PRISM: A VIDEO CODING ARCHITECTURE BASED ON DISTRIBUTED COMPRESSION PRINCIPLES

by

Rohit Puri and Kannan Ramchandran

PRISM: A VIDEO CODING
ARCHITECTURE BASED ON
DISTRIBUTED COMPRESSION PRINCIPLES

by

Rohit Puri and Kannan Ramchandran

ELECTRONICS RESEARCH LABORATORY

PRISM: A Video Coding Architecture Based on Distributed Compression Principles*

Rohit Puri   Kannan Ramchandran
{rpuri, kannanr}@eecs.berkeley.edu
EECS Department, University of California, Berkeley,
211 Cory Hall #1772,
Berkeley, CA-94720.
Ph: (510)-643-4034 Fax: (510)-642-9160

Abstract

We introduce PRISM (Power-efficient, Robust, hIgh-compression, Syndrome-based Multimedia coding), a video coding paradigm underpinned by the principles of distributed source coding from multi-user information theory. PRISM, which rests heavily on *channel coding concepts*, represents a radical departure from current state-of-the-art video coding architectures, including those driving popular standards like MPEG and H.263, that are based on a motion-compensated prediction framework. These are hampered by: (i) a rigid computational complexity partition between encoder (heavy) and decoder (light); (ii) high fragility to drift between encoder and decoder in the face of prediction mismatch, e.g. due to channel loss; and (iii) being part of a standards environment requiring a bulky encoding format that makes innovating within the standard cumbersome. In stark contrast, PRISM's architectural goals are: (i) to have a flexible **distribution of computational complexity** between encoder and decoder without compromising compression efficiency; (ii) to have **inbuilt robustness** to drift between encoder and decoder due to channel loss; and (iii) to have a **light yet rich encoding syntax** that can be standardized in a way that permits for creative growth and unencumbered innovating within the standard.

In this paper, we describe a specific implementation of PRISM corresponding to a near **reversal** in codec complexities, with a rough swap in encoder and decoder complexities vis-a-vis today's codecs. This leads to a novel light encoder and heavy decoder paradigm. Specifically, this involves *moving the expensive motion search component of the video codec from the encoder to the decoder*. Simultaneously, PRISM offers a low-latency, easily-tunable natural immunity to the drift problem. In a nutshell, the ideal is to marry the encoding lightness and robustness of intraframe coding (so-called motion JPEG) with the compression-efficiency of interframe coding (full-motion MPEG). Our preliminary simulations for a specific block-DCT based implementation of PRISM confirm the promise of realizing this ideal. Despite being based on very simple codes, our preliminary experiments reveal the strikingly superior performance potential of PRISM to state-of-the-art prediction-based codecs under the regime of packet/frame drops, as well as how surprisingly closely, considering this is a new untested paradigm, it approaches their subjective performance even in the zero-channel-loss regime.

1   Introduction

Today's digital video coding architectures have been driven primarily by the "downlink" television broadcast model of a heavy encoder and a multitude of light decoders. However, with the expected proliferation

---

of multiple video sources ranging from digital cameras to low-power video sensor networks to multimedia-equipped cellular phones to Webcams, the days of typecasting digital video transmission as a predominantly downlink experience are over. In a typical application scenario, we expect future multimedia systems to use multiple video input and output streams to enhance user experience. These streams need to be captured using a network of distributed devices and transmitted over a bandwidth-constrained, noisy wireless transmission medium, to a central location for processing, with the goal for example, of creating high-resolution video using inexpensive cameras. This is typical of a new class of "uplink" rich media applications having very different architectural requirements than those of the traditional downlink video delivery model. Specifically, the new demands include some or all of the following:

- low-power and light-footprint encoding due to limited battery power and/or device memory;

- high compression efficiency due to both bandwidth and transmission power limitations; and

- robustness to packet/frame loss caused by channel transmission errors.

Current video coding paradigms fail to simultaneously address these demanding requirements satisfactorily. Motion-compensated predictive coding or full-motion inter-frame video coding approaches that are part of popular standards like H.26x and MPEG [2, 3, 4] achieve state-of-the-art compression efficiency, but fail to meet the other two criteria, as they are computationally heavy at the encoder (primarily due to motion-search) while also being very fragile[1] to packet losses. Alternatively, intra-frame video coding methods or so-called motion-JPEG approaches (where each of the individual frames is encoded as a still-image) have low computational complexity, and are relatively robust to packet drops due to the lack of dependencies among frames, but they take a relatively high hit in compression efficiency and resulting transmission power. This raises the interesting question of whether it is possible to architect a new video coding paradigm that is driven to attain all of these requirements simultaneously. More succinctly, is it possible to achieve *full-motion-MPEG-like compression efficiency at motion-JPEG-like encoding complexity and robustness?*

We submit that an optimistic answer to this question calls for a serious questioning of the fundamental principles underpinning current video coding approaches and a possible rehaul of current architectures. It is in this context that we introduce PRISM (Power-efficient Robust hIgh-compression Syndrome-based Multimedia coding), a new video coding paradigm founded on the principles of distributed source coding [5, 6].

---

[1]In order to exploit the temporal correlation in video sequence, motion compensated prediction is used to come up with a "good" predictor based on the previous frame for the current frame that is to be coded. The residue error between the current frame and the predictor is what is actually encoded and transmitted for the current frame. This introduces dependencies between the various coded units leading to fragility. If the previous frame is lost during transmission then the availability of the coded unit for the current frame is of no use at the decoder.

The rest of this paper is organized as follows. We describe the basic philosophy and architecture behind PRISM in Section 2, and discuss related work. In Section 3 we illustrate the key intuition behind PRISM pertaining to the underlying theoretical concept of distributed source coding. In Section 4 we describe a specific implementation of the PRISM framework, inspired by this theory, corresponding to a block-motion, block-DCT framework, and discuss in-depth some of the key implementational issues. Section 5 revisits the features of the PRISM framework, and Section 6 details the simulation results. Finally, in Section 7 we offer conclusions and directions for further research.

2   Philosophy and Architecture of PRISM

Before outlining the architectural goals of PRISM, let us recall that today's inter-frame motion-compensated predictive coding framework, while being critical to achieving state-of-the-art compression performance, is hampered by:

- a rigid computational complexity partition between encoder (heavy) and decoder (light) where the encoding complexity is dominated by the motion search operation needed to strip temporal redundancy from video frames[2], whereas the conventional video decoder is a relatively lightweight device operating in a "slave" mode to the encoder;

- fragility to synchronization or "drift"[3] between encoder and decoder in the face of prediction mismatch e.g. due to channel loss, leading to well-known annoying motion-streak artifacts; and

- being part of a standards paradigm requiring a bulky encoding format, where the core intelligence needs to be "cast in stone" (e.g., precise motion-compensation method, motion accuracy, choice of what loop-filter to use in the compensation loop, etc. [2, 3, 4]), and enforced at the encoder for "spoon-feeding" to the decoder, leaving it with relatively little room for creativity within the standard.

    PRISM's architectural goals are accordingly three-fold:

---

[2]A "full-search" block motion estimation algorithm typically incurs approximately 65000 operations per pixel per second for a 30 frames per second video. The motion estimation module in a typical video codec takes between 75% to 90% of CPU time and is the single most computationally intensive part of the encoder. Motion estimation is also extremely demanding on the I/O transfer between CPU and memory, a notorious source of practical discomfort in terms of power consumption and speed.

[3]The drift problem in video coding is an artifact of the predictive coding framework. When, for some reason, the frame memories at the encoder and the decoder are not identical, then the residue error is encoded at the encoder off some predictor and decoded at the decoder off some other predictor. Scenarios like transmission losses, unequal machine precision at the encoder and the decoder etc. can lead to non-identical frame memories. The drift between the encoder and the decoder keeps accumulating and propagating and can lead to very displeasing visual artifacts. Drift between the encoder and the decoder can be corrected when they are synchronized by an intra-coded frame.

3

- to have flexibility in the **distribution of computational complexity between encoder and decoder** without compromising compression efficiency;

- to have **inbuilt robustness** to "drift" caused by loss of synchronization between encoder and decoder (e.g., due to channel loss); and

- to have a **light yet rich encoder syntax** that can be standardized while leaving far greater room for creative growth and unencumbered innovating within the standard than is possible today.
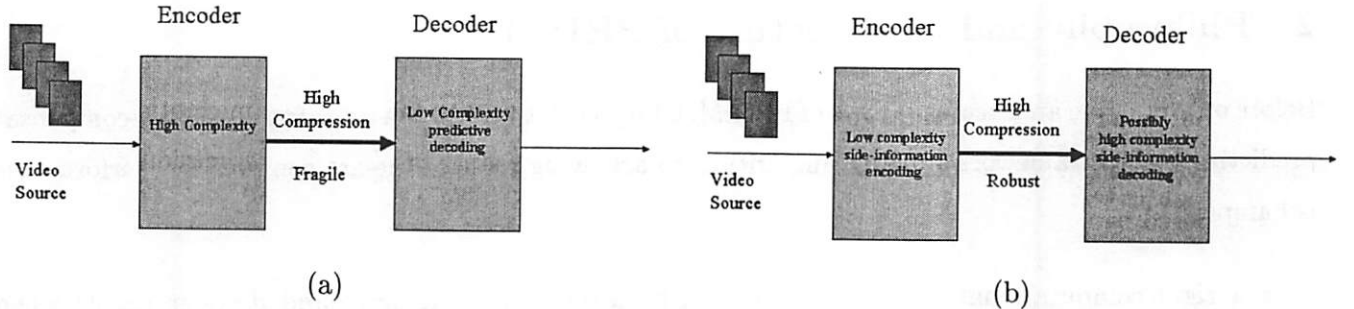

Figure 1: (a) Conventional video encoding architecture comprises of a high complexity encoder and a low complexity decoder. This has the features of high compression efficiency but also high fragility to channel noise. (b) One instance of the proposed new video coding paradigm (PRISM) comprising of a low complexity encoder and achieving the compression performance of the conventional framework. The goal is for System (b) to match the pure compression efficiency of System (a) with much-increased robustness to channel loss.

PRISM thus represents a robust, flexible-complexity, high-compression video encoding paradigm, in striking contrast with traditional video coding approaches. PRISM is founded on the principles of distributed source coding, and rests heavily on channel coding principles. Although this topic has been around in the form of asymptotic theorems in information theory for more than 30 years [5, 6], practical constructions that close the gap between theory and practice have been proposed only recently [7]. PRISM leverages these advances in distributed source coding in developing a novel approach to video coding that targets all the three architectural goals listed above. We briefly visit each architectural goal.

2.1 Distribution of complexity

A first goal is to allow for a much more flexible distribution of the computational complexity between encoder and decoder than is possible with today's architectures. PRISM targets this goal. For example, in uplink-rich media applications, it is desirable to move the bulk of the complexity from the complexity constrained encoder to the more capable decoder. More generally, it might be desirable to share the complexity burden between encoder and decoder more equally or in any desirable ratio as demanded by a specific application scenario.

PRISM targets this goal via the dominant-complexity component of motion search. In this paper, we will target the special case of having a maximally thin encoder in order to directly target the rich class of uplink video applications. This allows for a novel *near-reversal* in codec complexities, with encoder and decoding complexities being roughly swapped with respect to conventional codecs (see Figure 1). This is accomplished, as will be detailed, by *moving the expensive motion estimation component of the video codec from the encoder to the decoder* without loss of compression efficiency in theory (for appropriate signal model choices) and with acceptable loss of efficiency in practice.

Interestingly, it is possible to generalize this concept to arbitrary distribution of the motion search complexity between encoder and decoder with no loss of performance in theory for an interesting class of signal models [8], allowing for a flexible array of architectures that can be matched appropriately to specific applications of interest. These extensions will not be considered in this paper due to lack of space.

2.2 Robustness

A second architectural goal is to allow for far greater robustness to packet and frame drops than is possible with today's video codecs that are based on the fragile predictive coding framework. Predictive coding frameworks are fundamentally "broken" in this regard. It would also be desirable to have an easily-tunable robustness "knob" (rather than elaborate sets of bells and whistles) to match to specific noisy channel conditions.

PRISM targets this by dispensing with the predictive framework and exchanging it for a "universally robust" side-information based coding framework. Side-information source coding inherently consists of "good" source codes that are partitioned into 'good" channel codes (as will be elucidated shortly), and therefore has naturally inbuilt robustness to the synchronization loss issues that plague prediction-based codecs. PRISM also naturally comes with the easily-tunable "knob" to control the tradeoff between compression efficiency and noise-immunity. Specifically, PRISM is based on channel coding for the correlation "noise" between temporally adjacent video frames. In the face of channel noise, leading to packet drops, there will be a channel noise component to overcome in addition to the correlation noise. The key point is that the PRISM design depends only on the *sum of the two noise components*, and it does not care to distinguish between the individual components.

A valid question to ask is: how does the PRISM approach differ from that of conventional Forward Error Correction (FEC) based approaches that can be used to robustify video streams? There are two fundamental advantages to the PRISM approach. First, FEC-based solutions need large block-length channel codes, in addition to large block-length source codes, to attain peak performance. This leads to significantly high latencies that can be unacceptable for many real-time streaming applications. The PRISM approach on the other hand is a natural source-channel coding strategy that can attain reasonable

5

performance with relatively small block lengths. In fact, the implementation described in this paper incurs a very mild macro-block-level latency whereas FEC-based solutions incur latencies of the order of several macro-blocks or frames or Group Of Pictures (GOP's). Secondly, FEC-based solutions waste their resources if there is no channel loss: i.e., parity bits serve no useful purpose when the channel is clean. With the PRISM approach, on the other hand, even when there is no channel loss, there is an important "estimation-gain" component to be exploited (see [9] and Section 4.4, item 3) to improving the reconstruction quality. This is a particularly important feature at low bit rates.

2.3 Light Syntax

Thirdly, it would be desirable to have a much less bulky syntax than possible today. Recall that today's standards like MPEG and H.26L, by virtue of their being prediction-based, necessarily have to "cast in stone" the details of the prediction algorithm, such as the exact motion compensation method (e.g., block motion based), the exact nature (e.g., 4x4 block-based vs. 8x8 block-based) and precision of motion vectors (e.g. half-pixel vs. quarter-pixel based), the loop filter to be used in interpolation if any, etc. [4]. This is because the decoder has to operate in a prediction closed-loop. The process of innovating on the standards is also cumbersome, and has to go through a standardization bureaucratic chain. For example, to standardize an improved motion compensation method can be painful, time-consuming, and need the buy-in of a large number of standards body members.

In stark contrast, the PRISM encoding syntax, being based on distributed source coding principles, is naturally very light yet extremely rich. The bulk of the codec smarts are in the decoder. For example, the critical motion compensation method can be as wide open as block-based (as in commercial standards) versus optical-flow based [10] versus matching-pursuits based [11] or even object-oriented (as in MPEG-4), with the novelty that any of these is compatible with the encoding syntax. This would leave far greater room for proprietary innovations with full standards compatibility resulting in a natural complexity versus performance scalability. Smarter and harder-working decoders would have a natural performance advantage over less sophisticated ones, without the explicit need for approval by a standards body. In this regard, it is ironic to point out that the latest motion-compensation related advances commissioned by the ITU low bit rate standards committees for H.26L [4] (including multi-frame prediction [12], improved loop-filters, hybrid motion block-sizes, etc.) all fall within the direct scope of our proposed "reversed" PRISM architecture, with the twist that they are now responsibilities of the *decoder functionality* and need not be explicitly pencilled into the format.

In summary, since multimedia coding standards constrain the bit-stream syntax at the encoder, the fact that all the sophisticated signal processing tasks are performed at the PRISM decoder, results in an encoder syntax that is rich enough to accommodate a plethora of decoders that are all "syntax-compatible".

6

We emphasize that this is a completely different way to think, and it opens up the opportunity of a whole new set of creative algorithms/techniques for motion estimation, post processing and other critical signal processing tasks.

2.4 PRISM in a Network Configuration

It is valid to question the utility of shifting the complexity from the encoder to the decoder (or to share it arbitrarily) when in a codec solution, it is the *sum of these complexities* that is relevant. Specifically, if PRISM shifts the complexity to the decoder, isn't this going to make the sum codec complexity just as bad as before or worse, since it is typically desired to do both functionalities? As a way of addressing this legitimate question, we propose a network configuration for the PRISM codec as shown in Figure 2.

Here, the uplink direction consists of a transmit mobile station (or sensor node) employing the low-complexity PRISM encoder interfaced to a PRISM decoder in the base station or Access Point. The base station has a "transcoding proxy" that efficiently converts the PRISM bit-stream into a standard bit-stream (e.g., that output by a standard MPEG encoder). The downlink then consists of a receiving mobile station that has the standard low-complexity video decoder. Under this architecture, *the entire computational burden has been absorbed into the network device.* Both the end devices, which are battery-constrained, run power efficient and light encoding and decoding algorithms. In a nutshell, we propose a "half-duplex" format for uplink and downlink directions, suggesting the use of the PRISM format for uplink (to facilitate a light encoder) and a standard MPEG-like format for downlink (to leverage the existing light standard decoder).
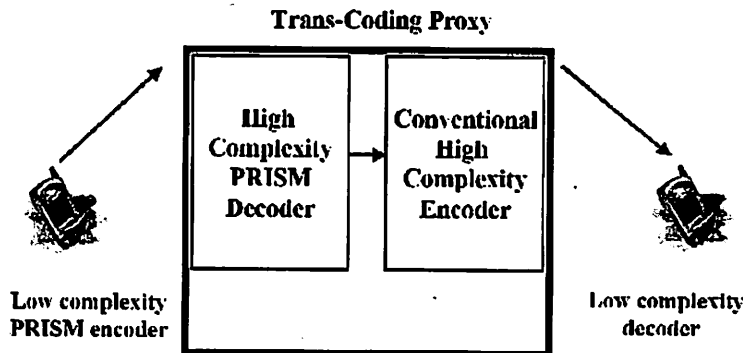

Figure 2: System level diagram for a network scenario with low complexity encoding and decoding devices.

2.5 PRISM Implementation: Motivation

In this work, we describe a specific instantiation of the PRISM framework based on a block motion-based DCT framework. Our motivation for resorting to this is two-fold. First, this is the most popularly used

framework in the video codec world, and allows for easy access to building blocks for the PRISM codec. Secondly, and more importantly, in light of the networking applications involving the PRISM transcoding architecture described in Section 2.4, it is critical to lighten the computational burden and latency of the transcoding proxy for streaming or other latency-critical applications. As we propose to transcode from the PRISM format to a standard MPEG-like format (see Figure 2), it is obviously desirable to keep the PRISM implementation as close to an MPEG-style implementation as possible. For example, in the "reversed" complexity setup of PRISM, the PRISM decoder will be entrusted with the motion compensation task. If this is based on a block-motion based framework, it eases the complexity and therefore the latency burden in transcoding to a block-motion MPEG format for use in the downlink transmission.

Finally, before we plunge into the details of PRISM, we hasten to point out that this paper is but a first step towards realizing the architectural dreams enlisted earlier. While the architectural goals enlisted for PRISM do stand on solid information-theoretical footing (this will be enunciated in Section 4.2), there is a big gap between theory and practice. This work represents but a first step towards narrowing that gap. We are heartened that our preliminary simulation results are promising (see Section 6), and validate the efficacy of the PRISM paradigm. Despite being based on very simple codes, our preliminary experiments confirm the strikingly superior performance of PRISM to state-of-the-art prediction-based codecs under the regime of packet/frame drops, while surprisingly closely approaching, considering this is a new untested paradigm, their subjective performance even in the zero-channel-loss regime. But several conceptual and implementational innovations needing much more in-depth study are in order before the full potential of PRISM can be realized. These are detailed in Section 7. We are optimistic however that this paper represents a promising start to this journey.

2.6   Related Work

PRISM is founded on the principles of distributed source coding, whose information theoretic origins go back to Slepian and Wolf in [5] and to Wyner and Ziv in [6] for the lossless and lossy cases respectively. Although this is over 30 years old, this theory is non-constructive and rests on asymptotic random coding arguments. The first constructive framework for generating practical codes for the distributed source coding problem, dubbed DISCUS (Distributed Source Coding Using Syndromes) was proposed in [7], where a coding construction based on trellis codes was also presented. Subsequently, more powerful code constructions for the distributed source coding problem based on turbo codes [13] have been presented by various researchers [14, 15, 16]. Recently, practical turbo codes based on the DISCUS framework that approach the theoretical bounds very sharply for Gaussian sources have been described in [17].

Application of distributed source coding to real-world images was first undertaken in [18], where the

problem of efficiently enhancing analog image transmission systems using digital side information coding was studied. Promising applications of distributed source codes for audio signals and sensor networks have been considered recently in [19] and [20] respectively. More directly related to this paper, the application of distributed compression ideas to video coding was introduced in [1], where the reversed complexity paradigm and superior robustness of PRISM were described. The video coding application of distributed source coding was also considered later in [21]. Recent work towards an information theory for the PRISM video framework presented here has been described in [8].

In the context of the proposed network transcoding architecture of PRISM, the idea of transferring the computational burden to the network was first presented in [22], where the task of motion estimation was essentially transferred from the video encoder to the network terminal. Our proposed PRISM framework however is different on three counts. First, PRISM is a natural joint-source-channel coding framework unlike the pure source coding framework of [22]. Secondly, while PRISM is founded on distributed source coding principles, the work of [22] was founded on the conventional predictive coding framework. Finally, the work of [22] is predicated on having network feedback. In PRISM, while network feedback is helpful, it is not required. To the best of our knowledge, PRISM represents the first self-contained competitive video codec built on distributed compression principles.

3 Basic Concepts

In this section, we first present a brief overview of the conventional video interframe predictive coding architecture. Following this, we introduce the basic concepts related to the problem of distributed source coding, and highlight the key intuition by means of an illustrative example which represents a microcosm of the PRISM framework. While this example may appear to caricature the workings of the PRISM codec, it nonetheless captures the salient features that we target in this work.

3.1 Background: Conventional Video Coding

Let us first provide a brief summary of the conventional video coding approach that has influenced the evolution and development of video coding standards such as H.26x and MPEG [2, 3, 4]. A video sequence is a collection of images (also called frames) in time. A typical video sequence exhibits high spatio-temporal redundancy with the temporal redundancy being particularly important. For the purpose of encoding, each of these frames is decomposed into non-overlapping 16x16 spatial blocks called **macro-blocks**. These are encoded primarily in the following two modes.

1. Intra-Coding (I) Mode: The intra-coding mode exploits only the spatial correlation present in the frame that contains the block being encoded. Since this mode does not exploit the temporal

correlation in video, it typically achieves *poor compression*. However, it incurs a *low encoding complexity* (spatial processing only; no temporal processing) and is *robust* in the sense that it is a self-contained description of the block being encoded that is not dependent on any other frame.

2. Inter-Coding or Predictive (P) Mode: In contrast to the intra-coding mode, the predictive or inter-coding mode is the "true" video compression mode. It exploits both the spatial and temporal correlation present in the video sequence, where the latter is done through the high-complexity motion estimation operation. Since this coding mode exploits temporal correlation, it achieves *high compression*. However, it also incurs *high encoding complexity* (primarily due to motion search) and is *fragile* in the sense that loss of the predictor renders the residue information useless from the point of view of decoding.

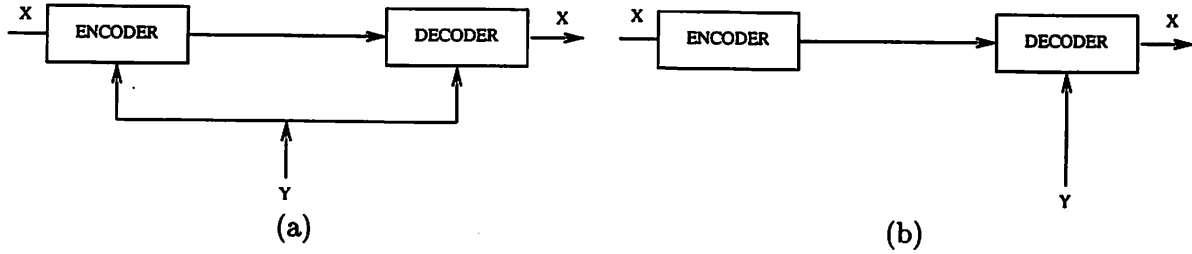## 3.2 Illustrative Example for Coding with Side Information

(a)   (b)

Figure 3: X and Y are correlated, length 3-bit binary data equally likely taking each of the 8 possible values, individually. The Hamming distance between the codeword for X and that for Y is at most 1. Y is first made available to the decoder using 3 bits. (a) Both encoder and decoder use the side information Y which is correlated to X. Here X can be encoded with 2 bits. (b) Only decoder accesses Y. Here too, X can be encoded using 2 bits.

Our goal here is to marry the high compression efficiency of the predictive-coding mode with the robustness and the low encoding complexity features of the intra-coding mode. To see how we can achieve this, it is instructive to examine the following example that was first presented in [7] (See Figure 3).

Let X and Y be length 3-bit binary data that can equally likely take on each of the 8 possible binary 3-tuples. However, X and Y are correlated random variables. The correlation between them is such that the Hamming distance between X and Y is at most 1. That is, given Y (e.g., [0 1 0]), X is either the same as Y ([0 1 0]) or off in the first bit ([1 1 0]) or off in the middle bit ([0 0 0]) or off in the last bit ([0 1 1]). The goal is to efficiently encode X in the two scenarios shown in Figure 3 so that it can be perfectly reconstructed at the decoder.

Scenario 1: In the first scenario (see Figure 3 (a)), Y is present both at the encoder and at the

decoder (Y can be made available to the decoder using 3 bits). Here X can be predicted from Y. The residue ($X \oplus Y$) or the error pattern of X with respect to Y takes 4 distinct values and hence can be encoded with 2 bits. This is the least possible (best) rate needed to encode X. The decoder can combine the residue with Y to obtain X. X is analogous to the current video block that is being encoded, Y is analogous to the predictor from the frame memory, the correlation between X and Y is analogous to the temporal correlation between successive video frames, and hence this mode of encoding is similar to **predictive coding**.

Scenario 2: In the second scenario (see Figure 3 (b)), Y has been made available to the decoder using 3 bits but the encoder for X does not have access to Y. However, it does know the correlation structure between them and also knows that the decoder has access to Y. What is the best that can be done in this case? Since this scenario is necessarily worse than the first scenario its performance is limited by that of the first scenario. The surprising answer, however, is that even in this seemingly worse scenario, we can achieve the same performance as in the first scenario. That is, here too, X can be encoded with 2 bits!

This can be done using the following approach. The space of codewords of X is partitioned into 4 sets each containing 2 codewords, namely, **Coset1** ([0 0 0] and [1 1 1]), **Coset2** ([0 0 1] and [1 1 0]), **Coset3** ([0 1 0] and [1 0 1]) and **Coset4** ([1 0 0] and [0 1 1]). The encoder for X identifies the set containing the codeword for X and sends the index for the set instead of the individual codeword. Since there are 4 sets, they can be indexed in 2 bits. The decoder, in turn, on the reception of the coset index, uses Y to disambiguate the correct X from the set by declaring the codeword that is closest to to Y as the answer. Note that the distance between X and Y is at most 1, and the distance between the 2 codewords in any set is 3. Hence, decoding can be done perfectly. (e.g., if Y is [0 0 1] and X is [0 1 1], then encoder sends the index for **Coset 4**. The decoder on receiving this index, calculates the distance between ([0 0 1] and [1 0 0]) which equals 2, and between ([0 0 1] and [0 1 1]) which equals 1. Since it is known that the distance between X and Y is at most 1, [0 1 1] is decoded as the observed codeword). This mode of encoding where the decoder has access to correlated side information is known as **side information coding**. It was shown in [5, 6] that in theory, for certain situations, the performance of a side information coding system can match that of one based on predictive coding. In a nutshell, the correlation between X and Y can help reduce the transmission rate. We now make the following observations from this example which hold in general, and which will be useful in the sequel.

- We note that **Coset1** is a repetition channel code [23] of distance 3 and the other sets are cosets [24, 25] of this code in the codeword space of X. We have used a channel code that is "matched" to the correlation distance (equivalently, noise) between X and Y to partition the source codeword space of X. This results in a side information encoding system that gives a **high compression**

performance identical to a predictive coding system.

- In practice, the partitioning of the source codeword space and index labeling of the resulting cosets (index labels for cosets are also called syndromes) can be done in a very *computationally efficient* way through the framework of coset codes [24, 25]. Thus, the encoder in a side information coding system incurs a **low encoding complexity**.

- Note that this partitioning of X is also *universal*. That is, the same partitioning of X works for all Y regardless of the value of Y as long as both X and Y satisfy the correlation structure. (e.g., if X is [0 1 0], then the same encoding for X (index of **Coset 3**) will be applicable to all cases of Y i.e., [0 1 0], [1 1 0], [0 0 0] and [0 1 1]. Thus if Y takes value [0 1 0] and during transmission to the decoder it gets "corrupted" to [1 1 0] so that the decoder has a corrupted version of Y, X can still be recovered correctly. This is because the corrupted version of Y also satisfies the correlation structure.) Thus, unlike a predictive coding setup there is no dependency between the encoding for X and the value of the correlated information Y thus providing **robustness**.

3.3 From the Illustrative Example to PRISM Video: Key Intuition

Motivated by the above example, we now revisit the video coding problem. Let **X** denote the current macro-block to be encoded (e.g., **X** is a vector of size 256 if macro-blocks of size 16×16 are chosen) . Let **Y** denote the best (motion-compensated) predictor for **X** in the previous frame (used for temporal prediction) and let $\mathbf{Y} = \mathbf{X} + \mathbf{N}$ (where we model **X**, and **N** as independent random vectors.). We first encode **X** in the intra-coding mode to come up with the quantized codeword for **X**. Now, using the insight from the above example, we find a channel code that is matched to the "correlation noise" **N**, and use that to partition the quantized codeword space of **X**. We can thus expect to approach the compression performance of predictive coding incurring only the complexity of intra-coding at the encoder. This is the main intuition behind the PRISM approach.

Note that in the above example we were dealing with relatively simple discrete sources exhibiting simple and precisely known correlation structures. In extending this to the video scenario, however, we recognize that we are dealing with real-valued sources having correlation noise structures with respect to the appropriate side-information units (from previous decoded frames) that are highly spatially-varying, are potentially unbounded in magnitude, and are imprecisely known, requiring estimation models. Thus while perfect decoding was possible in the above example (zero decoding error probability), there is, in general, a non-zero probability of decoding error in our case. Indeed, while the toy example does illustrate the key intuition behind our proposed video codec design, there is considerable "video engineering" needed to bridge the gap between a toy example and a working video codec. This is a daunting task, and we

12

now describe our initial attempts at bridging this gap.

4 PRISM Codec: A Modular Description and Some Practical Issues

We now give a modular description of a specific implementation of PRISM. As indicated earlier, at a high level this implementation involves roughly a swap in encoding and decoding complexities compared to traditional predictive coding based codecs, with the motion search operation being done at the decoder rather than at the encoder as in conventional codecs. We keep our design as close as possible to the MPEG and H.26x framework of block DCT and block motion compensation in order to better leverage existing codec modules and also for ease of transcoding in a network proxy application (see Section 2.4).

Given our task of building a new video codec based on distributed coding principles, we have been driven primarily by simplicity and proof-of-concept motivations. In this light, we choose syndrome codes based on relatively simple trellis codes [7], which work well even at reasonably small block-lengths (unlike more sophisticated channel codes such as LDPC codes [26], turbo codes [13] etc.): this has the added advantage of lowering the latency for real-time applications. As our design toolkit for trellis-based syndrome coding is currently confined to integer bit rate codes, our preliminary codec design is built around this constraint. This leads to specific choices in our preliminary design such as the use of a two-step quantization approach where the base quantizer attains a quality that is commensurate with the available side information and 1 bit/sample syndrome rate and the refinement quantizer subsequently improves the decoded quality to the target level. This will be more apparent in the following section.

The encoding and decoding operations in the PRISM codec are fundamentally driven by information-theoretic prescriptions from lossy source coding with side information or Wyner-Ziv coding [6]. We accordingly organize the constructive operations into functional modules that have related theoretical counterparts. We refer the reader to the excellent information-theory textbook by Cover and Thomas [27]) for an exposition of this theory, but summarize it here in a nutshell.

4.1 Brief summary of information-theoretic prescription

At a functional level, Wyner-Ziv compression of a source X in the presence of a correlated source Y at the decoder proceeds according to a two-step design process: (i) a rate-distortion source codebook is designed for quantizing X into its quantized description U; (ii) this source codebook is then "randomly" partitioned into bins or cosets (akin to the illustrative example of Section 3), where the number of cosets depends on the correlation structure between X and Y and the targeted encoding bit rate.

Encoding consists of (a) first quantizing X to U using the source rate-distortion codebook, and (b) then syndrome-encoding U, i.e. specifying the index of the coset into which U falls. Note that this is the key step to attaining a rate rebate by exploiting the correlation between X and Y. Decoding proceeds

13

by (a) first decoding U from its associated coset by using Y to find the "closest" entry in the list of codewords contained in the coset representation for X; and (b) then finding the best (MMSE) estimate for X using both U and Y, which are akin to "noisy observations" of X.

Further, for pure compression of Gaussian *vector* sources having independent but non-identically distributed components, there is an "inverse-waterfilling" prescription for optimally allocating bit rate among the source vector components [27] (each component will be at the same distortion-level at high enough rates). This inverse-waterfilling algorithm has been generalized to the case of side-information compression of arbitrarily correlated Gaussian vector sources in [28]. The prescription dictates that the optimal decorrelating transform for encoding the source in such a case corresponds to the Karhunen-Loeve (KL) transform [29] *of the innovations process* – even though the innovations signal is not present at the encoder – regardless of the spectrum of the source being encoded. The transform coefficients of the source then need to be optimally quantized using the inverse-waterfilling prescription applied to the *innovations noise vector components*. Note that this is a departure from the classical quantization of Gaussian vector sources, where the inverse waterfilling is in accordance with the input source spectrum.

4.2 From information theory to video codec practice

There are several complications in translating this theoretical prescription into constructive modules for a real video codec, which we summarize here. These will followed up in detail in Sections 4.3 and 4.4.

- First, while the theory assumes knowledge of the correlation structure between X and Y, in practice, this structure is not known precisely and needs to be estimated. In our approach, the classification module aims to classify the statistical nature of the correlated side-information into a prescribed set of classes (from zero to maximum correlation). Each video block will be accordingly classified into a prescribed discrete set of correlation noise classes, which in turn will influence the syndrome trellis channel code choice.

- Secondly, while information-theory dictates for the optimal decorrelating transform in the side-information coding case to be the KL transform of the innovations noise vector process, in our practical version, we will invoke the ubiquitous DCT.

- Thirdly, the rate-distortion source codebook prescribed in theory will be approximated with a scalar quantizer. The "random" partitioning operation dictated by information-theory will be approximated by trellis-based coset codes. Thus, while the theory calls for a random codebook to be partitioned into random cosets, our construction will be based on a scalar quantizer lattice partitioned into trellis-coded cosets.

14

- The maximum-likelihood syndrome decoding will be done through a Viterbi decoder. Further, unlike the classical Wyner-Ziv coding case, where there is a single known side-information Y, in the video case, there will be several side-information candidates Y_i corresponding to various motion-predictor choices that will be tried sequentially until we find a Y_i that is "matched" to the trellis code choice (akin to motion search).

- Finally, the syndrome-decoded output and the side-information source will be optimally "fused" linearly to form an MMSE estimate of the source reconstruction.

We now enlist the functional modules for both encoding and decoding based on the above "mappings" to the theoretical prescriptions.

4.3   Encoding

The video frame to be encoded is divided into non-overlapping spatial blocks (we choose blocks of size 16×16 or 8×8 in our implementations.). We now enlist the four main aspects of the encoding process.

1. **Classification:** As mentioned in Section 3.3, real video sources exhibit spatio-temporal correlation noise structures whose statistics are highly spatially varying. Within the same sequence, some spatial blocks that are a part of the scene background do not change much with time. That is, they are highly correlated with their temporal predictors (small **N**). On the other hand, there are some blocks that are a part of a scene change or occlusion. Such blocks have little correlation with the previous frame (large **N**). Thus within the same frame, different blocks exhibit different degrees of correlation with the previous frame. This motivates the modeling of the video source as a composite or a mixture source where the different components of the mixture correspond to sources with different correlation (innovation) noise.

   The classification step therefore aims at classifying these correlation noise structure at a fine-grained block level into a prescribed set of pre-designed classes from zero to maximum correlation. These classes correspond to syndrome channel code choices of varying error-correcting capabilities. That is, once a block is classified, it will be tagged with the appropriate channel code for side information coding of that block.

   In our current implementation, we use the squared error difference between the block to be encoded and the co-located block in the previous frame to model the correlation noise **N**. This squared error difference is thresholded and the block is classified into one of many classes enabling the use of the appropriately matched channel code. At one extreme is the SKIP mode, where the frame difference is so small that the block is not encoded at all, and at the other extreme is the INTRA

mode, where the frame difference is very large suggesting poor correlation, so that intra-coding is appropriate. There are various different syndrome coding modes in between these two extremes. After classification, the classification label is included as part of the header information for use by the decoder.

2. **Decorrelating transform:**

   As outlined in Section 4.2, we apply a DCT on the source vector to approximate the KL transform of the correlation noise innovations process between the source vector and its side-information counterpart. This is akin to the classical operation of a DCT on the motion-compensated Displaced Frame Difference (DFD) process in standard coders.

3. **Quantization:** This step involves scalar quantization of the DCT coefficients of the source vector (akin to the source rate-distortion codebook in Wyner-Ziv theory [6]). As outlined in Section 4.1, the information-theoretic prescription of bit allocation for independent correlated vectors [28] has to drive the design choice of the quantizer resolution for each DCT coefficient according to to the inverse-waterfilling prescription applied to the innovations (DFD) coefficients.

4. **Syndrome Encoding:** This step represents the constructive counterpart to the random coset partitioning operation, and involves the syndrome encoding of the quantized block codewords. This step partitions the codeword space of quantized codewords into cosets, each containing a collection (or uncertainty list) of codewords. A syndrome label is associated with each such coset. Compression is obtained here since instead of transmitting each individual codeword index, only the index of the set containing the codeword index is transmitted. If syndrome encoding is done using a code matched to the correlation noise (as was pointed out in the illustrative example in Section 3.2) we can then expect to achieve performance comparable with predictive coding.

   Our implementation of the above two steps proceeds as follows:

   (a) **Base Scalar Quantization:** The transform coefficients (which are real numbers when the DCT is used), are first quantized before encoding. When the total number of coset partitions is fixed (as is the case when trellis codes [7] with a syndrome rate of 1 bit/sample are used), the choice of the quantization step size is limited by the statistics of N. If a very fine step size is chosen to encode X, then there can be decoding errors, since the codewords will be too "close" so that the side information Y cannot disambiguate them correctly. This is illustrated through the example in Figure 4. Here the top line shows the quantized codeword set for X, and the two bottom lines show the partition of the space of quantized codewords (two partitions imply a rate of 1 bit). The rectangular box shows the observed codeword which lies

16

in the first partition. Since the magnitude of **N** is more than the quantization step size, the decoder uses the side information **Y** to decode the incorrect (circled) codeword. Thus, each of the elements of **X** is quantized with a step size proportional to the standard deviation of the corresponding element in **N**.
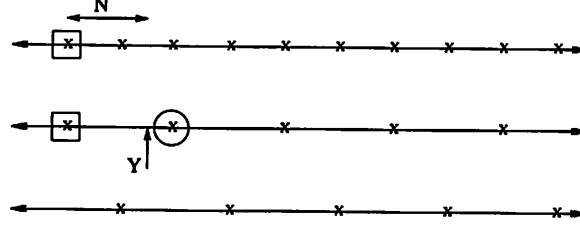

Figure 4: The top line shows the quantized codewords for X. The bottom two lines show the two partitions of the quantized codeword space of X. The box shows the observed codeword. The observed codeword lies in the first partition. The magnitude of N is more than the quantizer step size. Hence, the decoder decodes the circled codeword and makes a decoding error.

(b) **Zig-Zag Scan:** The quantized coefficients are arranged in a 1-dimensional order (size 256 or 64) by a doing a zig-zag scan [4] on the 2-dimensional block (size 16×16 or 8×8).

(c) **Syndrome Encoding:** Now the space of quantized codewords which has been appropriately generated using the statistics of **N** is partitioned using a Euclidean space trellis channel code [25, 30]. In our particular implementation, we use a memory-7 rate-1/2 trellis code from [24]. A rate-1/2 trellis code of block length N is a subspace of $\{0, 1, 2, 3\}^N$ (The repetition channel code of block length 3 ([0 0 0] and [1 1 1]) is a subspace of $\{0, 1\}^3$.). Hence, it can be used to partition the space $\{0, 1, 2, 3\}^N$. For this reason, we need to "convert" the space of quantized codewords to $\{0, 1, 2, 3\}^N$. This can be done by using a mod-4 labeling of the quantization lattice. An illustration of this is shown in Figure 5.
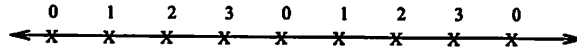

Figure 5: Mod-4 labeling of the space of quantized codewords of source X.

The transmission or the coset index rate [5] incurred in this case is 1 bit/sample. The generation of the coset index (syndrome) associated with each codeword can be accomplished in a computationally efficient manner through a simple convolution operation (linear in the number of coefficients) between the quantized codeword and the parity check matrix [23] of the trellis

---

[4]It has been observed in general that arranging 2-dimensional coefficients in a 1-dimensional order using a zig-zag scan pattern tends to organize them in decreasing order of energies (importance).

[5]A rate-1/2 trellis code of block length N which is a subspace of $\{0, 1, 2, 3\}^N$ has 2^N codewords in the space of size 4^N. Hence there are $\frac{4^N}{2^N} = 2^N$ cosets associated with it, which can be indexed by N bits, corresponding to a rate of 1 bit/sample.

code.

Further, in each block of each class, only the first fraction of the scanned coefficients are syndrome encoded. The remaining coefficients are purely intra-coded. This is based on the observation that for typical natural images, the first few transform coefficients contain most of the information about the block. We thus expect them to exhibit significant correlation with the corresponding predictor blocks. In our implementation, both with 8×8 blocks and 16×16 blocks typically only about 20% of the coefficients are syndrome encoded.

(d) **"Pure" Source Coding:** The remaining coefficients which comprise about 80% of the total coefficients are intra-coded in the conventional way. The coefficients are first quantized, then zig-zag scanned and finally are entropy coded using run-length Huffman coding.

(e) **Refinement Quantization:** A target reconstruction quality [6] corresponds to a particular quantization step size. (Higher desired quality corresponds to a finer quantization step size and lower corresponds to a coarser quantization step size). The coefficients that are purely intra-coded are quantized with a step size corresponding to the target quality. But, for the coefficients that are syndrome encoded, the choice of the base quantization step size is limited by N. This is done so as to minimize the probability of decoding error of the trellis codes. Hence, assuming that the base quantization interval can be conveyed correctly with high fidelity to the decoder, we refine it further to the target quantization step size. In our current implementation, the refinement operation is just a progressive sub-dividing of the base quantization interval, into intervals of size equal to the target quantization step size. The index of the refinement interval inside the base interval is transmitted to the decoder.

Note that our operation of base quantization followed by refinement quantization is a practical illustration of the inverse-waterfilling concept described earlier[7] This is because coefficients with significant correlation (small N) have a fine base quantization step size and hence the refinement stage results in fewer refinement intervals and hence fewer refinement bits. Thus the bit allocation is proportional to the correlation - fewer bits are required if correlation is high and more bits when the correlation is weak.

Figure 6 summarizes the encoding approach adopted for encoding coefficients in a particular block.

(f) **Cyclic Redundancy Check (CRC):** We note that at the encoder, side information encoding is done in principle with respect to the statistics of the motion compensated prediction error

---

[6]Quality is typically measured in PSNR (Peak Signal-to-Noise Ratio) (dB). PSNR $= \log_{10} \frac{255^2}{MSE}$ where MSE denotes squared error between the original block and the encoded block divided by the number of pixels in the block.

[7]The two-stage quantization process is necessitated by our operational constraint of a limited 1 b/sample trellis syndrome code.
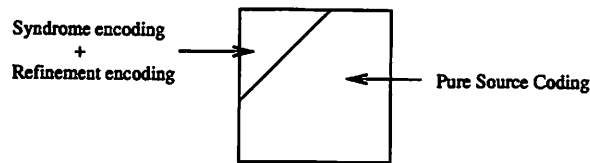
Figure 6: After zig-zag scanning, the first few coefficients are encoded by syndrome encoding followed by refinement encoding. The remaining fraction of coefficients are purely source coded (intra-coded).

between the block X that is to be encoded and the "best" predictor Y for this block in the frame memory. At the decoder, all that is available is the frame memory. The decoder does not know the "best" predictor for the block X. The encoder transmits not only the syndrome for the side information encoded coefficients but also a CRC check (of sufficient strength) of the quantized sequence of mod-4 codewords. This CRC check serves as a "signature" of the quantized codeword sequence. In contrast to the conventional paradigm, it is the decoder's task to do motion search here, and it searches over the space of candidate predictors one-by-one to decode a sequence from the set labeled by the syndrome. When the decoded sequence matches the CRC check, decoding is declared to be successful. Note that the CRC needs to be sufficiently strong so as to act as a reliable signature for the codeword sequence.
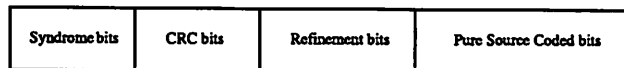

Figure 7: Bit-stream syntax associated with a block. Compare the lightness of this syntax with that of a typical H.26x encoder format.

The bit-stream associated with a block is illustrated in Figure 7. To summarize, the main complexity in the encoding process is incurred in steps 2 (complexity of the DCT) and in the entropy coding stage. Hence, the encoding complexity in the PRISM algorithm is of the order of standard intra-coding complexity. A block diagram of the encoding approach is illustrated in Figure 8.

4.4 Decoding

The PRISM decoder (see Figure 9), on the other hand, incurs a relatively high decoding complexity. The main modules are:

1. **Generation of Side Information (Motion Search):** The decoder does motion search to generate candidate predictors to decode the sequence of quantized codewords from the set indicated by the received syndrome. In our current implementation, half pixel motion search is used to obtain
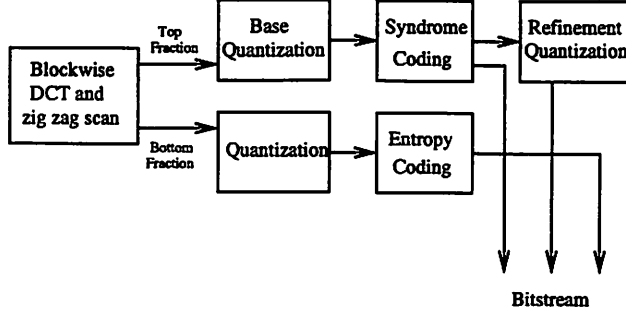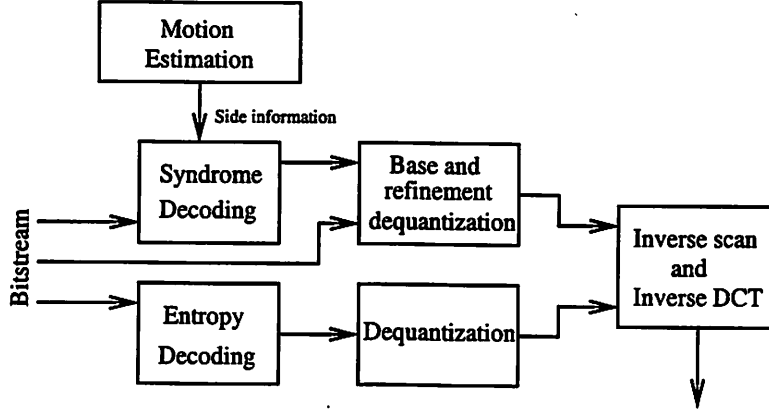
Figure 8: Functional block diagram of the encoder.

Figure 9: Functional block diagram of the decoder.

various candidate predictors as is also done at the encoding side in the standard video algorithms [2, 3, 4]. We reiterate that the framework is very general so as to accommodate any other sophisticated motion estimation procedures such as multi-frame prediction [12], variable block sized motion estimation [4], optical flow [10] etc. The choice of a more sophisticated algorithm can only serve to enhance the performance of the PRISM paradigm.

2. **Syndrome Decoding:** Each of the candidate predictors generated by the motion search module is used to decode a sequence of quantized codewords from the set indicated by the syndrome. Since we use trellis codes (in our implementation we chose a 128-state rate-1/2 trellis code from [24]), this decoding can be accomplished using the Viterbi algorithm [31]. Here the set of all sequences labeled by the received syndrome is represented on a trellis. The Viterbi algorithm is then used to identify the sequence in this set that is "nearest" to the candidate predictor. If this decoded sequence matches the CRC check, then the decoding is declared to be successful. Else using the motion search module, the next candidate predictor is obtained and then the whole procedure repeated.

3. **Estimation and Reconstruction:** Once the quantized codeword sequence is recovered, it is used along with the predictor to obtain the best reconstruction of the source. In our current implementation, we use the best linear estimate from the predictor and the quantized codeword to obtain the source reconstruction. However, any of the sophisticated signal processing algorithms (e.g., spatio-temporal interpolation) or post processing mechanisms can be deployed in this framework and these can only serve to improve the overall performance.

Besides the three main steps listed above, we need to go through the following steps so as to completely decode the PRISM bit-stream.

4. **"Pure" Source Decoding:** For the coefficients (about 80%) that have been intra-coded, the decoding action action consists of entropy decoding followed by dequantization.

5. **Inverse Zig-Zag Scan:** Once all the transform coefficients have been dequantized, the zig-zag scan operation carried out at the encoder is inverted to obtain a 2-D block of reconstructed coefficients.

6. **Inverse Transform:** The transformed coefficients are then inverted using the inverse transform so as to give reconstructed pixels.

To summarize, the PRISM framework is very general and can seamlessly incorporate both rich motion modeling procedures as well as sophisticated signal processing and error concealment mechanisms.

5 PRISM: Features

In light of the implementation details presented above, we now revisit the salient features of the PRISM framework. We evaluate the implications of these features from an architectural and a networking point of view.

1. **Low Encoding Complexity:** As pointed out in Section 4, the encoding complexity of the PRISM paradigm is nearly that of intra-coding (I). The need for motion search which can typically cost over 65,000 operations/pixel/second is completely obviated in the PRISM encoder. Further, since motion estimation is not performed at the encoder, frequent memory accesses to load the frame memory which are power and delay intensive, are avoided. This makes it especially suitable for wireless scenarios where the encoding devices are battery power constrained.

2. **High Decoding Complexity:** Operations like motion search that are performed by the encoder in the conventional paradigm, are performed by the decoder in the PRISM framework. Besides, the decoder performs an extra operation of Viterbi decoding per candidate predictor. In the scenario of

Figure 2, the PRISM decoding operation is accomplished by the "network proxy" which is endowed with lots of processing power.

3. **High Compression:** The principal goal of the PRISM approach is to approach the compression performance of inter-coding while incurring the encoding complexity of intra-coding. As explained in Section 4.3 compression is obtained here in the syndrome encoding stage. Details of the performance of the current implementation are presented in Section 6. Efficient compression reduces the size of the bit-stream and thus minimizes the total transmitted power prolonging the cell/sensor unit battery life.

4. **Robustness:** Since PRISM is based on the usage of channel codes for the correlation noise between temporally adjacent video frames, it also possesses the feature of robustness inherently. In the face of channel noise which can lead to packet drops, the same channel codes are also used to overcome the channel noise in addition to the correlation noise. For example, suppose the frame memory does not have the previous frame due to transmission loss but only the frame prior to it. Then as long as that frame is correlated enough so that it is "matched" to the channel code used for encoding the current block, it would still be usable for decoding. In Section 6 we present some results which highlight the robustness feature of PRISM.

In fact, the PRISM framework allows for a continuously tunable trade-off between compression efficiency and robustness. The base quantization step size is a key tunable parameter here. If this step size is chosen more coarsely than is required by the existing correlation, then this means that syndrome decoding would be successful even for weaker correlations. Thus, if some parts of the previous frame have been lost, then the corresponding parts from the frames before the previous frame, even though they are weakly correlated as compared to the current frame, can be used for decoding. A coarser quantization step size however, means a greater refinement layer rate for attaining the same desired quality. This tunability of base quantization step size thus offers a robustness-bitrate trade-off.

5. **Probability of Decoding Error:** As alluded to in Section 3.3, probability of decoding error is an artifact of the side information coding paradigm and is one of the drawbacks of PRISM. It leads to erroneous decoding of blocks, the effects of which can propagate resulting in displeasing visual artifacts. As mentioned above, this can be dealt with by the choice of the base quantization step-size. A conservative choice for the same reduces the probability of decoding error at the cost of compression efficiency and vice versa. This choice needs to be tightly coupled with the classification step in 4.3. Further, the deployment of sophisticated error concealment algorithms can minimize

these effects. Also, the availability of a feedback channel between the PRISM decoder and encoder can be used by the decoder to inform the encoder as to which blocks have been decoded in error.

6. **Efficient Trans-coding:** The transcoding proxy that resides in the base station can be implemented efficiently in the PRISM framework. Instead of first completely decoding the PRISM bit-stream and then re-encoding it afresh using the conventional video encoder, an efficient implementation would consist of using the predictors that have been obtained by motion search at the PRISM decoder for the conventional video encoder. Thus, the duplication in motion search can be avoided. In short, efficient transcoding algorithms are possible within the PRISM paradigm.

7. **A Flexible Encoding Syntax:** As opposed to the current video coding standards where the bit-stream contains fields like motion-vector fields, block-size used for motion compensation, reference frame used etc, the PRISM bit-stream is associated with a light syntax (see Figure 7). The advantages of this have been described in Section 2.3.

6 Preliminary Simulation Results

In this section, we present some preliminary simulation results that illustrate the various features of PRISM. Our current implementation operates well in the high quality (PSNR of the order of 30 dB) regime. The extension to lower bit rates is a bit more involved, and is part of our ongoing work.

We present compression results obtained for the first 15 frames of the Mother and Daughter (352x288), Carphone (176x144) and the Football (352x240) video during our experiments. Out of these sequences, Mother and Daughter has relatively less motion content and is easier to compress, the Carphone sequence has medium motion content and the Football sequence has high motion content. These sequences were chosen so as to test the validity of the PRISM paradigm. The reference system is an implementation of the H.263+ [2] video coder obtained from University of British Columbia, Vancouver.

For both PRISM and H.263+, the first frame is encoded in the intra mode (i.e., every block in the first frame is encoded in the intra-coding mode). The remaining frames are encoded in the non-intra mode. Figure 10 summarizes the rate-PSNR [8] performance of PRISM in comparison with the H.263+ coder for the three chosen sequences. From a pure objective compression point of view, we note that the performance of the current implementation of PRISM lies between the inter and intra coding modes of H.263+. Surprisingly, we discovered that the subjective performance of PRISM is virtually indistinguishable visually from that of the H.263+ coder for our simulations.

---

[8]As was mentioned in Section 3.3 in general there is a probability of decoding error associated with the coding with side information paradigm. In the above simulations, we targeted a block error rate of about 0.5%. This results in isolated block errors that virtually go unnoticed by the subjective eye with simple concealment approaches. The PSNR calculations for PRISM plotted in Figure 10 take into account the PSNR associated with the blocks that are decoded correctly.

We also conducted preliminary tests on the robustness of the proposed PRISM framework. For both PRISM and the reference system, we introduced a frame loss by removing the second frame in the video sequence from the frame memory. This while the third frame is encoded off the second frame at the encoder it is decoded off the first frame in the H.263+ case. This leads to drift which accumulates and propagates in this case. In contrast, the decoded quality is minimally affected in PRISM and drift does not occur. Figure 11 shows the decoded visual quality for the Football sequence in both cases. Figures 11 (a), (c) and (e) show respectively the decoded first, third and the fourteenth frames for the PRISM paradigm. Figures 11 (b), (d) and (f) show respectively the decoded first, third and the fourteenth frames for the H.263+ coder. We point out that in the decoding of the third frame, the first frame is used as side-information at the decoder. This leads to a moderate drop in quality (of the order of 0.2 dB) with respect to the case where the second frame is used as side information at the decoder. However in the case of H.263+ the drop in quality is very significant leading to displeasing visual artifacts (see Figure 11 (d)) which accumulate and propagate through the remainder of the sequence (see Figure 11 (f)). In particular, the jersey number of the football player with jersey 57 is not even visible in Figure 11 (f) while it is fairly clear in Figure 11 (e). These experiments thus highlight the inherent robustness of PRISM.

7 Conclusion and Future Directions

We have introduced PRISM, a video coding framework built on distributed source coding principles. Quintessentially, we have demonstrated the novel feasibility of building a competitive video codec predominantly on *channel coding* ideas. There are three main architectural goals of PRISM that make it radically different from existing video codecs: (i) flexible distribution of complexity, without compromising the compression performance, between encoder and decoder (including the special case of reversal of complexities with the decoder picking up the expensive task of motion search, as has been detailed in this work); (ii) naturally inbuilt robustness to drift between encoder and decoder caused by lack of synchronization due to channel loss (as has been demonstrated successfully through simulations in this work); and (iii) a very light yet powerful syntax that is well-suited for standardization and seamless innovation within the standard.

Note that the PRISM concept applies even if only a subset of these goals is desired. For example, if complexity is not an issue but robustness is, the PRISM approach can be used to robustify standard fragile prediction-based video streams as a superior low-latency replacement for FEC-based error-resiliency solutions. At the other extreme, robustness may be less important than low-complexity encoding, for example in high-end video camera applications, where full-motion MPEG represents a processing bottleneck due to a high frame-rate acquisition front-end. However, the full power of PRISM is expressed when both low-complexity and robustness are needed, for example in video-over-wireless applications

like surveillance cameras or video telephony, where device memory and/or battery life are important, and the transmission environment causes packet/frame drops. We have also argued how the lightness of the PRISM encoding syntax facilitates easy standardization and unencumbered innovating within the standard. Further, we have described how the PRISM concept can be packaged into a transcoding solution having a "half-duplex" format of uplink-PRISM and downlink-MPEG in order to allow for low-complexity end points for both transmit and receive, with all the complexity being "sucked" into a middle-man network device.

In describing a specific implementation of PRISM based on a block-DCT, block motion-compensation based framework, we have been driven by the motivation of staying "close" in architecture to existing popular commercial standards like MPEG-4 and H.26x. We have chosen to bias our initial implementation in the direction of very simple components, including simple scalar quantizers and integer-rate trellis-based syndrome codes. We have taken care, however, to ensure that all phases of our codec design chain are endorsed by fundamentally sound theoretical principles (as described in Sections 4.1 and 4.2). Despite the simplicity of the system components, our initial simulations on typical video test sequences of diverse scene and motion difficulties, validate the power of PRISM. This validation is both from a subjective pure compression performance standpoint with respect to existing state-of-the-art H.263+ codec implementations, and especially from a significantly superior robustness standpoint with respect to the existing class of fragile prediction-based codecs in the face of channel packet/frame drops.

While this fuels our optimism about the promise of PRISM for uplink-rich media applications, much work remains before a complete codec system can be endorsed. One needs only to witness the massive amount of effort that has gone into current commercial standards for video compression over the past two decades in order to appreciate the value of video engineering in innovating and improving upon a baseline system involving block-motion compensation based predictive coding. We envisage a similar scale of effort being needed to make the concepts of PRISM a reality for practical and ubiquitous deployment. There are numerous directions for future research that we are eager to follow up on.

- More extensive simulations at a variety of compression rates and picture sizes are in order. We have been somewhat constrained by the use of simple trellis codes for 1 bit/sample syndrome encoding in the PRISM implementation described in this work. This needs to be extended along two dimensions. First, more flexibility in syndrome channel coding rates is needed – we propose to study the rich framework of multi-level modulation codes [32] to this end. It is also worth noting in this regard that we incur a compression performance hit in our current implementation due to our use of a refinement quantizer stage. This is unavoidable given our limitation of needing a coarse quantizer first-stage that uses a 1 bit/sample syndrome code, followed by a refinement quantizer second stage. With more flexibility in code rates, we can directly target the finer quantizer step size and partition

25

it into cosets, avoiding the performance-hampering successive refinement quantization operation. Secondly, more sophisticated channel codes based on turbo codes [13, 17] and LDPC codes [26, 33] need to be integrated into the PRISM fold. We hope to heavily leverage recent promising work on turbo-trellis-code based syndrome codes in [17], where the gap between achievable practice and the Wyner-Ziv theoretical bounds is less than 1 dB at low bit rates.

- The classification phase of the codec in estimating the temporal correlation (innovations) noise variance is critical to the performance of PRISM. Basically, the success of PRISM rests on the accuracy of this estimation. While a full motion search at the encoder will allow for better estimation of the innovations noise, this is expensive computationally (and motion vectors are fragile to channel loss). We will accordingly direct our future study towards more robust and sophisticated approaches to classification, keying on the complexity versus performance trade-off benefits. The work of [34] on fast motion search appears to be a promising hunting ground for this, albeit in the "reversed" encoder-decoder paradigm. More fundamentally, more mature spatio-temporal modeling of motion and video will be fundamental to advances in the PRISM framework just as in classical prediction-based frameworks.

- Exploiting the fact that the PRISM syntax is transparent (modulo complexity) to the exact choice of motion compensation used, we will explore the benefits of more powerful motion compensation strategies than used in commercial standards today, including the use of multi-frame prediction [12], the use of matching-pursuit-based time-frequency atoms [11], etc. The recent advances on motion-compensated prediction endorsed by the low bit-rate video coding standards bodies (improved loop-filters, hybrid motion block sizes, etc.) are all fair game for incorporatation into the PRISM framework, albeit in a "reversed" way (i.e. at the decoder rather than at the encoder).

- We have exclusively focused here on the "reversed" complexity paradigm where encoder and decoder complexities are roughly swapped with respect to conventional schemes. However, PRISM allows for a much more flexible distribution of complexity. Specifically, the dominant motion-search complexity can be arbitrarily shared between encoder and decoder under the PRISM paradigm, and the full range of the benefits of this need to be explored. We propose to leverage the theoretical underpinnings studied in [8] to guide this effort.

- Finally, the entire deployment of PRISM inside a network in a transcoding proxy regime (Figure 2) is an open problem with issues relating to the role of network feedback in improving the performance, low-complexity transcoding strategies, etc. need to be studied from scratch. Of particular importance is the exploration of latency versus performance trade-offs for real-time streaming applications of the PRISM transcoder.

To conclude, this paper represents but a scratching of the surface of what we believe is an exciting new direction for video coding for a large class of emerging uplink-rich media applications. We are optimistic that this paper represents a promising start to this exciting journey.

References

[1] R. Puri and K. Ramchandran, "PRISM: A New Robust Video Coding Architecture Based on Distributed Compression Principles," *40th Allerton Conference on Communication, Control and Computing*, October, Allerton, IL, 2002.

[2] G. Cote, B. Erol, M. Gallant, and F. Kossentini, "H.263+: Video Coding at Low Bit Rates," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 849–66, November 1998.

[3] B. G. Haskell, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2.* Kluwer Academic Publishers, 1996.

[4] T. Wiegand and G. S. Sullivan, "The Emerging JVT/H.26L Standard," *Tutorial in International Conference on Image Processing (ICIP)*, September, Rochester, NY, 2002.

[5] D. Slepian and J. K. Wolf, "Noiseless Coding of Correlated Information Sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, July 1973.

[6] A. Wyner and J. Ziv, "The Rate-Distortion Function for Source Coding with Side Information at the Decoder," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, January 1976.

[7] S. S. Pradhan and K. Ramchandran, "Distributed Source Coding Using Syndromes (DISCUS): Design and Construction," *Proceedings of the Data Compression Conference (DCC)*, March, Snowbird, UT, 1999.

[8] P. Ishwar, V. M. Prabhakaran, and K. Ramchandran, "Towards a Theory for Video Coding Using Distributed Compression Principles," *International Conference on Image Processing (ICIP)*, submitted, 2003.

[9] S. S. Pradhan, R. Puri, and K. Ramchandran, "(n,k) Source Channel Erasure Codes: Can Parity Bits Also Refine Quality?," *Proceedings Conference on Information Sciences and Systems (CISS)*, March, Baltimore, MD, 2001.

[10] R. Krishnamurthy, J. M. Woods, and P. Moulin, "Frame Interpolation and Bidirectional Prediction of Video Using Compactly-encoded Optical Flow Fields and Label Fields,," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, pp. 713–726, August 1999.

[11] O. K. Al-Shaykh, E. Miloslavsky, T. Nomura, R. Neff, and A. Zakhor, "Video Compression Using Matching Pursuits,," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, pp. 123–143, February 1999.

[12] B. Girod and T. Wiegand, *Multiframe Motion-Compensated Prediction for Video Transmission.* Kluwer Academic Publishers, 2001.

[13] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-correcting Coding and Decoding: Turbo-codes (1)," *IEEE International Conference on Communications*, vol. 2, pp. 1064–1070, May 1993.

[14] Y. Zhao and J. Garcia-Frias, "Data Compression of Correlated Non-Binary Sources Using Turbo Codes," *Proceedings of the Data Compression Conference (DCC)*, April, Snowbird, UT, 2002.

[15] A. Aaron and B. Girod, "Compression with Side Information Using Turbo Codes," *Proceedings of the Data Compression Conference (DCC)*, April, Snowbird, UT, 2002.

[16] A. Liveris, Z. Xiong, and C. Georgihades, "A Distributed Source Coding Technique for Highly Correlated Images Using Turbo Codes," *International Conference on Accoustics, Speech and Signal Processing (ICASSP)*, May, Orlando, FL, 2002.

[17] J. Chou, S. S. Pradhan, and K. Ramchandran, "Turbo and Trellis-Based Constructions for Source Coding with Side Information," *Proceedings of the Data Compression Conference (DCC)*, March, Snowbird, UT, 2003.

[18] S. S. Pradhan and K. Ramchandran, "Enhancing Analog Image Transmission Systems Using Digital Side Information: a New Wavelet Based Image Coding Paradigm," *Proceedings of the Data Compression Conference (DCC)*, March, Snowbird, UT, 2001.

[19] A. Majumdar, K. Ramchandran, and I. Kozintsev, "Distributed Compression of Correlated Audio Sources," *International Conference on Image Processing (ICIP)*, submitted, 2003.

[20] J. Chou, D. Petrovic, and K. Ramchandran, "A Distributed and Adaptive Signal Processing Approach to Reducing Energy Consumption in Sensor Networks," *Proceedings of the IEEE INFOCOM*, April, San Francisco, 2003.

[21] A. Aaron, R. Zhang, and B. Girod, "Wyner-Ziv Coding of Motion Video," *36th Asilomar Conference on Signals, Systems, and Computers,*, November, Pacific Grove, CA, 2002.

[22] W. Rabiner and A. P. Chandrakasan, "Network-Driven Motion Estimation for Wireless Video Terminals," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 644–653, August 1997.

[23] F. J. Macwilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*. Elseiver-North-Holland, 1977.

[24] G. D. Forney, "Coset Codes-Part I: Introduction and Geometrical Classification," *IEEE Transactions on Information Theory*, vol. 34, pp. 1123–1151, September 1988.

[25] G. D. Forney, "Coset Codes-Part II: Binary Lattices and Related Codes," *IEEE Transactions on Information Theory*, vol. 34, pp. 1152–1187, September 1988.

[26] R. G. Gallager, "Low Density Parity Check Codes," *Ph.D Thesis, MIT*, CAmbridge, MA, 1963.

[27] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, 1991.

[28] S. S. Pradhan, "On Rate-Distortion of Gaussian Sources with memory in the Presence of Side Iinformation at the Decoder," *Project Report, ECE 480, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign*, December 1998.

[29] A. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs: Prentice Hall, 1989.

[30] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 55–67, January 1982.

[31] G. D. Forney, "The Viterbi Algorithm," *IEEE Proceedings*, vol. 61, pp. 268–278, March 1973.

[32] U. Wachsmann, R. F. H. Fischer, and J. B. Huber, "Multilevel Codes: Theoretical Concepts and Practical Design Rules," *IEEE Transactions on Information Theory*, vol. 45, pp. 1361–1391, July 1999.

[33] S. Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the Design of Low-density Parity-check Codes Within 0.0045 dB of the Shannon Limit," *IEEE Communications Letters*, vol. 5, pp. 58–60, February 2001.

[34] K. Lengwehasatit and A. Ortega, "Probabilistic Partial Distance Fast Matching for Motion Estimation,," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp. 139–152, February 2001.
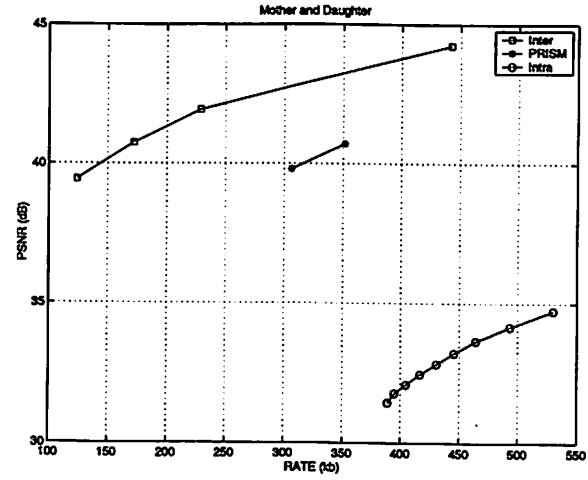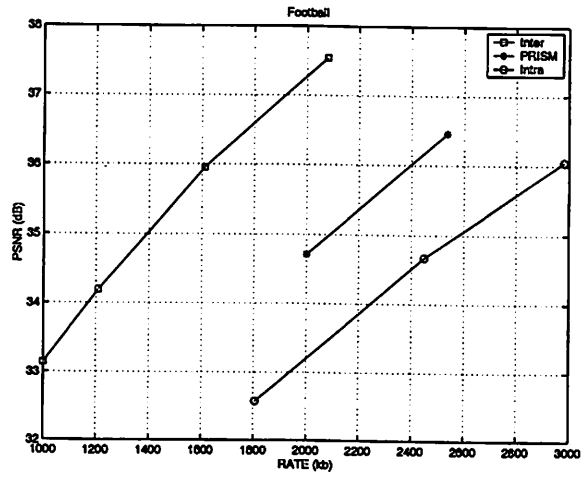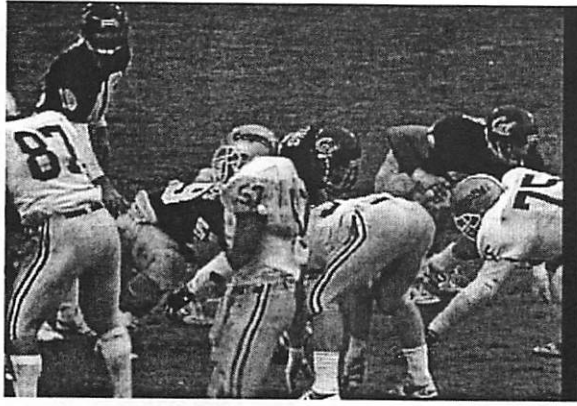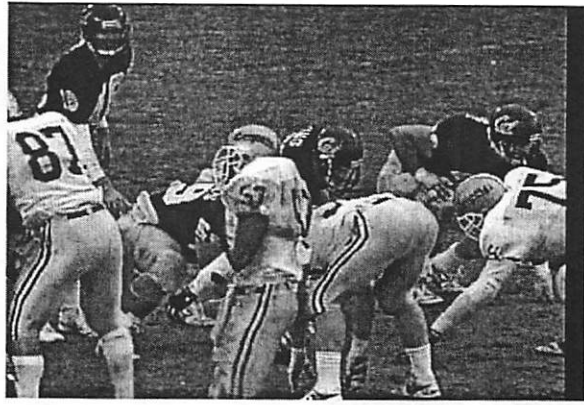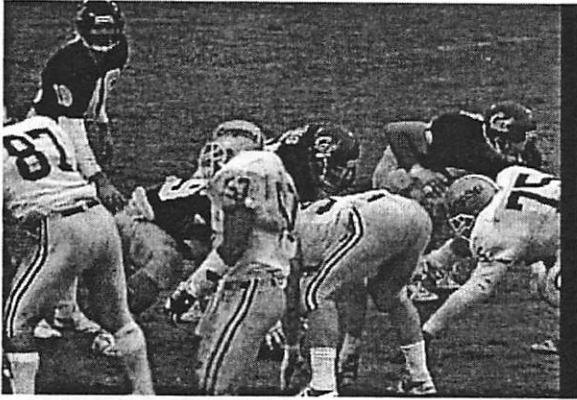
(a)
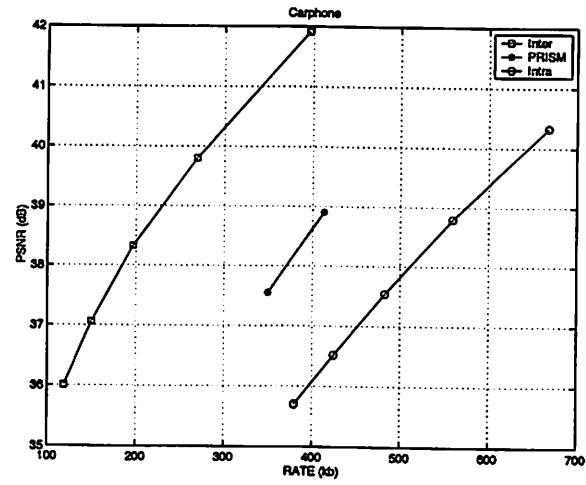
(b)

(c)

Figure 10: Compression Performance of PRISM and H.263+ coder in the case of frame loss. Fifteen frames of the Mother and Daughter, Carphone and Football video sequence were encoded in both cases.
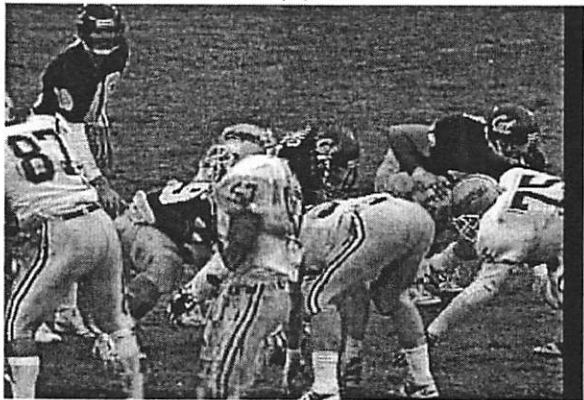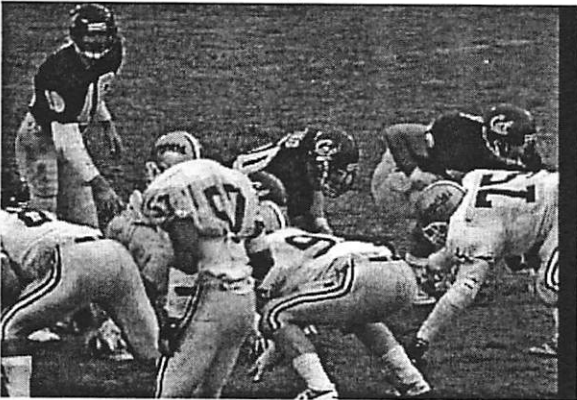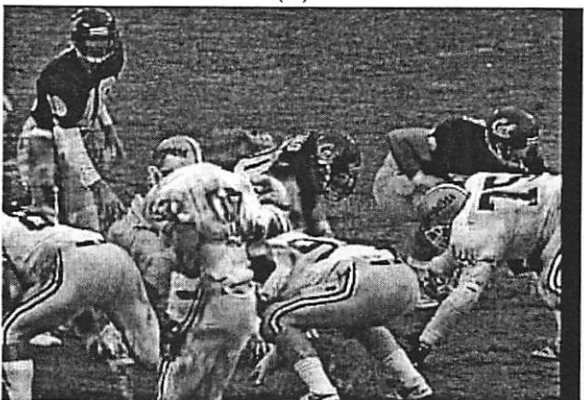
Figure 11: Performance of PRISM and H.263+ coder in the case of frame loss. Fifteen frames of the football video sequence were encoded in both cases and the second decoded frame was removed from the frame memory in both cases. The third frame was decoded using the first frame as side information for the proposed paradigm and a predictor for H.263+. Figures 11 (a), (c) and (e) show respectively the decoded first, third and the fourteenth frames for PRISM. Figures 11 (b), (d) and (f) show the same for the H.263+ coder. We see in Figure 11 (d) that displeasing visual artifacts arise because of the drift and Figure 11 (f) shows that they propagate for the remainder of the sequence. In particular the jersey number of the football player with jersey 57 cannot be seen in Figure 11 (f) while it is fairly clear in Figure 11 (e).

31