PRISM: A New Robust Video Coding Architecture Based on Distributed Compression Principles

Rohit Puri
EECS Dept., UC Berkeley
Berkeley, CA 94720
rpuri@eecs.berkeley.edu

Kannan Ramchandran
EECS Dept., UC Berkeley
Berkeley, CA 94720
kannanr@eecs.berkeley.edu

Abstract

We introduce PRISM (Power-efficient, Robust, hIgh-compression, Syndrome-based Multimedia coding), a new video coding paradigm based on the principles of distributed source coding that represents a near-antithesis to currently popular video compression architectures, such as typified by standards like MPEG and H.263. In direct contrast to conventional video coding architectures, PRISM's architectural goal is to *shift the computational burden from the encoder to the decoder* without compromising compression efficiency. Incurring the low encoding complexity of intra-frame (still-image) video coding, PRISM approaches the high compression efficiency of full-motion inter-frame video coding, while simultaneously offering natural robustness to the drift problem and channel loss in an easily tunable way. These traits make it well-matched to applications involving multimedia transmission over wireless networks (such as 802.11 or standard cellular or video sensor networks) which are characterized by the requirements of (i) computational power-efficiency at the mobile terminals/sensor nodes due to battery life considerations; (ii) high compression efficiency due to scarcity of wireless bandwidth; and (iii) robustness to channel loss due to the wireless medium. Preliminary simulations confirm the efficacy of the proposed paradigm.

1 Introduction

We are at the dawn of a new era where traditional views of video transmission (primarily television broadcast models) are being challenged. With the expected proliferation of digital cameras and their ability to be embedded in ordinary cellular phones (a number of these phones from NTT Docomo and their competitors are already flooding the Japanese telecom market), as well as the emergence of low-power surveillance and multimedia sensor networks, the days of typecasting media transmission as a "downlink" experience (e.g., TV broadcast) are over. Recall that existing video codec architectures have been driven primarily by the television broadcast model of a single possibly complex encoder and a multitude of cheap receivers. Not surprisingly, currently popular video compression standards such as H.263 and MPEG [1, 2] are based on the philosophy of a computationally "heavy" encoder and a "light" decoder. For example, the video encoder is the computational workhorse of the video codec, with its computational complexity dominated by the motion compensated prediction operation needed to strip temporal redundancy from video frames. The conventional video decoder on the other hand is a relatively lightweight device operating in a "slave" mode to the encoder.

Such a model is obviously at complete odds with the emerging class of "uplink" rich media applications such as video transmission over wireless networks such as wireless LANs (e.g., 802.11 based networks) and low-power video sensor networks such as those for surveillance or security applications. The architectural requirements here include:

1. **low-power and computational complexity at the mobile station (sensor node) for both encoding and decoding of video:** this is critical to prolonging battery life of these low-power devices;

2. **high compression efficiency:** both bandwidth and transmission power are at a premium, calling for maximal compression efficiency to minimize the number of transmitted bits over the wireless channel;

3. **robustness to channel loss:** the wireless medium is a harsh transmission channel, requiring resilience to packet drops or even frame drops which can lead to drift [1] between the encoder and the decoder.

Current video coding paradigms fail to simultaneously address these demanding requirements satisfactorily. The predictive or inter-frame video coding mode achieves high compression efficiency, but fails to meet the other two criteria, as it is computationally heavy at the encoder (primarily due to motion-search) while also being very fragile[2] to packet losses. Alternatively, intra-frame video coding methods (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 incur a relatively high transmit power due to poor compression efficiency. This raises the interesting question of whether it is possible to architect a new video coding paradigm that is driven to attain all these requirements. Specifically, is it possible to shift the computational burden from the encoder to the decoder without compromising compression efficiency, while additionally being robust to packet drops (see Figure 1)?

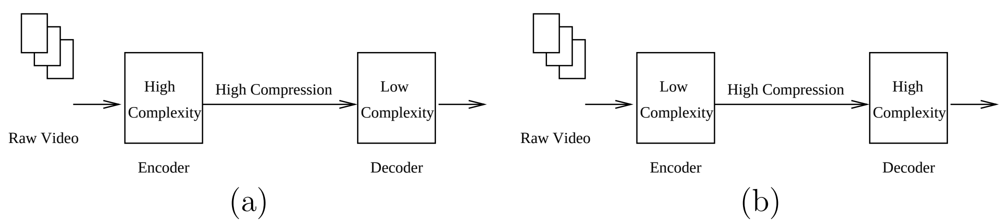


Figure 1: (a) Conventional video encoding architecture comprises of a high complexity encoder and a low complexity decoder. (b) Proposed video coding paradigm (PRISM) comprising of a low complexity encoder achieving the compression performance of the conventional framework.

[1]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.

[2]In the predictive coding setup 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.

Motivated by this, in this work, we present PRISM (Power-efficient, Robust, hIgh-compression Syndrome based Multimedia coding), a novel, robust, low-complexity, high-compression video encoding paradigm, that represents a significant departure from the traditional video coding paradigms. Leveraging the power of distributed compression methods [3, 4], PRISM incurs the encoding complexity of still image compression methods, approaches the performance of conventional video coding techniques and additionally offers the feature of robustness naturally. A typical network configuration involving the PRISM

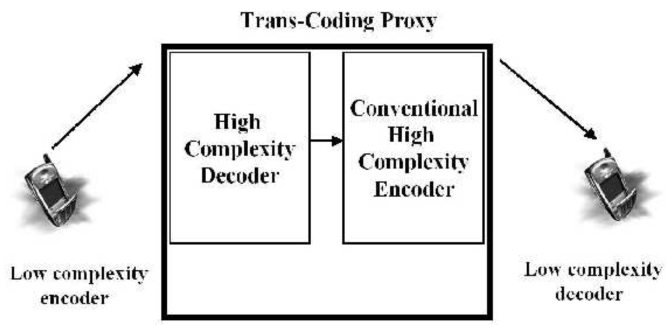


Figure 2: System level diagram for a wireless network scenario with low complexity encoding and decoding devices.

codec is as follows (See Figure 2). The uplink consists of a transmit mobile station (sensor node) employing the low-complexity PRISM encoder interfaced to a PRISM decoder in the base station or the access point. The base station has a “trans-coding proxy” that efficiently converts the PRISM bit-stream into a standard bit-stream (e.g., the output of an 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 encoding and decoding algorithms.

2 Basic Concepts

Our goal is to achieve the high compression efficiency of the predictive-coding mode while incurring the low encoding complexity 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 [5] (See Figure 3).

2.1 Illustrative Example for Coding with Side Information

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

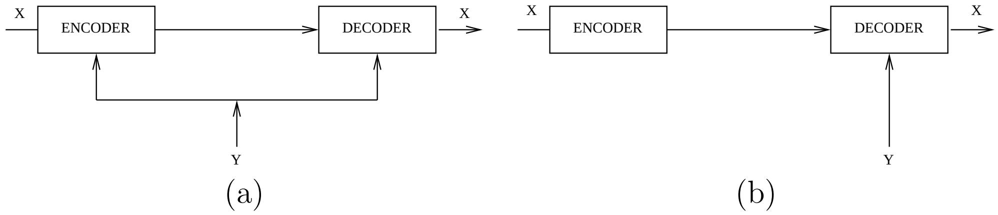


Figure 3: X and Y are correlated, length 3-bit binary data equally likely taking each of the 8 possible values, individually. The Hamming distance between the codeword for X and that for Y is at most 1. (a) Both encoder and decoder use the side information Y which is correlated to X. Here X can be encoded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

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. 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)) the encoder for X does not have access to Y. The performance of this scenario is thus limited by that of the first scenario. 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? The surprising answer 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 [3, 4] that, in theory, 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 [6] of distance 3 and the other sets are cosets [7, 8] 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 [7, 8]. 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 for some reason 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**.

We point out that the implications of the universal aspect of the coding with side information framework were discussed in [9] in an information theoretic setting. There, the idea of the same bit-stream serving to cater multiple receivers with different side informations was presented. From a practical point of view, this has the potential to combat the drift problem where the encoder does not know the contents of the frame memory (side information) at the decoder.

2.2 The PRISM approach

Motivated by the above example, we consider our video coding problem now. Let $\mathbf{X}$ denote the current macro-block to be encoded (e.g., $\mathbf{X}$ is a vector of size 256 if macroblocks of size 16×16 are chosen) . Let $\mathbf{Y}$ denote the best (motion-compensated) predictor for $\mathbf{X}$ in the previous frame and let $\mathbf{Y} = \mathbf{X} + \mathbf{N}$ (We model $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{N}$ as Gaussian random vectors.). We first encode $\mathbf{X}$ in the intra-coding mode to come up with the quantized codeword for $\mathbf{X}$. Now, using the insight from the above example, we find a channel code that is matched to the "correlation noise" $\mathbf{N}$, and use that to partition the quantized codeword space of $\mathbf{X}$. 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, however, that while in the above example we were dealing with relatively simple discrete sources exhibiting simple correlation, in the video case we are dealing with real-valued sources and potentially unbounded correlation noises. 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.

3 PRISM: Implementation

We now briefly present some implementation details of the PRISM approach. This will shed light into the various features of PRISM that are useful in an end-to-end setting, and also help understand various encoding/decoding complexity issues.

3.1 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 briefly enlist the basic steps in the encoding, which proceeds block-by-block.

1. **Transform Coding:** Every block is first transformed from the pixel domain to the frequency domain using the two-dimensional discrete cosine transform (DCT). This is done so as to more easily exploit the spatial correlation in the block. This process incurs the encoding complexity of intra-coding.

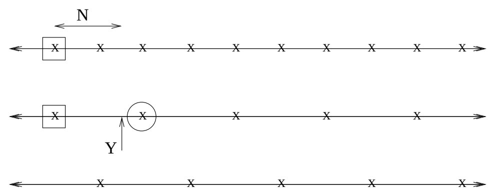


Figure 4: The top line shows the quantized codewords for X. The bottom two lines show the two partitions of the quantized codeword space of X. The box shows the observed codeword which lies in the first partition. The magnitude of N is more than the quantizer step size. Hence, the decoder decodes the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

2. **Scalar Quantization:** The DCT coefficients which are real numbers, need to be quantized before encoding. For quantization, the choice of the step size is proportional to standard deviation of $\mathbf{N}$. If a very fine step size is chosen to encode $\mathbf{X}$, then there can be decoding errors, since the codewords will be too "close" so that the side information $\mathbf{Y}$ cannot disambiguate them correctly. This is illustrated through the example in Figure 4. Here the top line shows the quantized codeword set for $\mathbf{X}$, and the two bottom lines show the partition of the space of quantized codewords. The rectangular box shows the observed codeword which lies in the first partition. Since the magnitude of $\mathbf{N}$ is more than the quantization step size, the decoder uses the side information $\mathbf{Y}$ to decode the incorrect (circled) codeword.

3. **Syndrome Encoding:** Now the space of quantized codewords which has been appropriately generated using the statistics of $\mathbf{N}$ can be partitioned using a Euclidean space trellis channel code [8, 10] analogous to the repetition channel code used to partition the source codeword space in the example in Section 2.1. In our particular implementation, we use a memory-7 rate-1/2 trellis code from [7]. 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 [6] of the trellis code.

4. **Refinement Quantization:** A target reconstruction quality 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). When the coefficients that are syndrome encoded, the choice of the base quantization step size is limited by $\mathbf{N}$. This is done so as to minimize the probability of decoding error. To attain the target quantization step size, the coefficient need to be re-quantized

further. This is accomplished in the refinement quantization stage. We note that the combination of scalar quantization followed by refinement quantization is a functional counterpart of the information theoretic concept of inverse water-pouring for the source coding with side information problem [11]. This is because for a large value of N the scalar quantization step-size is coarse and hence to get to the target step-size more bits need to be spent in the refinement stage. This amounts to inverse water-pouring on the innovations process.

3.2 Decoding

1. **Syndrome Decoding:** All the sequences that are labeled by the received syndrome can be represented on a trellis. The Viterbi algorithm [7] can be used on the 128-state rate-1/2 trellis to identify the correct sequence from the from the set of candidate sequences that is "nearest" to the candidate predictor. The PRISM framework allows for flexibility in the method of choice of the candidate predictor.

2. **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. 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.

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

4 PRISM: Features

We briefly summarize the salient features of the PRISM framework in this section. Firstly, the PRISM paradigm is characterized by **low encoding complexity** which is nearly that of intra-coding. Moreover, the PRISM framework represents a **parallelizable Framework**. The block-by-block encoding approach used in PRISM exhibits data level parallelism and is well-suited for implementation over multi-threaded architectures.

Secondly, the PRISM framework also exhibits inherent **robustness**. The PRISM encoding framework is a *joint source-channel coding framework* and is more robust to transmission losses than the conventional predictive coding paradigm. The conventional paradigm exhibits fragility in the sense that the loss of the predictor can render the residue information useless since the residue information is dependent on the predictor for decoding. The universality of the syndrome encoding paradigm (as explained in Section 2.1) which ensures that the same partitioning works for all realizations of the sources as long as they satisfy the joint statistics is what sets PRISM apart from the conventional approach. For example, if the frame memory does not have the previous frame due to some reason 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 partitioning it would still be usable for decoding. This is of significant value in dealing with the drift problem.

Finally, the PRISM framework is also associated with a **probability of decoding error**. As alluded to in Section 2.2, probability of decoding error is an artifact of the side information coding paradigm and is one of the drawbacks of PRISM. It can potentially lead to erroneous decoding of some blocks. If not checked, the effects of erroneous decoding can

Sequence	Rate (bits)	H.263+ PSNR (dB)	PRISM PSNR (dB)
Football	1400000	35.42	34.20
Euronews	1560000	36.91	35.61

Table 1: A comparison of the compression performance of PRISM with an H.263+ video coding system. Rate measures the total number of bits required to code the luminance part of the bit-stream. The PSNR is averaged over 15 frames.

propagate resulting in displeasing visual artifacts. This can be dealt with, first, through the deployment of sophisticated error concealment algorithms which can minimize these effects. Further, 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.

5 Simulation Results

In this section, we present some preliminary simulation results that illustrate the various features of PRISM. The current implementation of our coder 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 a part of the ongoing work.

We present results obtained for the first 15 frames of the Football video sequence (352 x 240) and the Euronews video sequence (320 x 240) during our experiments. Both sequences have high motion content and they were chosen to test the validity of the PRISM paradigm. The reference system is an implementation of the H.263+ [1] video coder obtained from University of British Columbia, Vancouver. The first frame in both cases 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.

We tested PRISM for performance comparison with H.263+ [1] from a pure compression point of view (see Table 1). We note that the current implementation of the proposed video coding paradigm performs within 1.2-1.3 dB of H.263+.

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. This leads to drift which accumulates and propagates in the H.263+ case. In contrast, the decoded quality is moderately affected [3] in PRISM and drift does not occur. Figure 5 compares the decoded visual quality for the Football sequence using PRISM and H.263+. Figures 5 (a), (c) and (e) show respectively the decoded first, third and the fourteenth frames for the PRISM paradigm. Figures 5 (b), (d) and (f) show respectively the decoded first, third and the fourteenth frames for the H.263+ coder. The drop in quality for PRISM is 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 (of the order of 8 dB) leading to displeasing visual artifacts (see Figure 5 (d) and 5 (f)). These experiments clearly illustrate the inherent robustness of PRISM. To

[3]In practice, we observed for our proposed system that there is an increase in the number of decoding errors when the third frame is decoded using the first frame as the side information. However, we believe that such errors can be concealed using post-processing mechanisms.

summarize, not only does the compression performance of PRISM system approach that of the predictive coding framework, PRISM is also inherently more robust.

6 Conclusions and Further Work

We have introduced PRISM – a novel, low encoding complexity, high performance and robust video coding paradigm. Under this paradigm, the encoding and the decoding complexities are roughly swapped with respect to the conventional paradigm resulting in a "light" encoder "heavy" decoder architecture.

Our present implementation of the framework, although promising, is far from complete and can be substantially enriched. Part of our ongoing work includes developing protocols/algorithms for implementing an end-to-end wireless system as in Figure 2 for extensively testing the PRISM algorithm.

References

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

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

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

[4] 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.

[5] 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.

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

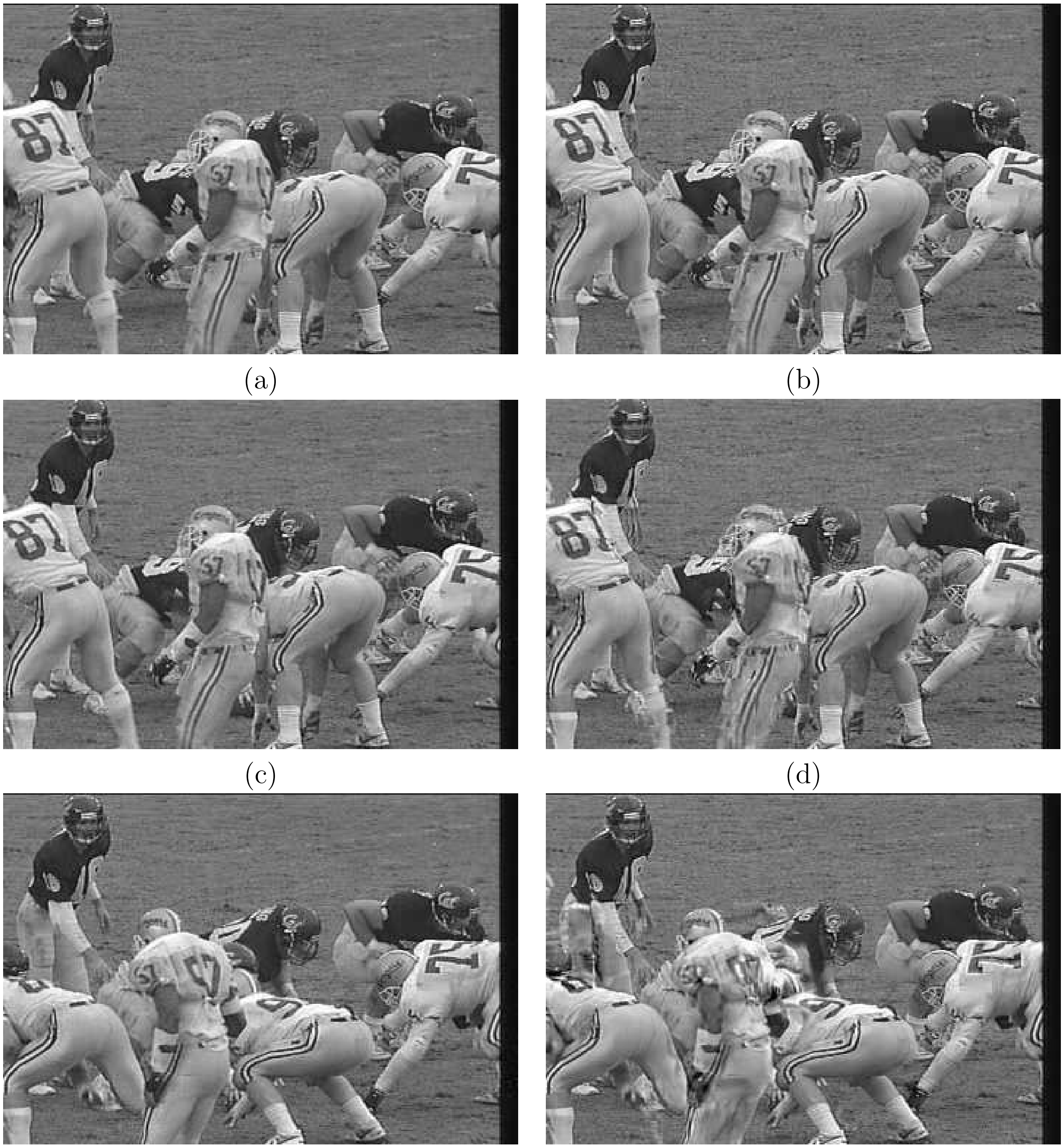
[7] G. D. Forney, "Coset Codes-Part I: Introduction and Geometrical Classification," *IEEE Transactions on Information Theory*, vol. 34, pp. 1123–1151, September 1988.

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

[9] S. S. Pradhan and K. Ramchandran, "Multiuser Successive Refinement," *Proceedings Conference on Information Sciences and Systems (CISS)*, March, Princeton, NJ, 2002.

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

[11] 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.



(a) (b) (c) (d) (e) (f)

Figure 5: 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 5 (a), (c) and (e) show respectively the decoded first, third and the fourteenth frames for PRISM. Figures 5 (b), (d) and (f) show the same for the H.263+ coder. We see in Figure 5 (d) that displeasing visual artifacts arise because of the drift and Figure 5 (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 5 (f) while it is fairly clear in Figure 5 (e).