Robust Predictive Coding and the Wyner-Ziv Problem

Anshul Sehgal and Narendra Ahuja
Beckman Institute, University of Illinois, Urbana, IL 61801
asehgal@uiuc.edu, ahuja@vision.ai.uiuc.edu

Abstract

This paper addresses the problem of robust communication of predictively encoded video in a joint source-channel setting. Specifically, the problem of predictive mismatch, where there is a drift between the state of the encoder and the decoder is addressed as a variant of the Wyner-Ziv problem. We propose a video encoding algorithm based on the H.26L video codec which prevents the propagation of error in predictively encoded video in the event of predictive mismatch (or drift) between the encoder and the decoder. One of the main advantages of the proposed approach is that there is minimal loss in performance over the standard H.26L encoder during error-free transmission, while simultaneously allowing error recovery in the event of errors. Using turbo codes as coset codes, we evaluate the performance of the proposed codec and demonstrate the efficacy of the proposed framework. The performance of the proposed approach can only improve with the use of superior coset codes.

1 Introduction

Predictive encoding is a well-known source coding technique for efficient, low-latency removal of temporal redundancy in audio and video compression systems. In the case of video, another alternative for removal of temporal redundancy is the use of 3D subband coding. However, 3D subband coding suffers from poor compression, introduces artifacts in the reconstructed video, requires high computation and has significant latency (due to the large number of frames that need to be accumulated before compression). Owing to these drawbacks, predictive coding remains the most viable means of removing temporal redundancy from multimedia streams. Unfortunately, in the context of transmission over lossy channels, it is also a fragile compression technique, as the successful decoding of any symbol is dependent on the successful decoding of all preceding symbols.

The key problem in the communication of predictively encoded media is that of predictive mismatch. Predictive mismatch refers to the scenario in which there is a mismatch between the predictor symbol at the encoder and the decoder, leading to an erroneous reconstruction of *all* subsequent reconstructed symbols at the decoder. Thus, in the communication of predictively encoded media, unlike non-predictively encoded media, it is *imperative* that adequate channel coding (in the form of FEC, ARQ, or any other method) is employed so as to prevent catastrophic decoding failure.

This paper addresses the elimination of the problem of predictive mismatch in the context of transmission over the Internet and transmission over the wireless medium. Firstly, it is noted

Proceedings of the Data Compression Conference (DCC'03)
1068-0314/03 $17.00 © 2003 **IEEE**

that the fragility of predictively encoded media is accentuated by the use of variable length codes for source compression in practical video encoding algorithms. Secondly, streaming media data-units have strict delivery deadlines. Consequently, even in the absense of errors, error-propagation may occur if a data unit arrives after its delivery deadline. This may occur, for example, due to limitations on the instantaneous bandwidth available on the server-receiver link. Thirdly, both channels exhibit burst error/erasure characteristics. Transmission over the Internet is reliable most of the time (the average packet drop probability on the Internet is $< 2\%$,[1]), however, periods of congestion in an intermediate router, could increase the packet drop probability to as much as 40% for a short interval of time. Similarly, with appropriate channel coding, the wireless channel functions at bit-error rates of 10^{-9} most of the time, however, during periods of deep fading, this probability can be as high as 10^{-4}.

Conventional methods for communication of media over lossy channels consist of approaches that use FEC [3] or ARQ [4], or both [5, 2], to protect against channel errors [1]. While ARQ is an efficient way of combating channel errors, it is inherently unsuited to certain applications (such as real-time streaming and multicasting). In FEC techniques, it is possible to choose the amount of protection based on the average loss characteristics of the channel. However, this implies that FEC would fail occasionally (due to the burstiness of the channel errors). While the break-down of FEC techniques is not critical to the communication of non-predictively encoded media, this could lead to severe degradation in performance if the media is predictively encoded.

One way to avoid this problem is to protect the media being transmitted based on the worst case characteristics of the channel. While heavily protecting critical parts of the stream, such as motion vectors and control information is plausible, heavily protecting the entire stream seems wasteful, especially if error bursts occur only rarely. However, even if only the critical information in a stream is heavily protected, loss of the residual information could lead to severe degradation in the quality of the reconstructed stream. This paper proposes a solution to precisely this problem by providing a second layer of protection along with primary channel coding techniques such as FEC or ARQ. The proposed solution assumes that motion vectors and other control parameters are decoded correctly at the receiver, while there could be errors in the decoding of the residual information. An encoding mechanism which eliminates error propagation in this scenario is proposed.

Specifically, the proposed algorithm facilitates the elimination of predictive mismatch by offering additional flexibility that is not present in conventional video coding. The proposed approach, however, does not prescribe any particular method to eliminate mismatch. This, in general, would be based on the specifics of the application at hand. Thus, we defer the transmission problem for future work. Our goal in the current paper will be the design of an efficient algorithm that facilitates recovery from predictive mismatch.

2 Previous Work

Numerous error-concealment techniques have been proposed in literature [6, 7]. However, most of these techniques accomplish their goal by using models of natural images. Thus, the performance of these techniques is limited by the assumptions made by their respective models. One technique that does not make restrictive assumptions on the statistics of the reconstructed

[1]The papers cited are an extremely small subset of those published.

video is the SP frame functionality of the H.26L algorithm. This method attempts to eliminate predictive mismatch by allowing special SP-frames to be predicted using one of multiple possible predictors. However, this solution precludes the scenario where each of the multiple predictors might have parts that are in error. Moreover, using SP-frames causes a severe penalty in terms of bit-rate, even when there are no errors, i.e. the solution does not scale with the number of errors.

In the present paper, we propose a framework for alleviating the problem of drift in predictively encoded streams, without overly sacrificing compression efficiency. The key underlying concept is that joint source-channel coding of predictively encoded media belongs to a class of predictive coding problems [8] that can be posed as a variant of the well-known Wyner-Ziv problem [10]. Accordingly, the proposed approach is based on the use of coset code constructions, which have been shown to perform close to the information-theoretic bound for the Wyner-Ziv problem [11, 12, 13]. Further, the work of [9] proves theoretically, that asymptotically, framing predictive coding as a side-information problem does not entail any loss in performance for Gauss-Markov processes.

The aforementioned literature forms the basis of the proposed video coder.

3 Illustrative Example

This section explains the proposed methodology with the help of a simple illustrative example. Consider the communication of a predictively encoded process from a server to a reciver over a delay-prone lossy link. Thus, data units communicated on the link are liable to arrive at the decoder past their delivery deadline or never arrive at the decoder at all. The stipulation of real-time streaming or multicasting, on such a link, further limits the scenario to "one-shot" transmission.

The server wishes to communicate a first-order process $\{x_k\}$, such that $x_k = x_{k-1} + n_k$ where $x_k \in \mathcal{Z}, n_k \in \mathcal{Z}, k \in \mathcal{Z}^+$ to the receiver. In the present example, the process $\{x_k\}$ satisfies the constraint,

$$|x_k - x_{k-1}| < d/2, \;\; d/2 \in \mathcal{Z}^+. \tag{1}$$

Thus, successive samples of $\{x_k\}$ are correlated. The server injects the symbol $t_k = x_k - x_{k-1}$, at time instant k, on the channel. Owing to the lossy nature of the server-receiver link, there is no guarantee that t_k will arrive at the receiver before its delivery deadline. In the event, the receiver does not receive t_k, it reconstructs $\widehat{x}_k = \widehat{x}_{k-1}$ (all decoder reconstructions are denoted with a hat symbol ($\widehat{}$)). If the receiver acquires t_k before its delivery deadline, it reconstructs x_k as $\widehat{x}_k = t_k + \widehat{x}_{k-1}$, where $\widehat{x}_{k-1}$ denotes the receiver's reconstruction of x_{k-1}. If $\widehat{x}_{k-1} = x_{k-1}$, x_k is reconstructed without any error. On the other hand, if $\widehat{x}_{k-1} \neq x_{k-1}$ (due to channel errors at time instants $j \leq k-1$), $\widehat{x}_k \neq x_k$, leading to a distorted reconstruction of x_k at the receiver. Moreover, this distortion is propagated over time to the receiver's reconstruction of each subsequent symbol $\widehat{x}_j$, $j > k$, even if the receiver receives the difference symbols corresponding to these samples correctly. This phenomenon is referred to as predictive mismatch. Our aim in the present paper is to ensure that even if a particular symbol is reconstructed erroneously, the error introduced is not propagated indefinitely.

Before describing the solution in the context of the above example, we introduce some terminology. We identify some samples in the sequence $\{x_k\}$ as 'peg samples' and associate an 'epoch' with each peg sample. For example, every tenth sample could be defined as a peg sample, i.e. x_j is a peg sample, if $j \ mod \ 10 = 0$. The epoch of a peg sample is defined as the set of samples in between the current and the preceding peg sample. Our aim is to ensure that errors introduced in the reconstruction of samples, is propagated no further than the nearest peg sample.

In the proposed approach, for a peg sample k', the encoder not only transmits $t_{k'} = x_{k'} - x_{k'-1}$ to the receiver, it also transmits $c^m_{k'} = x_{k'} \text{mod } md$, for some integer m. The coefficient $c^m_{k'}$ is refered to as the coset index of $x_{k'}$. As we shall show, coset index $c^m_{k'}$ is capable of correcting m erasures in the receiver's reconstruction, *irrespective of the position of the r erasures.* Thus, at the end of any epoch, the receiver can request for a particular coset index from the server, depending upon the number of erasures that the receiver encountered during that epoch. Alternately, the server can estimate the number of erasures experienced by the receiver in a particular epoch and transmit an appropriate coset index at the end of the epoch.

Next, we describe the decoding procedure for the peg samples. Consider the case in which the receiver had r erasures between peg samples k'' and k' and $\widehat{x}_{k''} = x_{k''}$. Thus, from Equation 1, at time instant k', the receiver's reconstruction of $\widehat{x}_{k'}$ satisfies

$$|\widehat{x}_{k'} - x_{k'}| < rd/2. \tag{2}$$

The receiver requests coset index $c^r_{k'}$ from the server and refines $\widehat{x}_{k'}$ as

$$\bar{x}_{k'} = \arg \min_{y:\, y \text{mod } rd = c^r_{k'}} |y - \widehat{x}_{k'}| \tag{3}$$

It is noted that Equation 3 quantizes $\widehat{x}_{k'}$ to a point in the set $\{\cdots, c^r_{k'} - rd, c^r_{k'}, c^r_{k'} + rd, \cdots\}$ using a nearest neighbor quantizer. Next, we show that $\bar{x}_{k'} = x_{k'}$ and thus, the propagation of error is curtailed. Note that $c^r_{k'} = x_{k'} \text{mod } rd$. Thus, from $c^r_{k'}$, the receiver knows that $x_{k'} \in \{\cdots, c^r_{k'} - rd, c^r_{k'}, c^r_{k'} + rd, \cdots\}$. Thus, as long as the receiver has an estimate of $x_{k'}$ which is within distance $rd/2$ of x_k, it can decode x_k perfectly by performing the quantization operation (Equation 3). Also, from Equation 2, $\widehat{x}_{k'}$ *is guaranteed* to be within distance $rd/2$ from $x_{k'}$. Thus, the receiver is able to decode x_k perfectly, thus thwarting the propagation of error.

It is also noted that in the aforementioned solution, the encoder need not know the r symbols that were erased. In the multicasting scenario, the receiver ascertains the number of symbols that it did not receive in any epoch and subscribes to an appropriate coset index from the server. In real-time streaming, the server determines the average number of symbols that were not received by the receiver and transmits an appropriate coset index to the receiver.

We note that, using fixed rate codes, the rate of encoding of symbol $c^r_{k'} = x_k \text{mod } rd$ is $\log_2(rd)$ bits. If we had access to a hypothetical omniscient server which had knowledge of the receiver's reconstruction of $\widehat{x}_{k'}$ and chose to correct it by transmitting $x_{k'} - \widehat{x}_{k'}$, then this server would use $\log_2(rd)$ bits as well (follows from Equation 2). Hence, the proposed approach performs as well as the omniscient server, while vastly simplifying the transmission procedure.

To recap, using the above example, we have demonstrated the efficacy of using coset codes for error recovery in predictively encoded sequences. Of course, the illustration assumed that the process $\{x_k\}$ satisfies Equation 1. This, in general, cannot be ensured. Nonetheless, as

shown in [9], asymptotically, if vector Gauss-Markov processes are considered, Equation 1 can be relaxed. Also, the coset code used in the preceding example is relatively simple. In the design of a practical encoder, we build coset codes using powerful low-density parity check codes (LDPCs), which, in the general setting of statistical correlation between samples of $\{x_k\}$, perform much better.

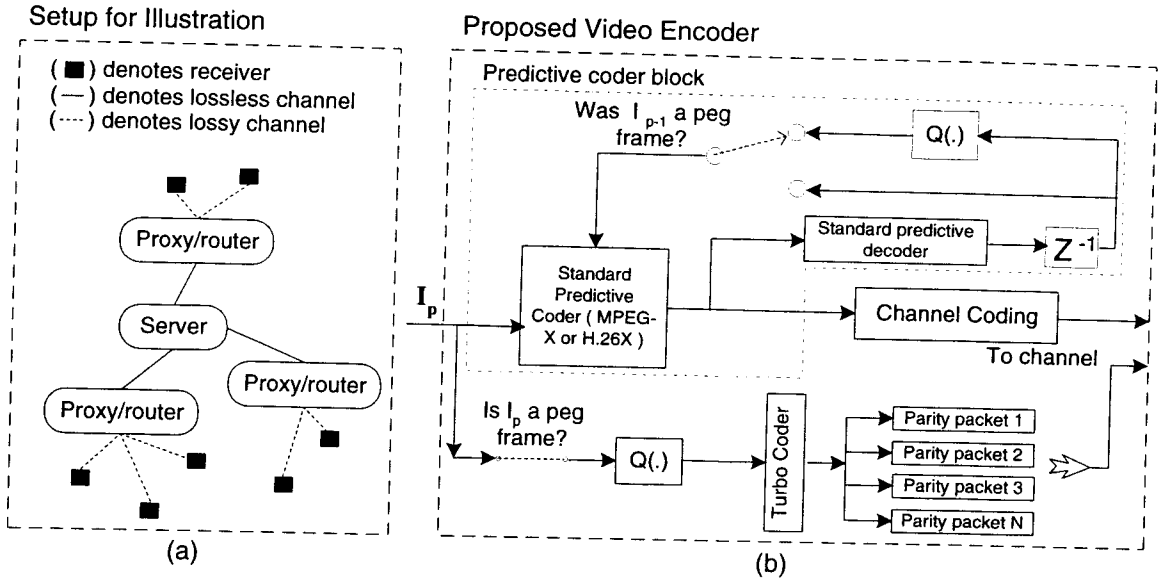

Figure 1: Figure (a) depicts the setup used in the example illustration in Section 3. Figure (b) shows a block diagram of the proposed encoder (Section 4).

4 Proposed Encoder

The encoding approach for video closely mimics the illustrative example from the previous section. Specifically, we define 'peg frames' similar to the peg samples, where drift errors in the decoder can be corrected. The epoch of a peg frame is defined similarly as well. We modify the H.26L, TML 8 video encoder to incorporate the proposed functionality. It is desirable to make minimal changes within H.26L encoder/decoder structure in order to ensure standard compatibility of the proposed framework.

Consider a video sequence that is encoded a priori to the streaming session and stored. The j^{th} frame of the video sequence is denoted as I_j, the j^{th} reconstructed frame is denoted as $\widehat{I}_j$, and the residual between I_j and I_{j-1} is denoted as P_j. The first frame, I_0, is intra-encoded, while all subsequent frames, I_j, $j > 0$, are encoded as P frames. Since errors introduced in bi-directionally encoded frames is not propagated, we consider only the P frames for clarity. As stated in Section 1, the motion vectors and the control information are adequately protected to ensure that these can be reconstructed at the receiver. The proposed approach corrects for errors in the reconstructed video from the loss of residual information.

The H.26L encoder and decoder are unaltered for all frames apart from the peg frames. Each reconstructed peg frame, $\widehat{I}_p$, is *requantized*using the H.26L encoding quantization scale A. We denote this requantized peg frame as $Q(\widehat{I}_p)$. The next P frame is predicted using this

requantized frame as the reference frame (Figure 1(b)). In the absence of any channel errors, the only loss in performance over an un-modified H.26L encoder is due to the re-quantization of the peg frames. As we shall show in the experimental results, this loss is small.

During encoding, the encoder also generates the coset information for each peg frame, to which a decoder can subscribe during the streaming session, in case channel errors were encountered during the epoch of a peg-frame.

The reconstructed peg frame $\hat{I}_p$ is encoded using a rate 1/2 turbo code. The H.26L forward transform for each 4×4 block of the peg frame is taken. This transformed image is then *requantized* to yield $Q(\hat{I}_p)$. Following this, all the DC transform coefficients of the transformed image are appended to form a vector $\mathbf{v}^0$. Similar vectors are generated for each frequency, to yield 16 vectors $\mathbf{v}^0, \mathbf{v}^1, ..., \mathbf{v}^{15}$. Each vector $\mathbf{v}^j$ is converted to a binary stream, which is encoded using a turbo encoder. Figure 2(a) shows the turbo encoding structure. Interleavers π_1 and π_2 randomize the transform coefficients so that the decoder sees the channel burst errors as independent bit errors.

The output of the turbo encoder, when $\mathbf{v}^j$ is input, consists of two parity streams $\mathbf{p}_1^j$ and $\mathbf{p}_2^j$. Consider the parity stream $\mathbf{p}_i^j$, $i = 1, 2$. This stream is punctured to form multiple packets. For example, bits at position i, such that $i \bmod M = m$ constitute one packet, where M is the total number of packets generated. This is done for both parity streams for each frequency.

Thus, side-information encoding yields a set of parity packets for each frequency in each peg frame. During streaming/multicasting, the receiver requests an appropriate subset of these packets (or none at all), depending on the transmission errors it encounters in the epoch corresponding to peg frame I_p.

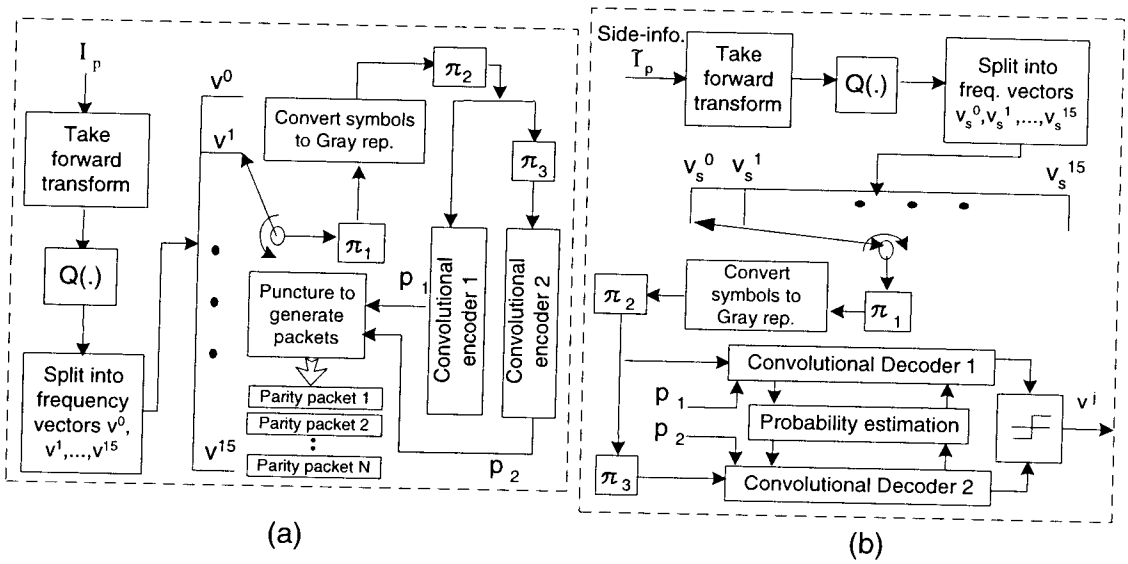

Figure 2: Figure (a) shows a block diagram of the turbo encoder. Figure (b) shows a block diagram of the turbo decoder.

5 Decoding Algorithm

All non-peg frames are reconstructed as standard H.26L frames. Consider the decoding of peg frame $\widehat{I}_p$. In the absence of any errors in the epoch of $\widehat{I}_p$, the receiver does not subscribe to any coset packets. The peg frame is decoded (as in H.26L), requantized, and used as a predictor for future frames. In the event of errors, the receiver subscribes to a subset of coset packets, depending on the amount of errors it experienced. The peg frame is reconstructed using the H.26L decoding algorithm to yield $\tilde{I}_p$. Frame $\tilde{I}_p$ is not equal to the encoder's reconstruction, $\widehat{I}_p$ due to channel errors. The receiver reconstructs $\widehat{I}_p$ using the subscribed coset packets in conjunction with the side-information $\tilde{I}_p$.

Turbo side-information decoding proceeds by first taking the forward transform of the erroneous peg frame $\tilde{I}_p$.

Sixteen side-information vectors $\mathbf{v}_s^0, \mathbf{v}_s^1, ..., \mathbf{v}_s^{15}$ are generated from the frequency coefficients of $\tilde{I}_p$, in a manner analogous to that used at the encoder. Decoding of $\mathbf{v}^j$ from $\mathbf{v}_s^j$ is performed independently of $\{\mathbf{v}^k : k \neq j\}$. Turbo decoding of $\mathbf{v}^j$ is performed using the punctured coset index stream and $\mathbf{v}_s^j$. We assume that each symbol of $\mathbf{v}^j$ is i.i.d distributed. With a slight abuse of notation, we denote the joint-distribution of a symbol of $\mathbf{v}_s^j$ and $\mathbf{v}^j$ as $f(\mathbf{v}_s^j, \mathbf{v}^j)$. It is noted that the turbo decoder requires knowledge of the joint distribution $f(\mathbf{v}_s^j, \mathbf{v}^j)$ to perform decoding. Empirical computation of this distribution requires knowledge of $\mathbf{v}_s^j$ and $\mathbf{v}^j$. The decoder, however, only has knowledge of $\mathbf{v}_s^j$. The estimation of $f(\mathbf{v}_s^j, \mathbf{v}^j)$ is described in the following section.

Next we describe a constraint on the turbo decoder that helps to improve the efficiency of the decoder. Consider a residual frame P_j in between peg frames I_{p-1} and I_p. Let there be errors in a subset of macroblocks of the receiver's reconstruction of P_j. These errors propagate over time. The receiver is capable of tracking these errors as they spread. By doing so, the decoder is able to track the subset of macroblocks of peg frame I_p where coefficients are in error. We denote the subset of coefficients which the decoder has identified as possibly erroneous by $\mathcal{S}$. While performing side-information decoding for frame $\widehat{I}_p$, the turbo decoder sets the a priori extrinsic probability for all coefficients $k \in \mathcal{S}^c$ to 1, since it knows that these coefficients are not in error, i.e. $P(\widehat{v}_s^j(k) = \widehat{v}^j(k)) = 1$. The computation of the a priori probabilities of the remaining coefficients is discussed in the following section.

6 Probability Estimation

In [12, 13], the authors propose the use of turbo codes for the Wyner-Ziv problem. Since they work with synthetically generated data, they assume that the joint-distribution of the side-information and the discrete source symbols is known to the decoder. Video streams, however, exhibit non-stationary behavior. Thus, the joint-statistics of the side-information $\mathbf{v}_s^j$ and source vector $\mathbf{v}^j$ are not known to either the encoder or the decoder (since the encoder has access only to $\mathbf{v}^j$ and the decoder has access to only $\mathbf{v}_s^j$). This section describes a decoding approach where the turbo decoder estimates this distribution as it decodes the source symbols.

In [14], the authors propose an algorithm for achieving the Slepian-Wolf bound where the joint distribution of the source and the side-information is obtained from the data itself. The solution of [14] cannot be directly used for the Wyner-Ziv problem. Intuitively, this is because the Euclidean distance between symbols of $\mathbf{v}^j$ and $\mathbf{v}_s^j$ can be very different from the Hamming

distance between the two symbols. However, if the Gray code representation of the symbols is used, there is correlation between the Euclidean distance and the Hamming distance. Thus, with a Gray code representation of the code words, we can use the solution of [14] for the Wyner-Ziv problem. Hence, in the proposed solution, the turbo decoder assumes that the streams $\mathbf{v}_s^j$ and $\mathbf{v}^j$ are correlated in the bit domain, and updates the conditional densities after each iteration. This does entail some loss in the performance of the code, however, it significantly increases the robustness of decoding. This loss is characterized in the experimental results section.

7 Results

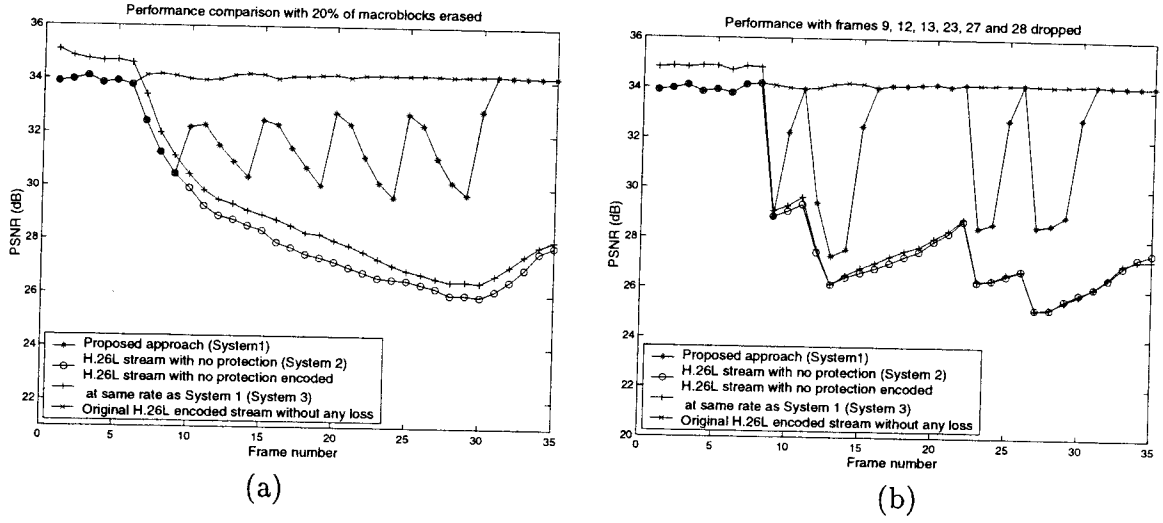

Figure 3: (a) compares the performance of the proposed approach with unprotected streams with 20% of the macroblocks erased between frames 7 and 30. (b) compares the performance with frames 9, 12, 13, 23, 27 and 28 erased.

It is noted that the proposed approach facilitates the elimination of predictive mismatch. Efficient elimination of mismatch for a practical application entails the usage of channel codes in conjunction with the proposed approach. Therefore, the efficacy of the proposed solution in a practical application would be influenced by numerous other factors as well. In light of this, we report results on the performance of a basic system based on the proposed framework, so as not to shadow the results by other decisions made in the streaming system.

Experimental results are reported for the Y component of the 'cheers' (CIF) video sequence. The sequence was encoded using the H.26L encoder with QP=16 at 30 fps. Peg frames were defined at intervals of 5 frames. Thus, the epoch of each peg frame was five video frames. The video sequences were encoded in the IPPP... format. The quantization reconstruction routine in the H.26L encoder was altered to reconstruct each point precisely (as opposed to the linear approximation made in the original H.26L code), this lead to an improvement in performance of 0.1 dB.

Two rate 1/2 convolutional coders with the generator matrix $G(D) = \left[1 \ \frac{1+D^2+D^3+D^4}{1+D+D^4}\right]$ constituted the turbo code. Interleavers π_1, π_2, π_3 were all S-Random interleavers with appropriate parameters (depending on the sequence length). Uniform puncturing with a period of 10 bits

was performed to generate the parity packets. Turbo decoding was terminated after 30 iterations.

Experiments were performed to assess the loss in performance due to the re-quantization of peg frames. With an epoch duration of five frames, re-quantization led to an approximate loss of 0.2 dB per video frame. This is the loss which is incurred over standard H.26L encoded video in the absence of errors. If the duration of each epoch is increased, the average loss decreases.

Our next experiment simulates burst errors where 20% of data in each macroblock is erased. This corresponds to transmission over the Internet where 20% of the transmitted packets are dropped, or over a wireless channel where bit errors in packets render 20% of the packets undecodable. Figure 3(a) compares the performance of the proposed approach in this case. Every 5^{th} frame represents a peg frame, the error burst starts at frame 7 and ends at frame 30. As can be seen from the graph, after the end of the error burst, the reconstructed PSNR of the proposed approach is the same as that of the H.26L encoded stream. Thus, the system recovers from the errors introduced in the stream, and consequently, there is no propagation of error. Assuming that the pdf $f(\mathbf{v}_j^s, \mathbf{v}_j)$ is known at the decoder, the coset information amounted to 442 *Kbytes*. If the estimation routine from Section 6 is used, 556 *Kbytes* of parity information was required for correct decoding. Thus, the inaccuracies of the estimation procedure required 26% increase in the amount of transmitted coset information. The total size of the H.26L encoded stream was 2.8 *Mb*. Thus, coset information constituted 16.5% of the total transmitted bits. Further, it accounted for 18.7% of the transmitted residual information.

Figure 3(a) compares the performance of the proposed framework with an unprotected H.26L stream (System 2) that is encoded at the same rate as the H.26L stream of the proposed method. Comparisons are also made with an H.26L stream coded at the same total rate as the sum of the H.26L stream and the coset information in the proposed system. For reference, the streams are compared with the original H.26L decoded stream at the encoder (in which there are no errors).

Our next experiment simulates transmission of pre-encoded video over a wireless channel with short bursts of deep fading. Owing to the fact that residual macroblocks are not protected very well, during periods of deep fading, a few bit errors in each macroblock render the entire macroblock undecodable. Thus, all macroblocks from a subset of frames of the video sequence were completely erased. For this case, if the distribution $f(\mathbf{v}_s^j, \mathbf{v}_j)$ is assumed known, 412 *Kbytes* of parity information is required for correct decoding. The amount of coset information required when $f(\mathbf{v}_s^j, \mathbf{v}_j)$ is estimated while decoding was 528 *Kbytes*. Thus, estimation led to a loss of 28% for this case. The transmitted coset information amounted to 15.7% of the total transmitted bits or 17.2% of the transmitted residual information for this case. Comparisons similar to those in Figure 3(a) are also shown in Figure 3(b).

In summary, the experiments demonstrate that the proposed approach is able to recover from errors in the event of channel failure, thus thwarting the propagation of error.

8 Conclusion

This paper proposes a solution for correcting the problem of drift in predictively encoded sequences when the residual information is not protected with respect to the worst case performance of the channel. It is also noted, that the solution can be employed for the scenario when the motion vectors are lost as well. The decoder can estimate the motion vectors using models

of natural motion, reconstruct the frame with missing motion vectors using the estimated vectors and use this for side-information decoding. Since the efficacy of this solution relies heavily on the choice of the tracking algorithm, we do not provide simulation results for this case.

As pointed out in the Introduction, when the proposed approach is used in conjunction with primary channel coding techniques such as FEC or ARQ, it acts as a second layer of protection. Our aim in this paper was the design of an encoding mechanism for this problem, we defer the treatment of a rate-distortion optimized transmission mechanism using the proposed framework for future work.

Acknowledgements

We wish to thank Dr. Philip Chou, Microsoft Research for bringing the use of Gray codes for compression problems to our notice, and Ashish Jagmohan, University of Illinois, for his help through out the course of this work.

References

[1] http://www.internettrafficereport.com

[2] W-T. Tan and A. Zakhor, "Video multicast using layered FEC and scalable compression," *IEEE Transactions on Circuits and Systems for Video Technology*, Volume 11, Issue 3 , March 2001, pages 373 -386.

[3] B. Kurceren, J. W. Modestino, "A joint source-channel coding approach to network transport on digital video," *Proceedings of IEEE INFOCOM* , Volume 2 , 2000, pages 717 -726.

[4] P. A. Chou and Z. Miao, "Rate-distortion optimized sender-driven streaming over best-effort networks," *IEEE Workshop on Multimedia Signal Processing*, Cannes, France, pages 587-592, October 2001.

[5] P. A. Chou, A. E. Mohr, A. Wang, and S. Mehrotra, "FEC and Pseudo-ARQ for Receiver-driven Layered Multicast of Audio and Video," *IEEE Data Compression Conference*, Snowbird, UT, pages 440-449, March 2000.

[6] S. S. Hemami, R. M. Gray, "Subband Coded Image Reconstruction for Lossy Packet Networks," *IEEE Transactions on Image Processing*, April 1997.

[7] D. S. Turaga, T. Chen, "Model-based error concealment for wireless video," *IEEE Transactions on Circuits and Systems for Video Technology*, volume 12, issue 6, June 2002, pages 483 -495.

[8] A. Jagmohan, A. Sehgal, and N. Ahuja, "Predictive Coding Using Coset Codes," *IEEE Int. Conf. on Image Processing*, to appear.

[9] A. Sehgal, A. Jagmohan, and N. Ahuja, "Layered Predictive Coding based on the Wyner-Ziv Problem," *IEEE Int. Conf. on Communication Systems*, under review.

[10] A.D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Info. theory*, vol. 22, pp. 1–10, Jan. 1976.

[11] S.S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (discus): design and construction," in *IEEE Data Compression Conference*, 1999, pp. 158–167.

[12] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *IEEE Data Compression Conference*, 2002, pp. 252–261.

[13] Y. Zhao and J. Garca-Fras, "Data Compression of Correlated Non-Binary Sources Using Punctured Turbo Codes", *Proc. DCC'02*, April 2002, Snowbird, UT.

[14] J. Garca-Fras and Y. Zhao, "Compression of Correlated Binary Sources Using Turbo Codes", *IEEE Communications Letters*, pages 417-419, October 2001.