PRISM: A REALTIME MULTIMEDIA CODING ARCHITECTURE FOR WIRELESS NETWORKS

by

Rohit Puri and Kannan Ramchandran

Memorandum No. UCB/ERL M02/22

20 May 2002

PRISM: A REALTIME MULTIMEDIA CODING ARCHITECTURE FOR WIRELESS NETWORKS

by

Rohit Puri and Kannan Ramchandran

Memorandum No. UCB/ERL M02/22

20 May 2002

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

PRISM: A Realtime Multimedia Coding Architecture for Wireless Networks

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

May 20, 2002

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 interframe video coding, while simultaneously offering natural robustness to 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. A typical uplink scenario consists of a wireless PRISM encoder at the transmit Mobile Station (or sensor node) interfaced to a PRISM decoder in the base-station Access Point (or controller node). Alternatively, it is possible to have a PRISM transcoder at the base station that efficiently converts the PRISM bit-stream to a standard video bit-stream for a standard low-complexity decoder at the downlink Mobile Station (or sensor node). The PRISM transcoding architecture thereby shields the computational burden from both low-power end-points. Preliminary simulations confirm the efficacy of the proposed paradigm.

1 Introduction

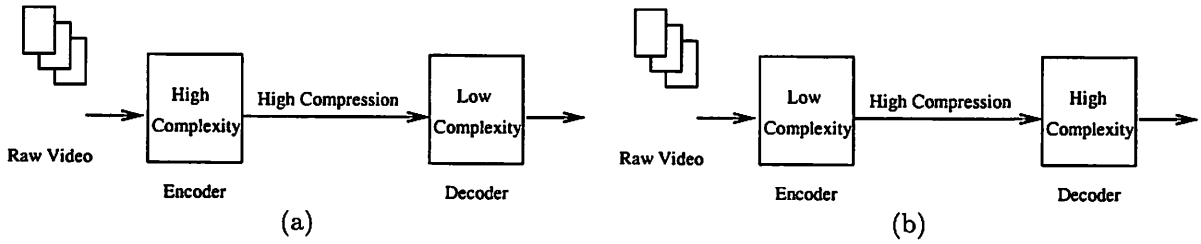

Figure 1: (a) Conventional video encoding architecture comprises of a high complexity encoder and a low complexity decoder. (b) Proposed video coding paradigm (PRISM) comprising of a low complexity encoder and achieving the compression performance of the conventional framework.

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.

This then calls for a serious questioning (and possible rehaul) of the fundamental principles underpinning current multimedia compression paradigms. 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 (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.). 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 those outlined earlier. Of particular interest are 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. Multimedia over wireless for uplink transmission applications places new architectural demands on video codecs that are an antithesis to existing architectures. Specifically, the requirements 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 due to bursty errors caused by deep fades in the channel.

Current video coding paradigms fail to simultaneously address these demanding requirements satisfactorily. Full-motion interframe video coders achieve high 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 (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

[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.

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)? Such a paradigm, if possible, would be well-suited to address the demands of these emerging class of multimedia applications around low-power wireless networks.

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 significant feature of PRISM is that it roughly swaps the encoder and decoder complexity with respect to the conventional framework. In fact, while tasks like motion prediction are performed at the encoder in the conventional paradigm, they are a responsibility of the decoder in this new paradigm. 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". It opens up the opportunity of a whole new set of creative algorithms/techniques for motion estimation, post processing and other signal processing tasks.

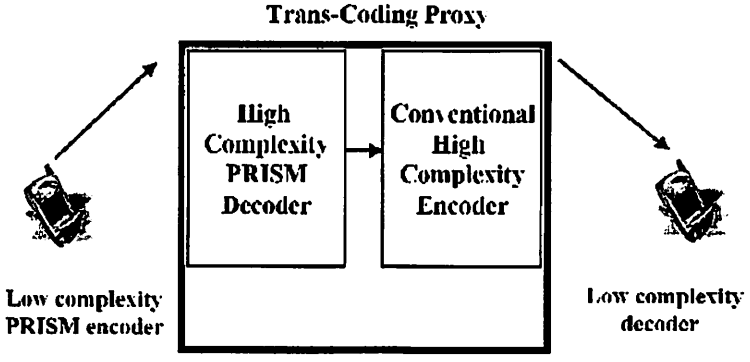

Figure 2: System level diagram for a wireless network scenario with low complexity encoding and decoding devices.

A typical network configuration involving the PRISM 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., that output by 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.

The idea of transferring the computational burden to the network was first presented in [5] 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 two different counts: first, PRISM is naturally a joint source channel coding framework unlike the purely source coding framework of [5]. Further, the work of [5] is predicated on having

network feedback. In PRISM, network feedback while serving to enhance the overall performance, is not required.

In this work, we introduce the PRISM framework, enlist its various enabling features and point out its limitations. Through our simulation results we confirm the validity of the proposed paradigm. Our ongoing work includes extensive testing of the end-to-end performance of the proposed framework in a setting as described above.

2 Basic Concepts

2.1 Background

Let us first provide a brief summary of the current video coding paradigm. A video sequence is a collection of images (also called frames) in time. It is typically highly spatio-temporally redundant with the temporal redundancy being particularly important. For the purpose of encoding, each of these frames are decomposed into non-overlapping 16x16 spatial blocks called **macro-blocks**. These are encoded in either of the following two modes.

1. Intra-Coding (I) Mode: The intra-coding mode is the image compression mode in video coding. It exploits only the spatial correlation correlation present in the video frame. Each spatial block is first transformed using the Discrete Cosine Transform (DCT), then the transformed coefficients are quantized and entropy coded using run-length Huffman coding. Since this mode does not exploit the temporal correlation in video, it typically achieves **poor compression**. However, it incurs a **low encoding complexity** (that of image compression) and is **robust** in the sense that it is a self-contained description of the block being encoded that is not dependent on anything else.

2. Predictive or Inter-Coding (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. To exploit the temporal correlation, an operation called **motion estimation** is performed. Here, typically, an exhaustive search is done in the previous frame (which is present in the frame memory at both the encoder and the decoder) within the confines of a search window to obtain a block that is the "closest" match to the block that is being encoded in the current frame. This "closest" matching block is then used as a predictor for the current block and the prediction error so obtained is encoded using DCT followed by quantization and entropy coding. The coordinates of the predictor block, called **motion vectors** are transmitted to the decoder along with the encoded residue so that the decoder can refer to the frame memory and use the same predictor to decode the residue and thus the operation of the encoder and the decoder stays synchronized. This coding mode 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.

Based on the encoding modes used for the blocks, the frames themselves are also classified as intra-coded or inter-coded. An intra-coded frame has all the blocks encoded in the intra-coding mode whereas an inter-coded frame can have both inter-coded and intra-coded blocks. Typically, a video sequence is divided into GOPs (Group of Pictures) containing about 12 to 15 frames. The first frame in a GOP is encoded in the intra mode and the remaining in the inter mode.

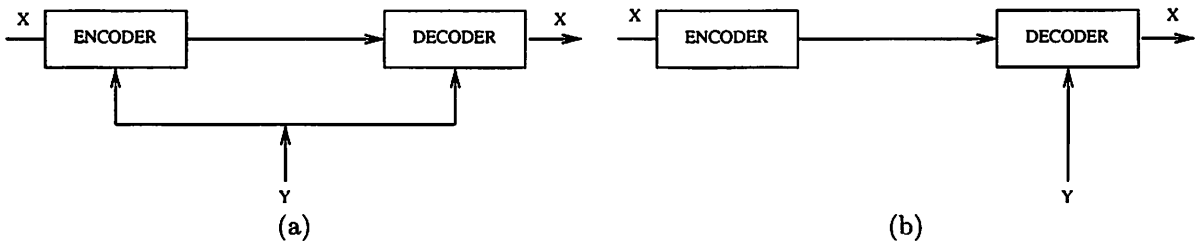

Figure 3: X and Y are correlated, length 3-bit binary data equally likely taking each of the 8 possible values, individually. The Hamming distance between the codeword for X and that for Y is at most 1. (a) Both encoder and decoder have access to side information Y which is correlated to X. Here X can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

2.2 Illustrative Example for Coding with Side Information

Our high-end 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 [6] (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. 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)), 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 [7] of distance 3 and the other sets are cosets [8, 9] 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 [8, 9]. 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, 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**.

2.3 The PRISM approach

Motivated by the above example, we consider our video coding problem now. Let $\mathbf{X}$ denote the current macroblock 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 enlist the main 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.

2. **Classification:** Typical video sequences are heterogeneous sources. Within the same sequence, some blocks that are a part of the scene background do not change much with time. That is, they are highly correlated with their predictors (small $\mathbf{N}$). 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 $\mathbf{N}$). Thus within the same frame, different blocks exhibit different degrees of correlation with the previous frame. In order to match the channel code to the block, we classify blocks based on the correlation with the previous frame, and then use the statistics of $\mathbf{N}$ corresponding to the particular class to dictate the appropriate partitioning strategy. In our current implementation, we use the energy in the block frame differences (a simple difference between the current block and the block in the previous frame in the same location) as a cue to classify the current block. We use a total of 16 (4 bits) coding modes or classes. 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 14 different syndrome coding modes in between these two extremes. The main computation carried out here is the evaluation of the block frame difference which is small (linear in the number of coefficients) in comparison with the DCT in the first step (quadratic in the number of coefficients).

3. **Base Scalar Quantization:** The DCT coefficients which are real numbers, need to be quantized before encoding. For quantization, the choice of the step size is limited by the statistics 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.

   Thus, each of the elements of $\mathbf{X}$ is quantized with a step size proportional to the standard deviation of the corresponding element in $\mathbf{N}$.

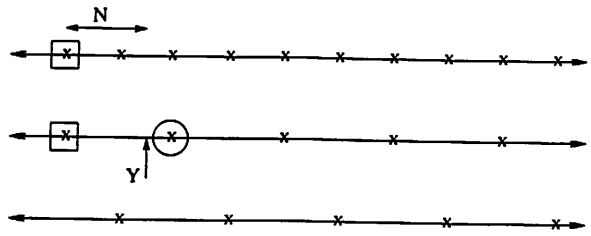

Figure 4: The top line shows the quantized codewords for X. The bottom two lines show the two partitions of the quantized codeword space of X. The box shows the observed codeword. The observed codeword lies in the first partition. The magnitude of N is more than the quantizer step size. Hence, the decoder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

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

5. **Syndrome Encoding:** Now the space of quantized codewords which has been appropriately generated using the statistics of N can be partitioned using a Euclidean space trellis channel code [9, 10]. This is analogous to the repetition channel code that was used to partition the source codeword space in the example in Section 2.2. In our particular implementation, we use a memory-7 rate-1/2 trellis code from [8]. 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.

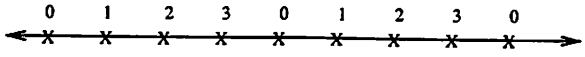

Figure 5: Mod-4 labeling of the space of quantized codewords of source X.

The transmission or the coset index rate [3] 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 [7] of the trellis 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.

6. **"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

[2] 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).

[3] 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.

entropy coded using run-length Huffman coding.

7. **Refinement Quantization:** A target reconstruction quality [4] 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. 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.

   The operation of base quantization followed by refinement quantization is a practical illustration of the information theoretic concept of *water pouring* [11]. 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.

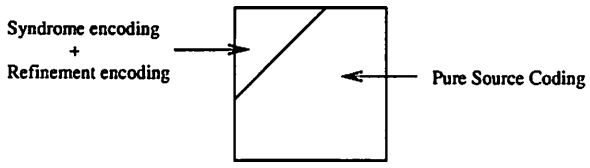

Figure 6: After zig-zag scanning, the first few coefficients are encoded by syndrome encoding followed by refinement encoding. The remaining fraction of coefficients are purely source coded (intra-coded).

8. **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 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.

The bit-stream associated with a block is illustrated in Figure 7.

[4] 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.

Figure 7: Bit-stream associated with a block.

To summarize, the main complexity in the encoding process is incurred in steps 1 (complexity of the DCT) and 6 (complexity of entropy coding). 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.

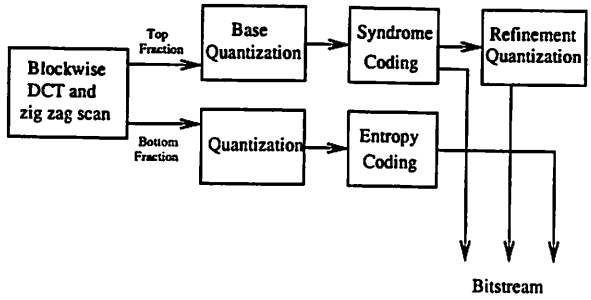

Figure 8: Functional block diagram of the encoder.

3.2 Decoding

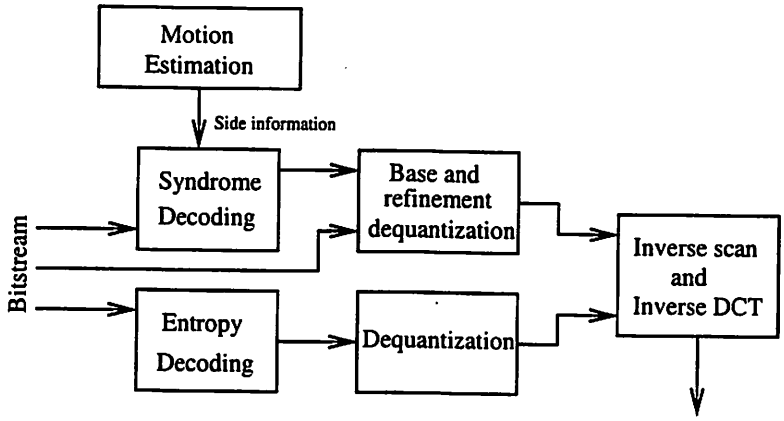

Figure 9: Functional block diagram of the decoder.

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

1. **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, exhaustive half pixel motion search is used to obtain various candidate predictors as is also done at the encoding side in the standard video algorithms [1, 2]. We reiterate that the framework is very general so as to accommodate any other sophisticated motion estimation procedures such as multi-frame prediction, optical

flow, control grid interpolation 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 [8], this decoding can be accomplished using the Viterbi algorithm [12]. 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.

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.

4 PRISM: Features

We point out 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 3, 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. **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.

3. **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" which is endowed with lots of processing power. The deployment of sophisticated motion prediction algorithms (e.g., multi-frame prediction, optical flow) and post-processing algorithms for de-blocking and error concealment for increasing the picture quality are easily accommodated within the PRISM framework.

4. **High Compression:** The principal goal of the PRISM approach is to have a substantial reduction in the encoding complexity while not compromising the compression efficiency of the encoder. Ideally, the framework approaches the performance of inter-coding while incurring the encoding complexity of intra-coding. Details of the performance of the current implementation are presented in Section 5. Efficient compression reduces the size of the bit-stream and thus minimizes the total transmitted power prolonging the cell/sensor unit battery life.

5. **Robustness:** Not only does the PRISM framework transfer the complexity from the encoder to the decoder, but it also possesses the feature of robustness inherently. 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 2.2) 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 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 partitioning it would still be usable for decoding. This is of significant value in dealing with the drift problem [5]. In Section 5 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. That is, by decreasing the coding efficiency (incurring more rate for coding the same quality) the bit-stream can be automatically made more robust to transmission losses. This is of great value in scenarios where network feedback is available. The encoder upon estimating the channel loss rate from the feedback can dynamically adapt the coding strategy to match it to the channel conditions at hand. 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.

[5]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.

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. To summarize, the PRISM encoder can utilize network feedback to dynamically adapt its coding strategy to match the channel conditions so as to maximize the delivered quality at the decoder.

6. **Probability of Decoding Error:** As alluded to in Section 2.3, 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 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. The encoder can then send the corresponding blocks in an intra-coded fashion so as to stop the error propagation. Thus, the network by virtue of feedback, can serve to significantly improve the decoded quality at the decoder by checking error propagation.

7. **Efficient Trans-coding:** The trans-coding 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.

8. **A Flexible Encoding Syntax:** Since multimedia coding standards constrain the bit-stream syntax at the encoder, the fact that all the sophisticated processing is 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". It creates opportunity for a whole new set of creative algorithms/techniques for motion estimation, error concealment, post processing etc. and other sophisticated signal processing tasks.

In a nutshell, the various features of PRISM make it naturally suited for multimedia coding scenarios for battery power constrained devices as in wireless networks.

5 Preliminary 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 latter sequence which was grabbed off the satellite news channel Euronews aired in Switzerland attributes its motion to camera pan and zoom. The reference system is an

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.

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. Table 1 summarizes the performance of the proposed video coder in comparison with the H.263+ coder. From a pure compression point of view, we note that the current implementation of the proposed video coding paradigm performs worse than H.263+ by about 1.2-1.3 dB.

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 moderately affected [6] in PRISM and drift does not occur. Figure 10 shows the decoded visual quality for the Football sequence in both cases. Figures 10 (a), (c) and (e) show respectively the decoded first, third and the fourteenth frames for the PRISM paradigm. Figures 10 (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 (of the order of 8 dB) leading to displeasing visual artifacts (see Figure 10 (d)) which propagate and accumulate through the remainder of the sequence (see Figure 10 (f)). In particular, the jersey number of the football player with jersey 57 is not even visible in Figure 10 (f) while it is fairly clear in Figure 10 (e). These experiments thus point out the inherent robustness of PRISM.

To summarize, while the compression performance of PRISM system approaches that of the predictive coding framework, the framework is 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

[6] 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.

wireless system as in Figure 2 for extensively testing the PRISM algorithm.

One direction of research would be improving the performance of PRISM as a codec e.g., enabling the operation of this framework under low bit rate regimes. Another possibility could be the deployment of richer motion estimation paradigms (such as multi-frame prediction, optical flow etc.) and post processing mechanisms at the decoder. Still another direction could be developing the network feedback mechanism so as to dynamically adapt the PRISM encoding strategy for the channel at hand.

7 Acknowledgments

The authors would like to thank Prof. Sandeep Pradhan (University of Michigan, Ann Arbor) for inspiring discussions that contributed significantly towards the development of the PRISM framework.

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] 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.

[6] 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.

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

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

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

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

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

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

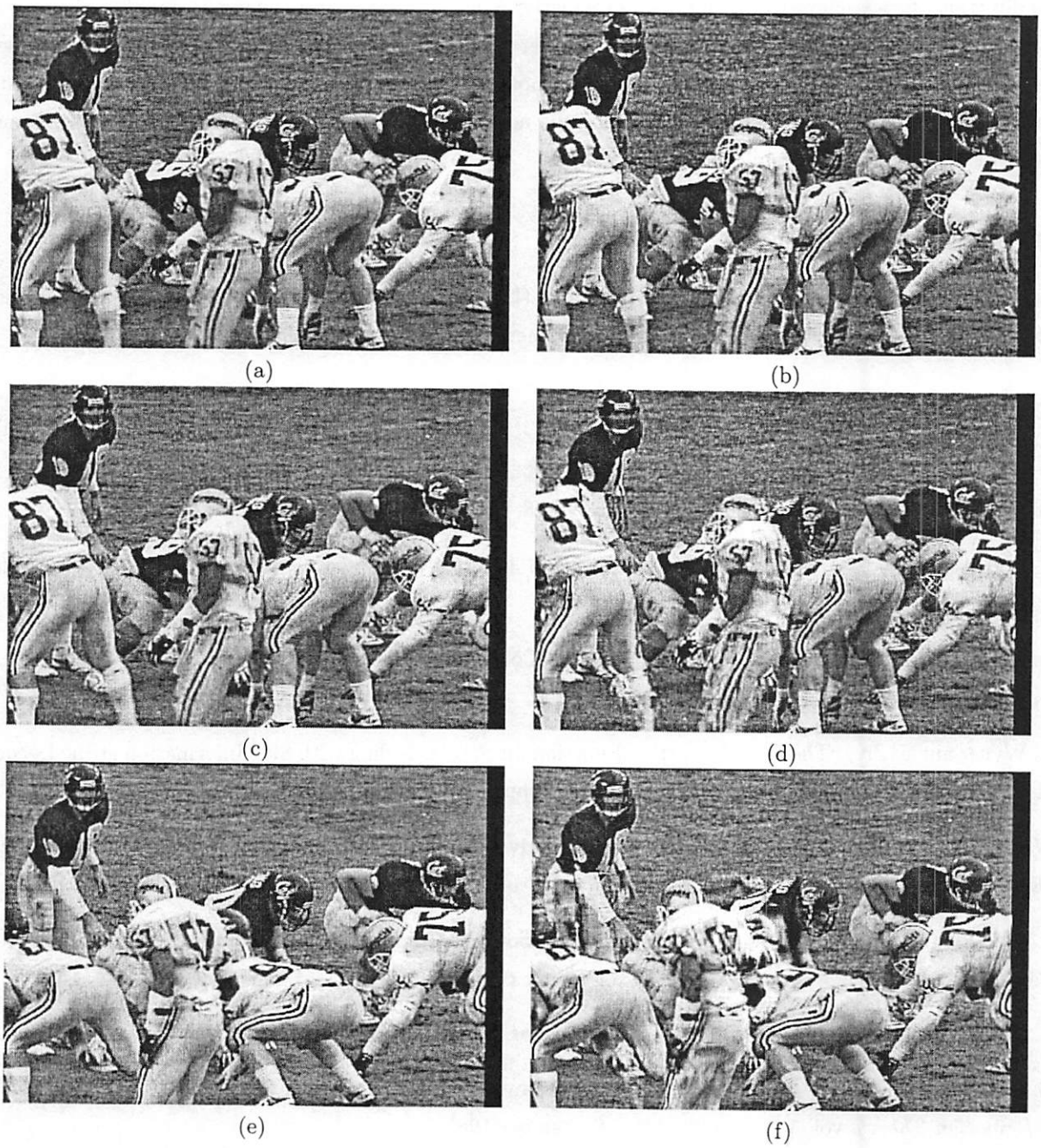

Figure 10: Performance of PRISM and H.263+ coder in the case of frame loss. Fifteen frames of the football video sequence were encoded in both cases and the second decoded frame was removed from the frame memory in both cases. The third frame was decoded using the first frame as side information for the proposed paradigm and a predictor for H.263+. Figures 10 (a), (c) and (e) show respectively the decoded first, third and the fourteenth frames for PRISM. Figures 10 (b), (d) and (f) show the same for the H.263+ coder. We see in Figure 10 (d) that displeasing visual artifacts arise because of the drift and Figure 10 (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 10 (f) while it is fairly clear in Figure 10 (e).