frames.

5.2. The support of VCR functionality

Fig. 13 shows the frame-by-frame comparison of the required number of bits transmitted over the network during the reverse playback of the sequence “foreman” in four systems: the conventional H.264 system, the H.264 based dual bitstream scheme [3], the H.264 scheme with special GOP structure [11] and our system proposed in section 4.1.

From this figure, first we can see that the bandwidth requirement for H.264 can be very large especially for the last few P-frames in each GOP. The scheme in [11] avoids this by encoding the last several frames in every GOP in reverse order, but still needs large bandwidth for the first several frames in each GOP. Both the proposed system and [3] require very low bandwidth, significantly lower than H.264 and [11], with [3] being slightly lower.

An further comparison is given in Table 2. We try to control the quantization step size such that all the schemes have similar PSNR and can be compared fairly in terms of bit rate. We can see that the H.264 scheme has very large backward bit rate which means it requires very large bandwidth for backward play as expected. The backward bit rate is reduced to about 50% with the scheme in [7], but is still very large. The scheme in [3] has low forward rate and backward rate, but needs double the storage space. The scheme in [11] has lowest storage cost but large forward rate and backward rate. [7] and [11] are good when the bandwidth is not very critical, and [3] is good when the storage is not very critical. While each of H.264, [3], [7] and [11] is significantly worse than the others in at least one category of forward rate, backward rate and storage rate, the proposed scheme can achieve a balanced tradeoff, and is suitable when both bandwidth and storage are critical.

In Table 2, the ‘fast forward’ is the minimum jump in fast forward operation, and n is GOP length. [3] is better than the other schemes because it can jump not only a whole
Table 2
VCR tests (foreman qcif at 30Hz, GOP=30)

<table>
<thead>
<tr>
<th>test schemes</th>
<th>PSNR (dB)</th>
<th>bit rate (kb/s)</th>
<th>fast forward</th>
<th>random access</th>
<th>coding delay</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>forward</td>
<td>backward</td>
<td>storage</td>
<td></td>
</tr>
<tr>
<td>H.264</td>
<td>40.82</td>
<td>861.49</td>
<td>13519.43</td>
<td>861.49</td>
<td>×n</td>
</tr>
<tr>
<td>[3]</td>
<td>40.82</td>
<td>861.49</td>
<td>860.57</td>
<td>1722.06</td>
<td>×n/2</td>
</tr>
<tr>
<td>[7]</td>
<td>40.82</td>
<td>861.49</td>
<td>8068.40</td>
<td>861.49</td>
<td>×n</td>
</tr>
<tr>
<td>[11]</td>
<td>40.81</td>
<td>4184.22</td>
<td>4515.86</td>
<td>855.95</td>
<td>×n</td>
</tr>
<tr>
<td>proposed</td>
<td>40.81</td>
<td>915.66</td>
<td>939.16</td>
<td>998.53</td>
<td>×n</td>
</tr>
</tbody>
</table>

GOP but also half of a GOP each time. The ‘random access’ is the average number of frames needed to be decoded in order to access a randomly selected frame. Our scheme is one of the fastest requiring n/4 frames to be decoded. In our scheme, a random frame in the first half of GOP can be accessed by forward decoding requiring an average of n/4 frames to be decoded. Similarly, a random frame in the second half is backward decoded and requires n/4 frames to be decoded. The ‘coding delay’ is the delay caused by GOP structure in terms of number of frames. The proposed scheme introduces a 1-frame delay as explained in section 3, slightly worse than H.264 and [7] but significantly better than [3] and [11].

As to the complexity, Table 3 shows that decoding a Wyner-Ziv frame is 40 ~ 500 times more complex than decoding an H.264 P-frame. This is basically consistent with the previous result in the literature [40]. To reduce the complexity, we then applied in our scheme a near optimal fast LDPC decoding method [41]. This method approximates the log-tanh calculation in standard belief propagation algorithm by normalized min-sum operation. In our experiment the normalizer (α in [41]) is fixed to 1.25. As shown in Table 3, with [41], the proposed scheme achieves 5 ~ 6 times speedup with 0.2% ~ 3.5% penalty in bitrate.

Table 3
Complexity comparison for forward decoding

<table>
<thead>
<tr>
<th>GOP structure</th>
<th>IPPP 30Hz</th>
<th>IBBBP30Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>decoding time (in seconds)</td>
<td>WZ-BID</td>
<td>5.33</td>
</tr>
<tr>
<td></td>
<td>WZ-BID(normalized min-sum)</td>
<td>0.860</td>
</tr>
<tr>
<td></td>
<td>H.264</td>
<td>0.010</td>
</tr>
<tr>
<td>bitrate (kb/s)</td>
<td>WZ-BID</td>
<td>998.53</td>
</tr>
<tr>
<td></td>
<td>WZ-BID(normalized min-sum)</td>
<td>1033.78</td>
</tr>
<tr>
<td>ΔBr</td>
<td>3.53%</td>
<td>2.24%</td>
</tr>
</tbody>
</table>

In forward decoding, [3] and [7] have basically the same complexity as H.264, but [11] is more complex due to large forward bitrate as shown in Table 2. For the same reason, in backward decoding, [3] has similar low complexity, but H.264, [11] and [7] will be more complex than [3]. In both forward decoding and backward decoding, the proposed scheme is the most complex one in most cases. However, as a newly emerged video coding scheme,
Wyner-Ziv video coding may be further optimized in the future. Notice that in H.264 decoder there are very few floating point calculations. A possible further optimization of the proposed scheme is to replace the floating point calculations by fix point.

5.3. Application in error resilient video streaming

In Fig. 14 we illustrate a frame-by-frame comparison between 4 schemes: the proposed WZ-BID scheme with M-frame only, the proposed hybrid WZ-BID scheme, the normal H.264 (with no intra refresh), and the H.264 with 10% intra block refresh. The test sequence is foreman (qcif) at 7.5Hz and the bit rate is 138kb/s. GOP length is 11 and thus frame 1, 12, 23, 34, 45 are intra frames. The frames 6, 19, 21 and 35, are assumed to be lost in the channel. In the two proposed schemes, the inter frames before the first lost frame in each GOP are decoded in forward order, and the M-frames after the last lost frame are decoded in backward order. Other inter frames are concealed by the weighted average of the two nearest available frames. In the two H.264 schemes, each lost frame is concealed by copying its previous frame.

In the two proposed schemes, the M-frames after the last lost frame in each GOP are correctly decoded in reverse order, and therefore have significantly higher PSNR (see Fig.14) and better visual quality (Fig.15) than the P-frames in the two H.264 schemes. When there are more than 1 frame lost in a GOP, the frames between the first lost frame
and the last lost frame (e.g. frame 19∼21) also have better quality than H.264 P-frames because they are concealed by using both forward and backward nearest available frames. When no frame is lost (e.g. the 3rd GOP in Fig.14), H.264 schemes perform better than the two proposed schemes. The hybrid scheme performs better than the WZ-BID in all GOPs except the last GOP, in which a frame (frame 35) before the last P-frame (frame 37) is lost (m=3 in this test).

Fig. 16. Error resilience performance (foreman qcif at 7.5Hz): (a) at bitrate 50kb/s (b) at bitrate 317kb/s

Fig.16 is the comparison of the error resilience performance at different frame loss rate including 0%, 3%, 5%, 10%, 20%. Fig.17 is the comparison of the rate distortion performance at loss rate 3% and 10%. For each loss rate, we simulate 100 different loss patterns obtained from [42]. In the tests the proposed hybrid scheme performs the best. It gains 0.5dB over the proposed WZ-BID scheme at low loss rate and low bitrate, and 3 ∼ 3.5dB over the two H.264 schemes at high bitrate and high loss rate.

Table 4 gives the optimal $m$ ($m$ is the number of P-frames encoded in each GOP) calculated by Eq.15 for ‘foreman qcif’ at different bit rate and packet loss rate, and the
Table 4
The optimal $m$ and the PSNR gain of the hybrid scheme

<table>
<thead>
<tr>
<th>Bit rate</th>
<th>$p=0$</th>
<th>$p=0.03$</th>
<th>$p=0.05$</th>
<th>$p=0.1$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$m$ gain (dB)</td>
<td>$m$ gain (dB)</td>
<td>$m$ gain (dB)</td>
<td>$m$ gain (dB)</td>
</tr>
<tr>
<td>50kb/s</td>
<td>10</td>
<td>6</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>136kb/s</td>
<td>10</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>317kb/s</td>
<td>10</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

PSNR gain of the hybrid WZ-BID scheme over the non-hybrid WZ-BID scheme. Note that the optimal $m$ is different for different bit rate because the $g_1$ and $g_2$ in Eq.15 change with bit rate. The results suggest that the hybrid scheme can achieve considerable PSNR gain over the non-hybrid scheme. The gain tends to be larger at lower bit rate and/or lower lost rate. And the optimal $m$ also tends to be larger at lower bit rate and/or lower lost rate.

5.4. Application in stream switching

![Stream switching](image)

Fig. 18. Stream switching (foreman qcif): stream 1 is at 43.5kb/s; stream 2 is 284kb/s.

Here we illustrate the application of the proposed (non-hybrid) WZ-BID in stream switching with a small example in Fig.18. In this figure, ‘stream 1’ is ‘foreman’ coded at low bit rate, and ‘stream 2’ is ‘foreman’ at high bit rate. Both of the two curves are obtained through forward decoding. ‘Transmitted stream’ is the result of stream switching. The switching points are at frames 6, 21 and 36, while I-frames are located at frames 1, 12, 23, 34, 45, ... (GOP length=11). In the transmitted stream, the frames 7$\sim$11, 22 and 37$\sim$44 are backward decoded, while the remaining inter frames are forward decoded. In this figure we can see that the stream switching does not introduce error drifting and quality degradation.

For each frame, both the forward reconstruction and the backward reconstruction given by Eq.13 are exactly the same. However, the PSNR in Fig.18 are of the displayed reconstructions given by Eq.14. In Eq.14 the $Y^*$ in are different in forward decoding and backward decoding. Therefore, the backward decoded frames (7$\sim$11, 22 and 37$\sim$44) in the transmitted stream are slightly different from the original forward decoded frames in stream 1 and stream 2. However, this fluctuation is basically acceptable since it is much smaller than the difference between adjacent P and I frames (e.g. 22 and 23).
6. Conclusion

In this paper, we propose a novel Wyner-Ziv based bidirectionally decodable (WZ-BID) video compression scheme which supports both forward and backward decoding. Unlike existing schemes for backward playback, our scheme achieves reversibility with low extra storage and low extra bandwidth. In error resilient experiments, our scheme outperforms H.264 by up to 3.5dB in PSNR at the same bit rate and loss rate. We also proposed a hybrid WZ-BID scheme which further improves the end-to-end RD performance especially at low bit rate and low packet loss rate. The proposed (non-hybrid) WZ-BID scheme also provides more flexibility for stream switching.

In future works, we will seek to improve the compression efficiency of M-frames especially at low bit rate. Currently, it has lower RD performance than H.264. This is also a common problem of the other Wyner-Ziv based video coding schemes. Another work is to further reduce the complexity of the LDPC decoder.

References