CERIAS Tech Report 2001-113
Wavelet Based Rate Scalable Video Compression
by K Shen, E Delp
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Wavelet Based Rate Scalable Video Compression

Ke Shen and Edward J. Delp *
Video and Image Processing Laboratory (*VIPER*)
School of Electrical and Computer Engineering
Purdue University

Corresponding Author:
Professor Edward J. Delp
School of Electrical and Computer Engineering
1285 Electrical Engineering Building
Purdue University
West Lafayette, IN 47907-1285
USA
Telephone: +1 765 494 1740
Fax: +1 765 494 0880
Email: ace@ecn.purdue.edu

*This work was supported by a grant from the AT&T Foundation, and the Rockwell Foundation. Address all correspondence to E. J. Delp, ace@ecn.purdue.edu, http://www.ece.purdue.edu/∼ace, or +1 765 494 1740.

Abstract

In this paper, we present a new wavelet based rate scalable video compression algorithm. We shall refer to this new technique as the **Scalable Adaptive Motion COmpensated Wavelet** (***SAMCoW***) algorithm. *SAMCoW* uses motion compensation to reduce temporal redundancy. The prediction error frames and the intra-coded frames are encoded using an approach similar to the embedded zerotree wavelet (EZW) coder. observed. An adaptive motion compensation (AMC) scheme is described to address error propagation problems. We show that using our AMC scheme the quality of the decoded video can be maintained at various data rates. correlation. large transitions, it is highly likely for the luminance signal to have large transitions. We also describe an EZW approach that exploits the interdependency between color components in the luminance/chrominance color space. We show that in addition to providing a wide range of rate scalability, our encoder achieves comparable performance to the more traditional hybrid video coders, such as MPEG1 and H.263. Furthermore, our coding scheme allows the data rate to be dynamically changed during decoding, which is very appealing for network oriented applications.

Index Terms: Rate scalable, video compression, motion compensation, wavelet transform.

1. Introduction

Many applications require that digital video be delivered over computer networks. The available bandwidth of most computer networks almost always pose a problem when video is delivered. A user may request a video sequence with a specific quality. However, the variety of requests and the diversity of the traffic on the network may make it difficult for a video server to predict, at the time the video is encoded and stored on the server, the video quality and data rate it will be able to provide to a particular user at a given time. One solution to this problem is to compress and store a video sequence at different data rates. The server will then deliver the requested video at the proper rate given network loading and the specific user request. This approach requires more resources to be used on the server in terms of disk space and management overhead. Therefore scalability, the capability of decoding a compressed sequence at different data rates, has become a very important issue in video coding. Scalable video coding has applications in digital libraries, video database system, video streaming, video telephony and multicast of television (including HDTV).

The term "scalability" used here includes data rate scalability, spatial resolution scalability, temporal resolution scalability and computational scalability. The MPEG-2 video compression standard incorporated several scalable modes, including signal-to-noise ratio (SNR) scalability, spatial scalability and temporal scalability [1, 2]. However, these modes are layered instead of being continuously scalable. Continuous rate scalability provides the capability of arbitrarily selecting the data rate within the scalable range. It is very flexible and allows the video server to tightly couple the available network bandwidth and the data rate of the video being delivered.

A specific coding strategy known as *embedded rate scalable coding* is well suited for continuous rate scalable applications [3]. In embedded coding, all the compressed data is embedded in a single bit stream and can be decoded at different data rates. In image compression, this is very similar to the progressive transmission. The decompression algorithm receives the

compressed data from the beginning of the bit stream up to a point where a chosen data rate requirement is achieved. A decompressed image at that data rate can then be reconstructed and the visual quality corresponding to this data rate can be achieved. Thus, to achieve the best performance the bits that convey the most important information need to be embedded at the beginning of the compressed bit stream. For video compression, the situation can be more complicated since a video sequence contains multiple images. Instead of sending the initial portion of the bit stream to the decoder, the sender needs to selectively provide the decoder with portions of the bit stream corresponding to different frames or sections of frames of the video sequence. These selected portions of the compressed data achieve the data rate requirement and can then be decoded by the decoder. This approach can be used if the position of the bits corresponding to each frame or each section of frames can be identified.

In this paper, we propose a new continuous rate scalable hybrid video compression algorithm using the wavelet transform. We shall refer to this new technique as the **Scalable Adaptive Motion COmpensated Wavelet** (***SAMCoW***) algorithm. *SAMCoW* uses motion compensation to reduce temporal redundancy. The prediction error frames (PEFs) and the intra-coded frames (I frames) are encoded using an approach similar to embedded zerotree wavelet (EZW) [3], which provides continuous rate scalability. The novelty of this algorithm is that it uses an adaptive motion compensation (AMC) scheme to eliminate quality decay even at low data rates. A new modified zero-tree wavelet image compression scheme that exploits the interdependence between the color components in a frame is also described. The nature of *SAMCoW* allows the decoding data rate to be dynamically changed to meet network loading. Experimental results show that *SAMCoW* has a wide range of scalability. For medium data rate (CIF images, 30 frames per second) applications, the scalable range of 1 megabits per second (Mbps) to 6 Mbps can be achieved. The performance is comparable to that of MPEG-1 at fixed data rates. For low bit rate (QCIF images, 10–15 frames per

second) applications, the data rate can be scaled from 20 kilobits per second (Kbps) to 256 Kbps.

In Section 2, we provide an overview of wavelet based embedded rate scalable coding and the motivation for using motion compensated scheme in our new scalable algorithm. In Section 3, we describe our new adaptive motion compensation scheme (AMC) and *SAMCoW*. In Section 4, we provide implementation details of the *SAMCoW* algorithm. Simulation results will be presented in Section 5.

2. Rate Scalable Coding

2.1 Rate Scalable Image Coding

Rate scalable image compression, or progressive transmission of images, has been extensively investigated [4, 5, 6]. Reviews on this subject can be found in [7, 8]. Different transforms, such as the Laplacian pyramid [4], the discrete cosine transform (DCT) [6], and the wavelet transform [3, 9], have been used for progressive transmission.

Shapiro introduced the concept of embedded rate scalable coding using the wavelet transform and spatial-orientation trees (SOTs) [3]. Since then, variations of the algorithm have been proposed [10, 9, 11]. These algorithms have attracted a lot of attention due to their superb performance and are candidates for the baseline algorithms used in JPEG2000 and MPEG-4. In this section we provide a brief overview of several wavelet based embedded rate scalable algorithms.

A wavelet transform corresponds to two sets of analysis/synthesis digital filters, $g/\tilde{g}$ and $h/\tilde{h}$, where g is a high pass filter and h is a low pass filter. By using the filters g and h, an image can be decomposed into four bands. Subsampling is used to translate the subbands to a baseband image. This is the first level of the wavelet transform (Figure 1). The operations can be repeated on the low-low (LL) band. Thus, a typical 2-D discrete wavelet transform used in image processing will generate a hierarchical pyramidal structure shown in

Figure 2. The inverse wavelet transform is obtained by reversing the transform process and replacing the analysis filters with the synthesis filters and using up-sampling (Figure 3). The wavelet transform can decorrelate the image pixel values and result in frequency and spatial-orientation separation. The transform coefficients in each band exhibit unique statistical properties that can be used for encoding the image.

For image compression, quantizers can be designed specifically for each band. The quantized coefficients can then be binary coded using either Huffman coding or arithmetic coding [12, 13, 14]. In embedded coding, a key issue is to embed the more important information at the beginning of the bit stream. From a rate-distortion point of view, one wants to quantize the wavelet coefficients that cause larger distortion in the decompressed image first. Let the wavelet transform be $\boldsymbol{c} = T(\boldsymbol{p})$, where p is the collection of image pixels and c is the collection of wavelet transform coefficients. The reconstructed image $\hat{p}$ is obtained by the inverse transform $\hat{\boldsymbol{p}} = T^{-1}(\hat{\boldsymbol{c}})$, where $\hat{c}$ is the quantized transform coefficients. The distortion introduced in the image is $D(\boldsymbol{p}-\hat{\boldsymbol{p}}) = D(\boldsymbol{c}-\hat{\boldsymbol{c}}) = \sum_i D(c_i - \hat{c}_i)$, where $D(\cdot)$ is the distortion metric and the summation is over the entire image. The greatest distortion reduction can be achieved if the transform coefficient with the largest magnitude is quantized and encoded without distortion. Furthermore, to strategically distribute the bits such that the decoded image will look "natural", progressive refinement or bit-plane coding is used. Hence, in the coding procedure, multiple passes through the data are made. Let C be the largest magnitude in c. In the first pass, those transform coefficients with magnitudes greater than $\frac{1}{2}C$ are considered significant and are quantized to a value of $\frac{3}{4}C$. The rest are quantized to 0. In the second pass, those coefficients that have been quantized to 0 but have magnitudes in between of $\frac{1}{4}C$ and $\frac{1}{2}C$ are considered significant and are quantized to $\frac{3}{8}C$. Again the rest are quantized to zero. Also those significant coefficients in the last pass are refined to one more level of precision, i.e. $\frac{5}{8}C$ or $\frac{7}{8}C$. This process can be repeated until the data rate meets the requirement or the quantization step is small enough. Thus, we can achieve the

largest distortion reduction with the smallest number of bits, while the coded information is distributed across the image.

However, to make this strategy work we need to encode the position information of the wavelet coefficients along with the magnitude information. It is critical that the positions of the significant coefficients be encoded efficiently. One could scan the image in a given order that is known to both the encoder and decoder. This is the approach used in JPEG with the "zig-zag" scanning. A coefficient is encoded 0 if it is insignificant or 1 if it is significant relative to the threshold. However the majority of the transform coefficients are insignificant when compared to the threshold, especially when the threshold is high. These coefficients will be quantized to zero, which will not reduce the distortion even though we still have to use at least one symbol to code them. Using more bits to encode the insignificant coefficients results in lower efficiency.

It has been observed experimentally that coefficients which are quantized to zero at a certain pass have structural similarity across the wavelet subbands in the same spatial orientation. Thus spatial-orientation trees (SOTs) can be used to quantize large areas of insignificant coefficients efficiently (e.g. zerotree in [3]). structure per si is not a necessary condition. encoding the position information efficiently.

The EZW algorithm proposed by Shapiro [3], and the SPIHT technique proposed by Said and Pearlman [9] use slightly different SOTs (shown in Figure 4). The major difference between these two algorithms lies in the fact that they use different strategies to scan the transformed pixels. The SOT used by Said and Pearlman [9] is more efficient than Shapiro's [3].

2.2 Scalable Video Coding

One could achieve continuous rate scalability in a video coder by using a rate scalable still image compression algorithms such as [6, 3, 9] to encode each video frame. This is known as the "intra-frame coding" approach. We used Shapiro's algorithm [3] to encode each frame

of the *football* sequence. The rate-distortion performance is shown in Figure 5. A visually acceptable decoded sequence, comparable to MPEG-1, is obtained only when the data rate is larger than 2.5 Mbps for a CIF (352x240) sequence. This low performance is due to the fact that the temporal redundancy in the video sequence is not exploited. Taubman and Zakhor proposed an embedded scalable video compression algorithm using 3-D subband coding [15]. Some draw backs of their scheme are that the 3-D subband algorithm can not exploit the temporal correlation of the video sequence very efficiently, especially when there is a great deal of motion. Also since 3-D subband decomposition requires multiple frames to be processed at the same time, more memory is needed for both the encoder and the decoder, which results in delay. Other approaches to 3-D subband video coding are presented in [16, 17].

Motion compensation is very effective in reducing temporal redundancy and is commonly used in video coding. A motion compensated hybrid video compression algorithm usually consists of two major parts, the generation and compression of the motion vector (MV) fields and the compression of the I frames and prediction error frames. Motion compensation is usually block based, i.e. the current image is divided into blocks and each block is matched with a reference frame. The best matched block of pixels from the reference frame are then used in the current block. The prediction error frame (PEF) is obtained by taking the difference between the current frame and the motion predicted frame. PEFs are usually encoded using either block-based transforms, such as DCT [8], or non- block-based coding, such as subband coding or the wavelet transform. The DCT is used in the MPEG and H.263 algorithms [18, 1, 19]. A major problem with a block-based transform coding algorithm is the existence of the visually unpleasant block artifacts, especially at low data rates. This problem can be eliminated by using the wavelet transform, which is usually obtained over the entire image. The wavelet transform has been used in video coding for the compression of motion predicted error frames [20, 21]. However these algorithms are not scalable. If

we use wavelet based rate scalable algorithms to compress the I frames and PEFs, rate scalable video compression can be achieved. Recently, a wavelet based rate scalable video coding algorithm has been proposed by Wang and Ghanbari [22]. In their scheme the motion compensation was done in the wavelet transform domain. However, in the wavelet transform domain spatial shifting results in phase shifting, hence motion compensation does not work well and may cause motion tracking errors in high frequency bands. Pearlman [23, 24] has extended the use of SPIHT to describe a three dimensional SOT for use in video compression.

3. A New Approach: *SAMCoW*

3.1 Adaptive Motion Compensation

One of the problems of any rate scalable compression algorithm is the inability of the codec in maintaining a constant visual quality at any data rate. Often the distortion of a decoded video sequence varies from frame to frame. Since a video sequence is usually decoded at 25 or 30 frames per second (or 5-15 frames per second for low data rate applications), the distortion of each frame may not be discerned as accurately as when individual frames are examined due to temporal masking. Yet, the distortion of each frame contributes to the overall perception of the video sequence. When the quality of successive frames decreases for a relatively long time, a viewer will notice the change. This increase in distortion, sometimes referred to as "drift," may be perceived as an increase in fuzziness and/or blockiness. in the scene. This phenomenon can occur due to artifact propagation, which is very common when motion compensated prediction is used. This can be more serious with a rate scalable compression technique.

Motion vector fields are generated by matching the current frame with its reference frame. After the motion vector field $\boldsymbol{m}$ is obtained for the current frame, the predicted frame is generated by rearranging the pixels in the reference frame relative to $\boldsymbol{m}$. We denote this

operation by $\boldsymbol{M(\cdot)}$, or

$$\boldsymbol{p_{pred} = M(p_{ref}, m),}$$

where $\boldsymbol{p_{pred}}$ is the predicted frame and $\boldsymbol{p_{ref}}$ is the reference frame. The prediction error frame is obtained by taking the difference between the current frame and the predicted frame

$$\boldsymbol{p_{diff} = p - p_{pred}.}$$

At the decoder, the predicted frame is obtained by using the decoded motion vector field and the decoded reference frame

$$\boldsymbol{\hat{p}_{pred} = M(\hat{p}_{ref}, \hat{m}).}$$

The decoded frame, $\boldsymbol{\hat{p}}$, is then obtained by adding the $\boldsymbol{\hat{p}_{pred}}$ to the decoded PEF $\boldsymbol{\hat{p}_{diff}}$

$$\boldsymbol{\hat{p} = \hat{p}_{pred} + \hat{p}_{diff}.}$$

Usually the motion field is losslessly encoded, by maintaining the same reference frame at the encoder and the decoder, i.e. $\boldsymbol{p_{ref} = \hat{p}_{ref}}$, then

$$\boldsymbol{\hat{p}_{pred} = p_{pred}.}$$

This results in the decoded PEF, $\boldsymbol{\hat{p}_{diff}}$, being the only source of distortion in $\boldsymbol{D(p - \hat{p})}$. Thus, one can achieve better performance if the encoder and decoder use the same reference frame. For a fixed rate codec, this is usually achieved by using a prediction feedback loop in the encoder so that a decoded frame is used as the reference frame (Figure 6). This procedure is commonly used in MPEG or H.263. However, in our scalable codec, the decoded frames have different distortions at different data rates. Hence, it is impossible for the encoder to generate the exact reference frames as in the decoder for all possible data rates. One solution is to have the encoder locked to a fixed data rate (usually the highest data rate) and let the decoder run freely, as in Figure 6. The codec will work exactly as the non-scalable codec,

when decoding at the highest data rate. However, when the decoder is decoding at a low data rate, the quality of the decoded reference frames at the decoder will deviate from that at the encoder. Hence, both the motion prediction and the decoding of the PEFs contribute to the increase in distortion of the decoded video sequence. This distortion also propagates from one frame to the next within a group of pictures (GOP). If the size of a GOP is large, the increase in distortion can be unacceptable.

To maintain video quality, we need to keep the reference frames the same at both the encoder and the decoder. This can be achieved by adding a feedback loop in the decoder (Figure 7), such that the decoded reference frames at both the encoder and decoder are locked to the same data rate—the lowest data rate. We denote this scheme as *adaptive motion compensation* (AMC) [25, 26]. We assume that the target data rate $\boldsymbol{R}$ is within the range $\boldsymbol{R_L} \leq \boldsymbol{R} \leq \boldsymbol{R_H}$ and the bits required to encode the motion vector fields have data rate $\boldsymbol{R_{MV}}$, where $\boldsymbol{R_{MV}} < \boldsymbol{R_L}$. At the encoder, since $\boldsymbol{R_{MV}}$ is known, the embedded bit stream can always be decoded at rate $\boldsymbol{R_L} - \boldsymbol{R_{MV}}$, which is then added to the predicted frame to generate the reference frame $\hat{\boldsymbol{p}}_{\boldsymbol{ref}}$. At the decoder, the embedded bit stream is decoded at two data rates, the targeted data rate $\boldsymbol{R} - \boldsymbol{R_{MV}}$ and the fixed data rate $\boldsymbol{R_L} - \boldsymbol{R_{MV}}$. The frame decoded at rate $\boldsymbol{R_L} - \boldsymbol{R_{MV}}$ is added to the predicted frame to generate the reference frame, which is exactly the same as the reference frame $\hat{\boldsymbol{p}}_{\boldsymbol{ref}}$ used in the encoder. The frame decoded at rate $\boldsymbol{R} - \boldsymbol{R_{MV}}$ is added to the predicted frame to generate the final decoded frame. This way, the reference frames at the encoder and the decoder are identical, which leaves the decoded PEF $\hat{\boldsymbol{p}}_{\boldsymbol{diff}}$ as the only source of distortion. Hence, error propagation is eliminated.

3.2 Embedded Coding of Color Images

Many wavelet based rate scalable algorithms, such as EZW [3] and SPIHT [9], can be used for the encoding of I frames and PEFs. However, these algorithms were developed for grayscale images. To code a color image, the color components are treated as three individual grayscale

images and the same coding scheme is used for each component. The interdependence between the color components is not exploited. To exploit the interdependence between color components, the algorithm may also be used on the decorrelated color components generated by a linear transform. In Said and Pearlman's algorithm [9], the Karhunen-Loeve (KL) transform is used [27]. The KL transform is optimal in the sense that the transform coefficients are uncorrelated. The KL transform, however, is image dependent, i.e. the transform matrix needs to be obtained for each image and transmitted along with the coded image.

The red-green-blue (RGB) color space is commonly used because it is compatible with the mechanism of color display devices. Other color spaces are used, among these are the luminance and chrominance (LC) spaces which are popular in video/television applications. An LC space, e.g. YCrCb, YUV or YIQ, consists of a luminance component and two chrominance (color difference) components. The LC spaces are popular because the luminance signal can be used to generate a grayscale image, which is compatible with monochrome systems, and the three color components have little correlation, which facilitates the encoding and/or modulation of the signal [28, 29].

Although the three components in a LC space are uncorrelated, they are not independent. Experimental evidence has shown that at the spatial locations where chrominance signals have large transitions, the luminance signal also has large transitions [30, 31]. Transitions in an image usually correspond to wavelet coefficients with large magnitudes in high frequency bands. Thus, if a transform coefficient in a high frequency band of the luminance signal has small magnitude, the transform coefficient of the chrominance components at the corresponding spatial location and frequency band should also have small magnitude [22, 32]. In embedded zerotree coding, if a zerotree occurs in the luminance component, a zerotree at the same location in the chrominance components is highly likely to occur. This interdependence of the transform coefficients signals between the color components is incorporated

into *SAMCoW*.

In our algorithm, the YUV space is used. The algorithm is similar to Shapiro's algorithm [3]. The SOT is described as follows: The original SOT structure in Shapiro's algorithm is used for the three color components. Each chrominance node is also a child node of the luminance node at the same location in the wavelet pyramid. Thus each chrominance node has two parent nodes: one is of the same chrominance component in a lower frequency band, and the other is of the luminance component in the same frequency band. A diagram of the SOT is shown in Figure 8.

In our algorithm, the coding strategy is similar to Shapiro's algorithm. The algorithm also consists of dominant passes and subordinate passes. The symbols used in the dominant pass are positive significant, negative significant, isolated zero and zerotree. In the dominant pass, the luminance component is first scanned. For each luminance pixel, all descendents, including those of the luminance component and those of the chrominance components, are examined and appropriate symbols are assigned. The zerotree symbol is assigned if the current coefficient and its descendents in the luminance and chrominance components are all insignificant. The two chrominance components are alternatively scanned after the luminance component is scanned. The coefficients in the chrominance that have already been encoded as part of a zerotree while scanning the luminance component are not examined. The subordinate pass is essentially the same as that in Shapiro's algorithm.

4. Implementation of *SAMCoW*

The discrete wavelet transform was implemented using the biorthogonal wavelet basis from [33] — the 9-7 tap filter bank. Four to six levels of wavelet decomposition were used, depending on the image size.

The video sequences used in our experiments use the YUV color space with color components downsampled to 4:1:1. Motion compensation is implemented using macroblocks,

i.e. 16x16 for the Y component and 8x8 for the U and V components, respectively. The search range is ± 15 luminance pixels in both the horizontal and vertical directions. Motion vectors are restricted to integer precision. The spatially corresponding blocks in Y, U and V components share the same motion vector. One problem with block based motion compensation is that it introduces blockiness in the prediction error images. The blocky edges cannot be efficiently coded using the wavelet transform and may introduce unpleasant ringing effects. To reduce the blockiness in the prediction error images, overlapped block motion compensation is used for the Y component [34, 20, 19]. Let $L^{i,j}$ be the ith row and jth column macroblock of the luminance image and let $m^{i,j} = [m_x^{i,j}, m_y^{i,j}]$ be its motion vector. The predicted pixel values for $L^{i,j}$ are the weighted sum

$$\begin{aligned} L^{i,j}(k,l) = \; & w_c(k,l) L_{ref}^{i,j}(k + m_y^{i,j}, l + m_x^{i,j}) \\ & + w_t(k,l) L_{ref}^{i,j}(k + m_y^{i-1,j}, l + m_x^{i-1,j}) \\ & + w_b(k,l) L_{ref}^{i,j}(k + m_y^{i+1,j}, l + m_x^{i+1,j}) \\ & + w_l(k,l) L_{ref}^{i,j}(k + m_y^{i,j-1}, l + m_x^{i,j-1}) \\ & + w_r(k,l) L_{ref}^{i,j}(k + m_y^{i,j+1}, l + m_x^{i,j+1}), \end{aligned}$$

where $k, l \in \{0 \ldots 15\}$. The weighting values for the current block are

$$w_c = \begin{pmatrix} 4 & 5 & 5 & 5 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 5 & 5 & 5 & 4 \\ 5 & 5 & 5 & 5 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 5 & 5 & 5 & 5 \\ 5 & 5 & 6 & 6 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 6 & 6 & 5 & 5 \\ 5 & 5 & 6 & 6 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 6 & 6 & 5 & 5 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 6 & 6 & 7 & 7 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 7 & 7 & 6 & 6 \\ 5 & 5 & 6 & 6 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 6 & 6 & 5 & 5 \\ 5 & 5 & 6 & 6 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 6 & 6 & 5 & 5 \\ 5 & 5 & 5 & 5 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 5 & 5 & 5 & 5 \\ 4 & 5 & 5 & 5 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 5 & 5 & 5 & 4 \end{pmatrix} /8.$$

The weighting values for the top block are

$$\boldsymbol{w_t} = \begin{pmatrix} 2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2\\ 1&1&2&2&2&2&2&2&2&2&2&2&2&2&1&1\\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} /8,$$

and the weighting values for the left block are

$$\boldsymbol{w_t} = \begin{pmatrix} 2&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&2&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 2&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} /8$$

The weighting values for the bottom and right blocks are $\boldsymbol{w_b(i,j) = w_t(15-i,j)}$ and $\boldsymbol{w_r(i,j) = w_l(i,15-j)}$, respectively, where $\boldsymbol{i,j \in \{0\ldots15\}}$. Obviously, $\boldsymbol{w_t(i,j) + w_b(i,j) + w_l(i,j) + w_r(i,j) = 1}$, which is the necessary condition for overlapped motion compensation. The motion vectors are differentially coded. The prediction of the motion vector for the current macroblock is obtained by taking the median of the motion vectors of the left, the top and the top-right adjacent macroblocks. The difference between the current motion vector and the predicted motion vector is entropy coded.

In our experiments, the GOP size is 100 or 150 frames with the first frame of a GOP

being intra-coded. To maintain the video quality of a GOP, the intra-coded frames need to be encoded with relatively more bits. We encode an intra-coded frame using 6 to 10 times the number of bits used for each predictively coded frame. In our experiments, no bidirectionally predictive-coded frames (B frames) are used. However, the nature of our algorithm does not preclude the use of B frames.

The embedded bit stream is arranged as follows. The necessary header information, such as the resolution of the sequence and the number of levels of the wavelet transform, is embedded at the beginning of the sequence. In each GOP, the I-frame is coded first using our rate scalable coder. For each P-frame, the motion vectors are differentially coded first. The PEF is then compressed using our rate scalable algorithm. When decoding, after sending the bits of each frame, an end-of-frame (EOF) symbol is transmitted. The decoder can then decode the sequence without prior knowledge of the data rate. Therefore the data rate can be changed dynamically in the process of decoding.

5. Experimental Results and Discussion

Throughout this paper we use the term "visual quality" of a video sequence (or an image), which is the fidelity, or the closeness of the decoded and the original video sequence (or image) when perceived by a viewer. We believe that there does not exist an easily computable metric that will accurately predict how a human observer will perceive a decompressed video sequence. In this paper we will use the peak signal-to-noise ratio (PSNR) based on mean-square error as our "quality" measure. We feel this measure, while unsatisfactory, does track "quality" in some sense. PSNR of the color component $\boldsymbol{X}$, $\boldsymbol{X} \in \{\boldsymbol{Y}, \boldsymbol{U}, \boldsymbol{V}\}$, is obtained by:

$$\boldsymbol{SNR_X} = 10 \log \frac{\boldsymbol{255^2}}{\boldsymbol{mse(X)}},$$

where $\boldsymbol{mse(X)}$ is the mean square error of $\boldsymbol{X}$. When necessary, the overall or combined PSNR is obtained by:

$$\boldsymbol{SNR = 10 \log \frac{255^2}{(mse(Y) + mse(U) + mse(V))/3}.}$$

The effectiveness of using AMC is shown in Figure 9. From the figure we can see that the non-AMC algorithm works better at the highest data rate, to which the encoder feedback loop is locked. However, for any other data rates, the PSNR performance of the non-AMC algorithm declines very rapidly while the error propagation is eliminated in the AMC algorithm. Data rate scalability can be achieved and video quality can be kept relatively constant even at a low data rate with AMC. One should note that the AMC scheme can be incorporated into any motion compensated rate scalable algorithm, no matter what kind of transform is used for the encoding of the I frames and PEFs.

In our experiment, two types of video sequences are used. One type is a CIF (352x240) sequence with 30 frames per second. The other is a QCIF (176x144) sequence with 10 frames per second or 15 frames per second [1].

The CIF sequences are decompressed using *SAMCoW* at data rates of 1 megabits per second (Mbps), 1.5 Mbps, 2 Mbps, 4 Mbps and 6 Mbps. A representative frame decoded at the above rates is shown in Figure 10. At 6 Mbps, the distortion is imperceptible. The decoded video has an acceptable quality when the data rate is 1 Mbps. We used Taubman and Zakhor's algorithm [15] and MPEG-1 to encode/decode the same sequences at the above data rates [2]. Since MPEG-1 is not scalable, the sequences were specifically compressed and decompressed at each of the above data rates. The overall PSNRs of each frame in a GOP are shown in Figures 11 and 12. The computational rate-distortion in terms of average PSNR

[1] The original sequences along with the decoded sequences using *SAMCoW* are available at `ftp://skynet.ecn.purdue.edu/pub/dist/delp/samcow`.

[2] Taubman and Zakhor's software was obtained from the authors.

over a GOP is shown in Table 1. The data indicates that *SAMCoW* has very comparable performance to the other methods tested. Comparison of a decoded image quality using *SAMCoW*, Taubman and Zakhor's algorithm and MPEG-1 is shown in Figure 13. We can see that *SAMCoW* out performs Taubman and Zakhor's algorithm, visually and in terms of PSNR. Even though *SAMCoW* does not perform as well as MPEG-1 in terms of PSNR, subjective experiments have shown that our algorithm produces decoded video with comparable visual quality as MPEG-1 at every tested data rate.

The QCIF sequences are compressed and decompressed using *SAMCoW* at data rates of 20 kilobits per second (Kbps), 32 Kbps, 64 Kbps, 128 Kbps, and 256 Kbps. The same set of sequences are compressed using the H.263 algorithm at the above data rates [3]. Decoded images using *SAMCoW* at different data rates, along with that using H.263, are shown in Figure 14. The overall PSNRs of each frame in a GOP are shown in Figures 15 and 16. The computational rate-distortion in terms of average PSNR over a GOP is shown in Tables 2 and 3. Our subjective experiments have shown that at data rates greater than 32 Kbps *SAMCoW* performs similar to H.263. Below 32 Kbps when sequences with high motion are used, such as the *Foreman* sequence, our algorithm is visually inferior to H.263. This is partially due to the fact that the algorithm cannot treat "active" and "quiet" regions differently, besides using the zerotree coding. At low data rates a large proportion of the wavelet coefficients are quantized to zero and, hence, a large number of the bits are used to code zerotree roots, which does not contribute to distortion reduction. On the contrary, H.263, using a block based transform, is able to selectively allocate bits to different regions with different types of activity. It should be emphasized that the scalable nature of *SAMCoW* makes it very attractive in many low bit rate applications, e.g. streaming video on the Internet. Furthermore, the decoding data rate can be dynamically changed.

[3] The H.263 software was obtained from `ftp://bonde.nta.no/pub/tmn/software`.

6. Summary

In this paper, we proposed a hybrid video compression algorithm, ***SAMCoW***, that provides continuous rate scalability. The novelty of our algorithm includes the following. First, an adaptive motion compensation scheme is used, which keeps the reference frames used in motion prediction at both the encoder and decoder identical at any data rate. Thus error propagation can be eliminated, even at a low data rate. Second, we introduced a spatial-orientation tree in our modified zerotree algorithm that uses not only the frequency bands but also the color channels to scan the wavelet coefficients. The interdependence between different color components in LC spaces is exploited. Our experimental results show that *SAMCoW* out performs Taubman and Zakhor's 3-D subband rate scalable algorithm. In addition, our algorithm has a wide range of rate scalability. For medium to high data rate applications, it has comparable performance to the non-scalable MPEG-1 and MPEG-2 algorithms. Furthermore, it can be used for low bit rate applications with a performance similar to H.263. The nature of *SAMCoW* allows the decoding data rate to be dynamically changed. Therefore, the algorithm is appealing for many network oriented applications because it is able to adapt to the network loading.

7. REFERENCES

[1] ISO/IEC 13818-2, *Generic coding of moving pictures and associated audio information.* MPEG (Moving Pictures Expert Group), International Organization for Standardisation, 1994. (MPEG2 Video).

[2] B. G. Haskell, A. Puri, and A. N. Netravali, *Digital video: an introduction to MPEG-2.* New York: International Thomson Publishing, 1997.

[3] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, pp. 3445–3462, December 1993.

[4] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31, no. 4, pp. 532–540, April 1983.

[5] H. M. Dreizen, "Content-driven progressive transmission of gray level images," *IEEE Transactions on Communications*, vol. COM-35, pp. 289–296, March 1987.

[6] Y. Huang, H. M. Dreizen, and H. P. Galatsanos, "Prioritized DCT for compression and progressive transmission of images," *IEEE Transactions on Image Processing*, vol. 1, no. 4, pp. 477–487, October 1992.

[7] K. H. Tzou, "Progressive image transmission: a review and comparison," *Optical Engineering*, vol. 26, pp. 581–589, 1987.

[8] K. R. Rao and P. Yip, *Discrete cosine transform: algorithms, advantages, and applications*. Academic Press, 1990.

[9] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, June 1996.

[10] B. Yazici, M. L. Comer, R. L. Kashyap, and E. J. Delp, "A tree structured Bayesian scalar quantizer for wavelet based image compression," *Proceedings of the 1994 IEEE International Conference on Image Processing*, vol. III, November 13–16 1994, Austin, Texas, pp. 339–342.

[11] C. S. Barreto and G. Mendoncca, "Enhanced zerotree wavelet transform image coding exploiting similarities inside subbands," *Proceedings of the IEEE International Conference on Image Processing*, vol. II, September 16–19 1996, Lausanne, Switzerland, pp. 549–552.

[12] D. A. Huffman, "A method for the construction of minimim redundancy codes," *Proceedings of the IRE*, vol. 40, pp. 1098–1101, September 1952.

[13] I. Witten, R. Neal, and J. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, pp. 520–540, 1987.

[14] M. Nelson and J. Gailly, *The data compression book*. M&T Books, 1996.

[15] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 572–588, September 1994.

[16] C. I. Podilchuck, N. S. Jayant, and N. Farvardin, "Three dimensional subband coding of video," *IEEE Transaction on Image Processing*, vol. 4, no. 2, pp. 125–138, February 1995.

[17] Y. Chen and W. A. Pearlamn, "Three dimensional subband coding of video using the zero-tree method," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, March 28–30 1996, San Jose, California.

[18] ISO/IEC 11172-2, *Coding of moving pictures and associated audio for digital storage media at up ot about 1.5 Mbit/s*. MPEG (Moving Pictures Expert Group), International Organization for Standardisation, 1993. (MPEG1 Video).

[19] ITU-T, *ITU-T Recommendation H.263: Video coding for low bitrate communication.* The International Telecommunication Union, 1996.

[20] M. Ohta and S. Nogaki, "Hybrid picture coding with wavelet transform and overlapped motion-compensated interframe prediction coding," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3416–3424, December 1993.

[21] S. A. Martucci, I. Sodagar, T. Chiang, and Y.-Q. Zhang, "A zerotree wavelet video coder," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 109–118, February 1997.

[22] Q. Wang and M. Ghanbari, "Scalable coding of very high resolution video using the virtual zerotree," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 5, pp. 719–727, October 1997.

[23] B. J. Kim and W. A. Pearlamn, "Low-delay embedded 3-D wavelet color video coding with SPIHT," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, January 28–30 1998, San Jose, California.

[24] B. J. Kim and W. A. Pearlamn, "An embedded wavelet video coder using three dimensional set partitioning in hierarchical trees(SPIHT)," *Proceedings of IEEE Data Compression Conference*, March 25–27 1997, Snowbird, Utah.

[25] M. L. Comer, K. Shen, and E. J. Delp, "Rate-scalable video coding using a zerotree wavelet approach," *Proceedings of the Ninth Image and Multidimensioanl Digital Signal Processing Workshop*, March 3-6 1996, Belize City, Belize, pp. 162–163.

[26] K. Shen and E. J. Delp, "A control scheme for a data rate scalable video codec," *Proceedings of the IEEE International Conference on Image Processing*, September 16–19 1996, Lausanne, Switzerland, pp. Vol II, pp 69–72.

[27] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, pp. 417–441 and 498–520, 1933.

[28] C. B. Rubinstein and J. O. Limb, "Statistical dependence between components of a differentially quantized color signal," *IEEE Transactions on Communications Technology*, vol. COM-20, pp. 890–899, October 1972.

[29] P. Pirsch and L. Stenger, "Statistical analysis and coding of color video signals," *Acta Electronica*, vol. 19, no. 4, pp. 277–287, 1976.

[30] A. N. Netravali and C. B. Rubinstein, "Luminance adaptive coding of chrominance signals," *IEEE Transactions on Communications*, vol. COM-27, no. 4, pp. 703–710, April 1979.

[31] J. O. Limb and C. B. Rubinstein, "Plateau coding of the chrominance component of color picture signals," *IEEE Transactions on Communications*, vol. COM-22, no. 3, pp. 812–820, June 1974.

[32] K. Shen and E. J. Delp, "Color image compression using an embedded rate scalable approach," *Proceedings of IEEE International Conference on Image Processing*, October 26–29 1997, Santa Barbara, California.

[33] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Transactions on Image Processing*, vol. 1, no. 2, pp. 205–220, April 1992.

[34] H. Watanabe and S. Singhal, "Windowed motion compensation," *SPIE Conference on Visual Communications and Image Processing*, November 1991, Boston, Massachusetts, pp. 582–589.

sequence		*football*				*flowergarden*			
components		All	Y	U	V	All	Y	U	V
1 Mbps	*SAMCoW*	27.1	25.8	26.6	29.9	24.3	22.4	24.4	27.3
	Taubman	25.2	21.5	28.8	32.2	21.1	17.3	24.7	28.9
	MPEG-1	28.8	25.7	31.2	33.3	25.7	22.5	27.2	29.9
1.5 Mbps	*SAMCoW*	28.1	27.1	27.7	30.1	25.4	24.1	25.2	27.8
	Taubman	26.3	22.6	29.6	32.8	21.6	17.9	25.0	29.4
	MPEG-1	30.2	27.3	32.2	33.9	27.0	24.4	28.5	30.4
2 Mbps	*SAMCoW*	28.8	28.2	28.4	30.4	26.7	25.7	26.6	28.4
	Taubman	26.7	23.1	30.1	33.0	22.3	18.7	25.5	29.6
	MPEG-1	31.2	28.5	33.0	34.5	28.8	26.5	29.9	31.3
4 Mbps	*SAMCoW*	30.9	30.9	30.4	31.8	28.7	28.8	28.2	29.3
	Taubman	28.9	25.0	31.5	33.9	24.1	20.6	26.7	30.7
	MPEG-1	34.2	32.2	35.3	36.1	32.6	30.9	33.6	34.1
6 Mbps	*SAMCoW*	34.9	35.3	34.6	35.1	33.8	34.8	33.4	33.4
	Taubman	29.8	26.5	32.6	34.6	25.6	22.3	28.1	31.0
	MPEG-1	36.8	35.2	37.8	38.2	35.4	33.8	36.4	36.5

Table 1: PSNR of CIF sequences, average over a GOP. (30 frames per second)

sequence		*akiyo*				*foreman*			
components		All	Y	U	V	All	Y	U	V
20 Kbps	*SAMCoW*	32.8	31.7	32.3	35.0	28.6	26.1	30.4	31.4
	H.263	37.5	35.6	38.1	40.1	30.3	27.2	33.5	33.6
32 Kbps	*SAMCoW*	34.6	33.3	34.4	36.8	30.1	27.3	33.4	32.2
	H.263	39.1	37.4	39.7	41.2	31.1	28.1	34.0	34.2
64 Kbps	*SAMCoW*	38.3	37.2	38.7	39.4	31.6	29.4	33.6	33.4
	H.263	43.1	40.8	44.8	45.1	33.9	31.1	36.2	36.8
128 Kbps	*SAMCoW*	43.2	41.9	43.7	44.3	33.3	31.5	34.4	34.7
	H.263	46.3	44.6	47.4	47.7	36.6	34.0	38.3	39.3
256 Kbps	*SAMCoW*	47.5	46.8	47.9	48.1	35.2	34.1	35.5	36.4
	H.263	49.1	48.4	49.3	49.7	38.4	36.1	39.8	41.1

Table 2: PSNR of QCIF sequences, averaged over a GOP. (15 frames per second)

sequence		*akiyo*				*foreman*			
components		All	Y	U	V	All	Y	U	V
20 Kbps	*SAMCoW*	34.0	32.5	33.9	36.3	29.5	26.6	32.7	31.9
	H.263	39.3	36.6	41.4	42.4	30.4	27.1	33.9	34.2
32 Kbps	*SAMCoW*	36.3	34.9	36.5	38.0	30.5	28.0	33.2	32.3
	H.263	40.3	38.7	40.8	42.1	32.3	29.3	35.2	35.5
64 Kbps	*SAMCoW*	40.3	38.9	40.8	41.5	32.0	30.0	33.6	33.7
	H.263	43.4	42.1	44.0	44.8	34.8	32.0	37.0	37.7
128 Kbps	*SAMCoW*	44.9	43.9	45.6	45.7	33.9	32.4	34.8	35.3
	H.263	46.4	44.7	47.5	47.6	37.2	34.7	38.7	39.9
256 Kbps	*SAMCoW*	48.8	48.6	48.7	49.0	36.1	35.3	36.0	37.3
	H.263	49.3	48.7	49.5	49.9	39.3	37.1	40.5	41.9

Table 3: PSNR of QCIF sequences, averaged over a GOP. (10 frames per second)

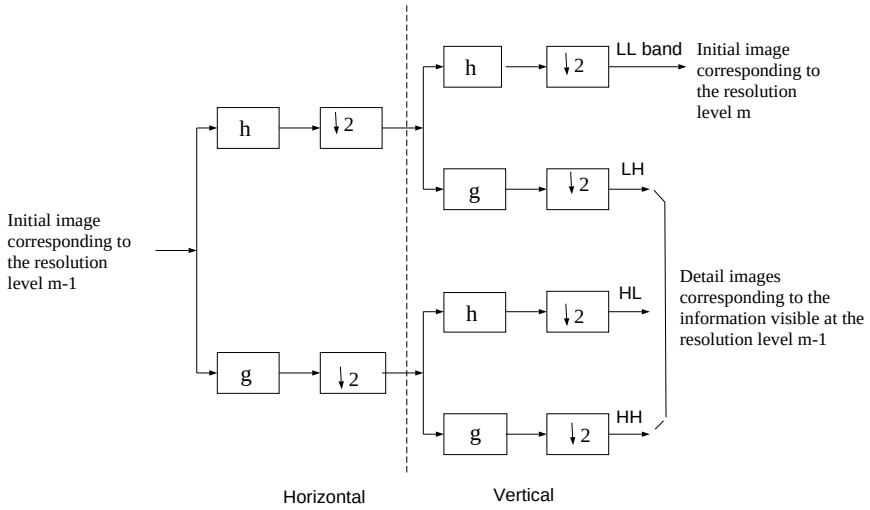


Figure 1: One level of the wavelet transform.

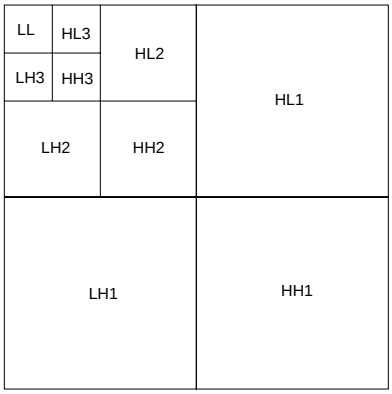


Figure 2: Pyramid structure of a wavelet decomposed image. Three levels of the wavelet decomposition are shown.

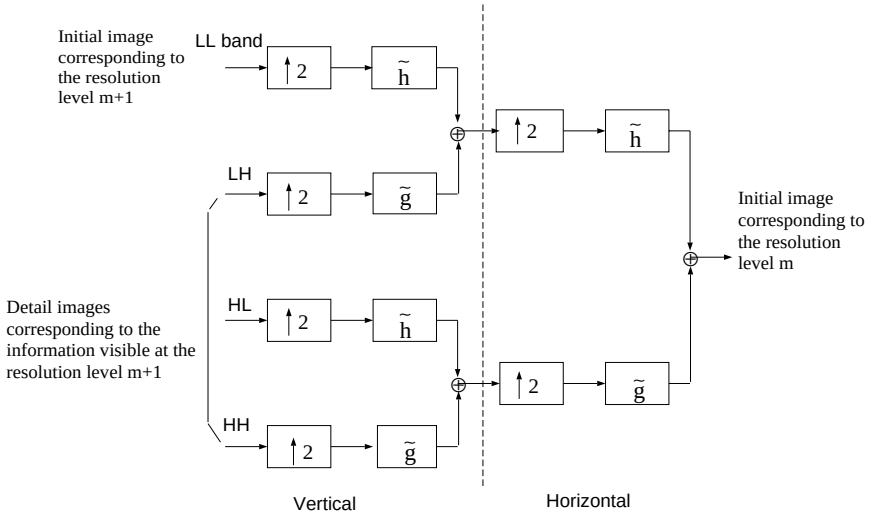
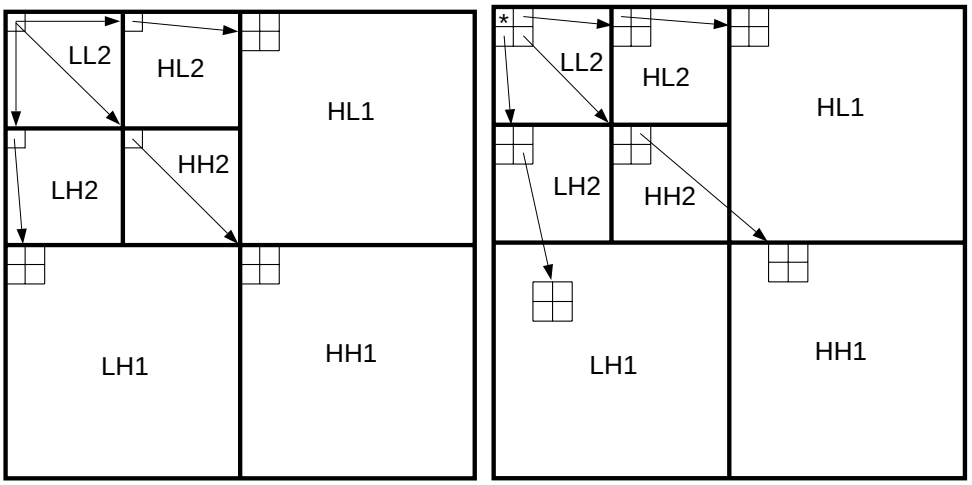


Figure 3: One level of the inverse wavelet transform.



(a) (b)

Figure 4: Diagrams of the parent-descendent relationships in the spatial-orientation trees. (a) Shapiro's algorithm. Notice that the pixel in the LL band has 3 children. Other pixels, except for those in the highest frequency bands, have 4 children. (b) Said and Pearlman's algorithm. One pixel in the LL bands (noted with "*") does not have a child. Other pixels, except for those in the highest frequency bands, have 4 children.

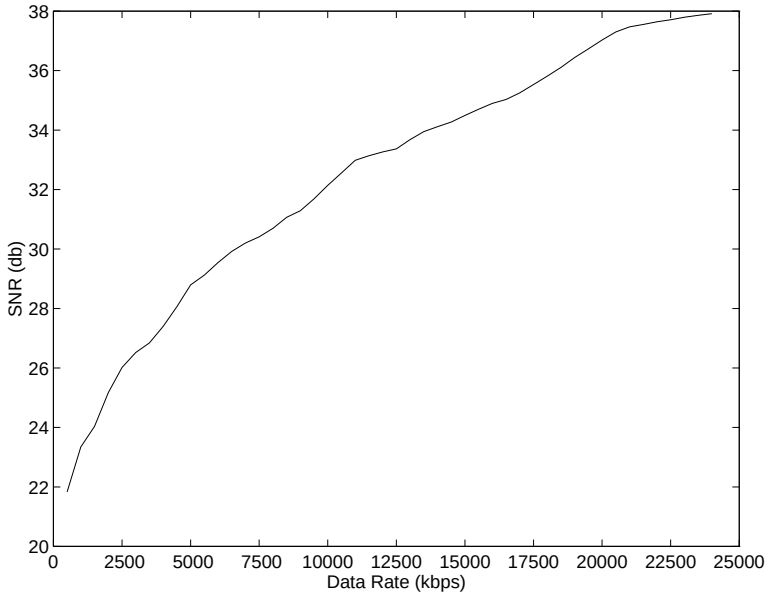


Figure 5: Average PSNR of EZW encoded *footballl* sequence (I frame only) at different data rates. (30 frames per second)

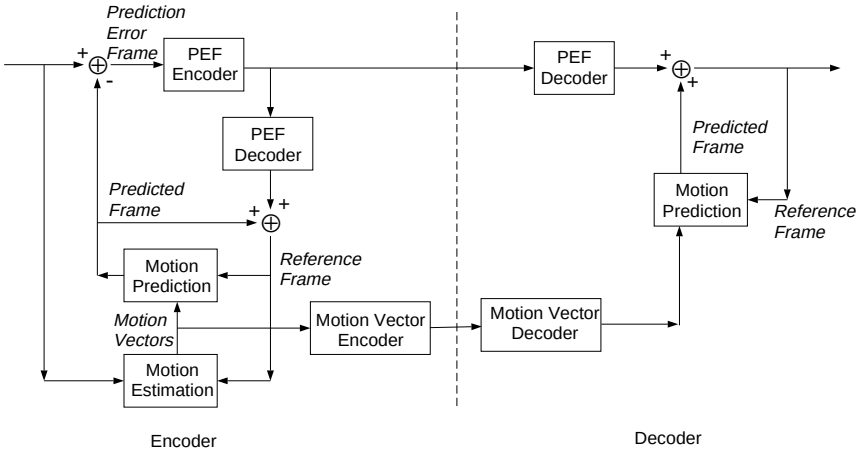


Figure 6: Block diagram of a generalized hybrid video codec for predictively coded frames. Feedback loop is used in the encoder. Adaptive motion compensation is not used.

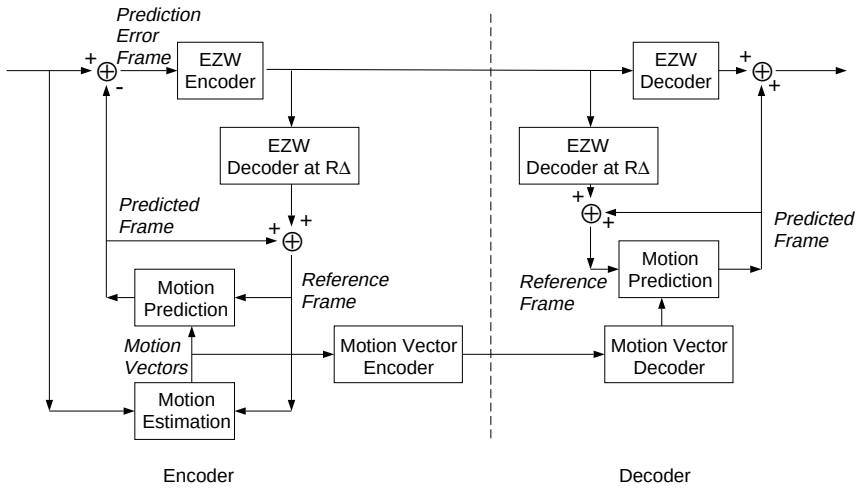


Figure 7: Block diagram of the proposed codec for predictively coded frames. Adaptive motion compensation is used.

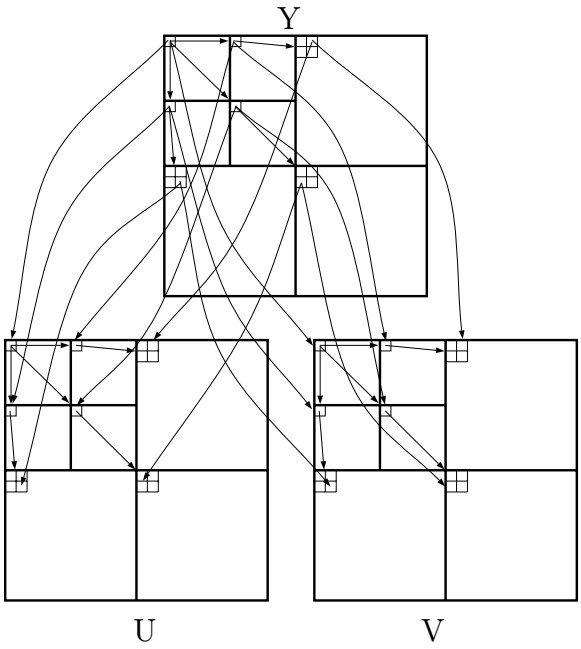


Figure 8: Diagram of the parent-descendent relationships in *SAMCoW* algorithm. This tree is developed on the basis of the tree structure in Shapiro's algorithm. The YUV color space is used.

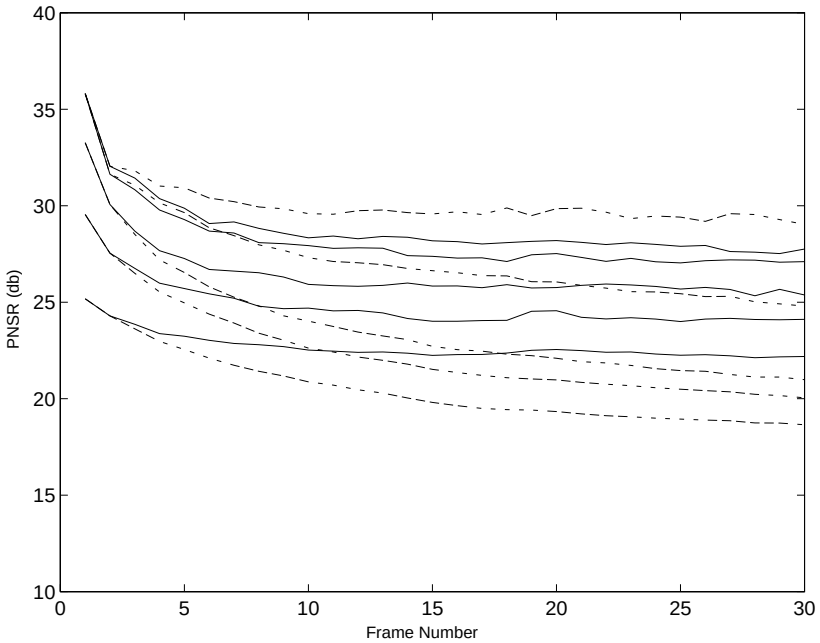
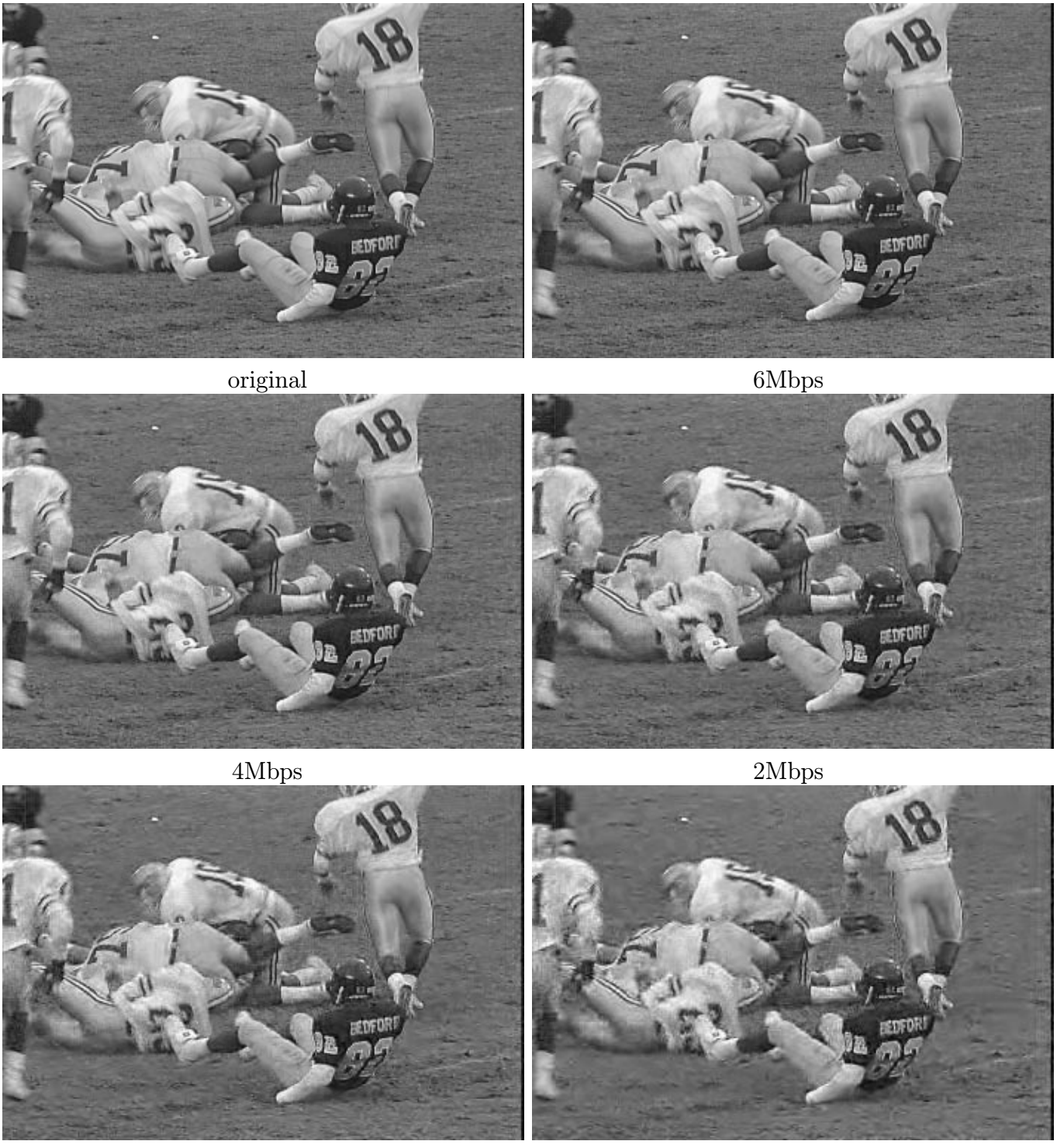


Figure 9: PSNR of each frame within a GOP of the *football* sequence at different data rates. Solid lines: AMC; dashed lines: non-AMC; Data rates in kbps(from top to bottom): 6000, 5000, 3000, 1500, 500.

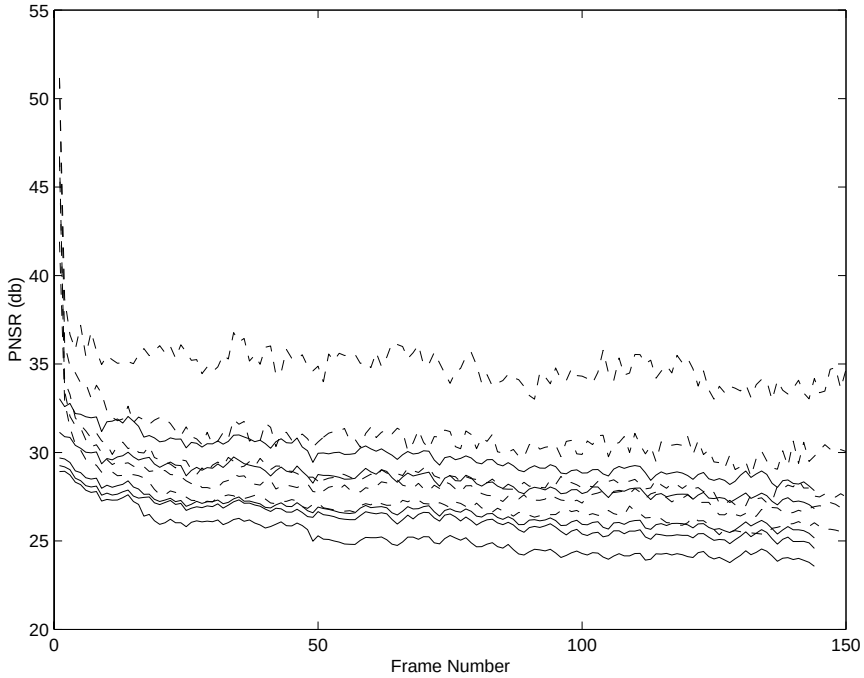


original 6Mbps

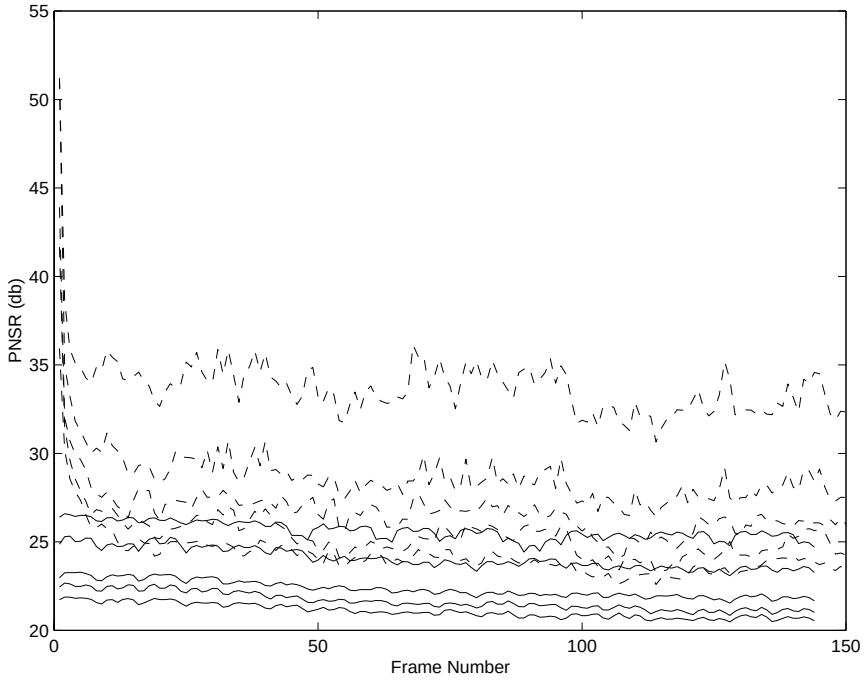
4Mbps 2Mbps

1.5Mbps 1Mbps

Figure 10: Frame 35 (P frame) of the *football* sequence, decoded at different data rates using *SAMCoW* (CIF, 30 frames per second).

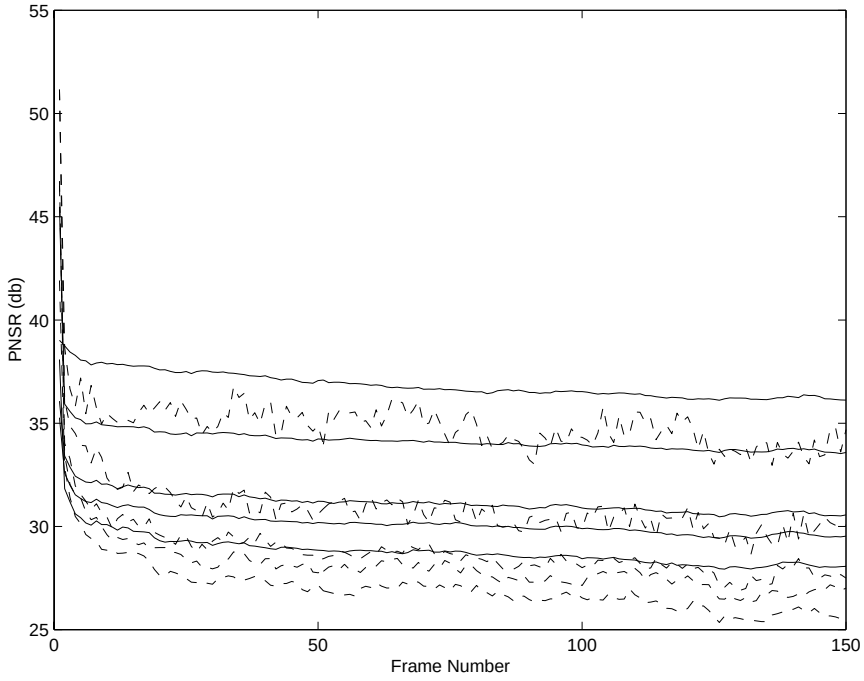


a. *football*

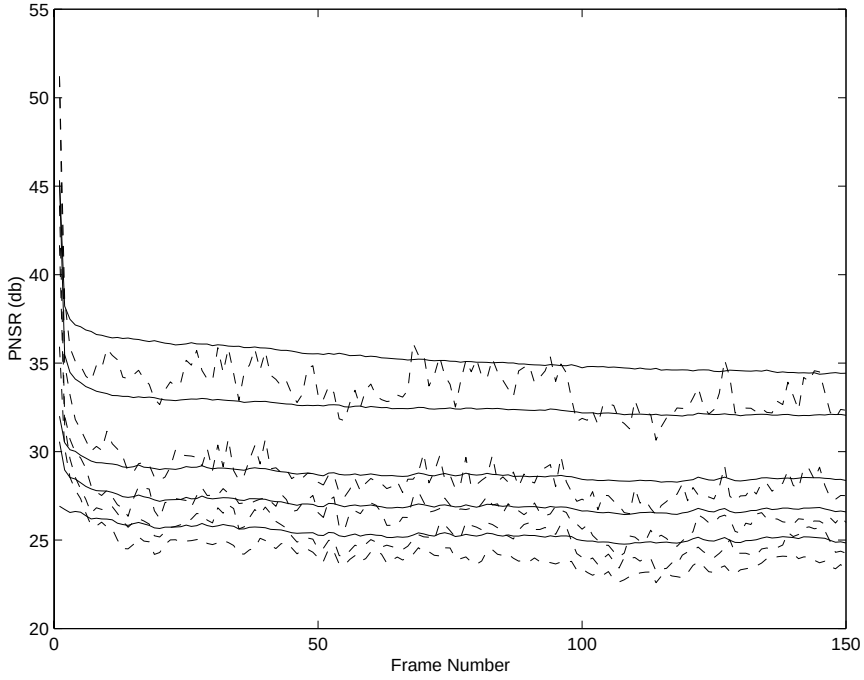


b. *flowergarden*

Figure 11: Comparison of the performance of *SAMCoW* and Taubman and Zakhor's algorithm. Dashed lines: *SAMCoW*; solid lines: Taubman and Zakhor's algorithm. The sequences are decoded at 6 Mbps, 4 Mbps, 2 Mbps, 1.5 Mbps and 1 Mbps, which respectively correspond to the lines from top to bottom.

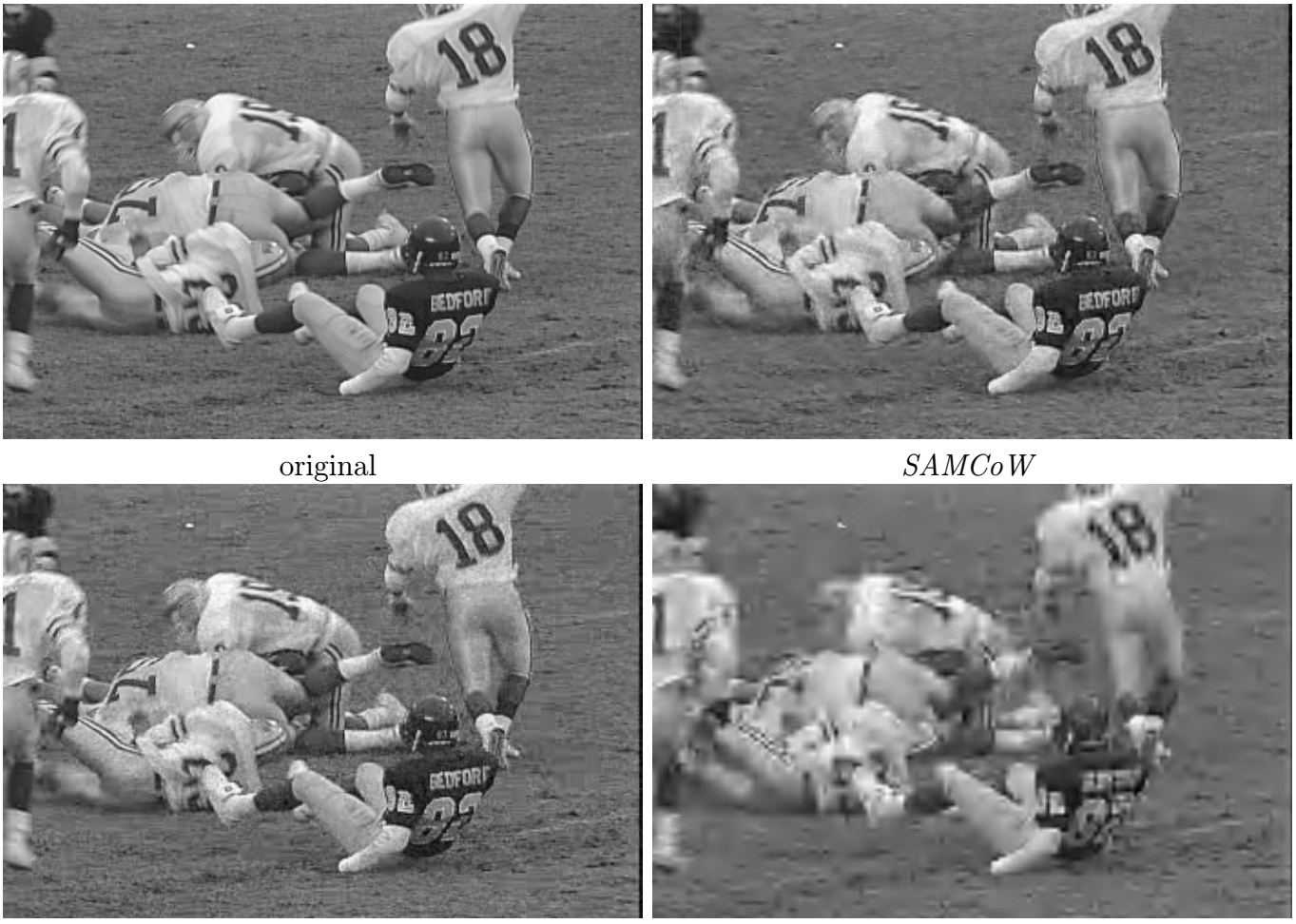


a. *football*



b. *flowergarden*

Figure 12: Comparison of the performance of *SAMCoW* and MPEG-1. Dashed lines: *SAMCoW*; solid lines: MPEG-1. The sequences are decoded at 6 Mbps, 4 Mbps, 2 Mbps, 1.5 Mbps and 1 Mbps, which respectively correspond to the lines from top to bottom.

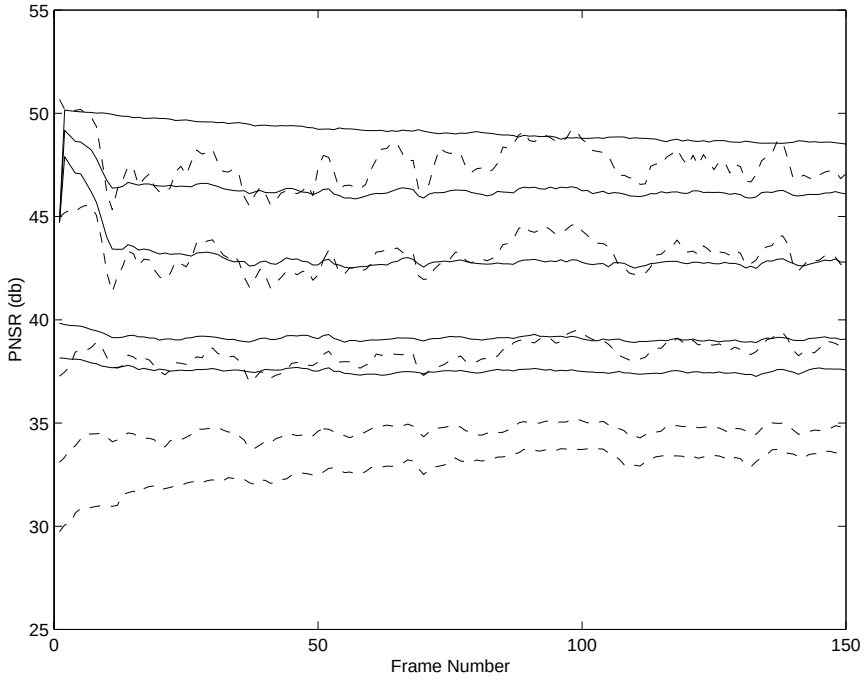


original | *SAMCoW*

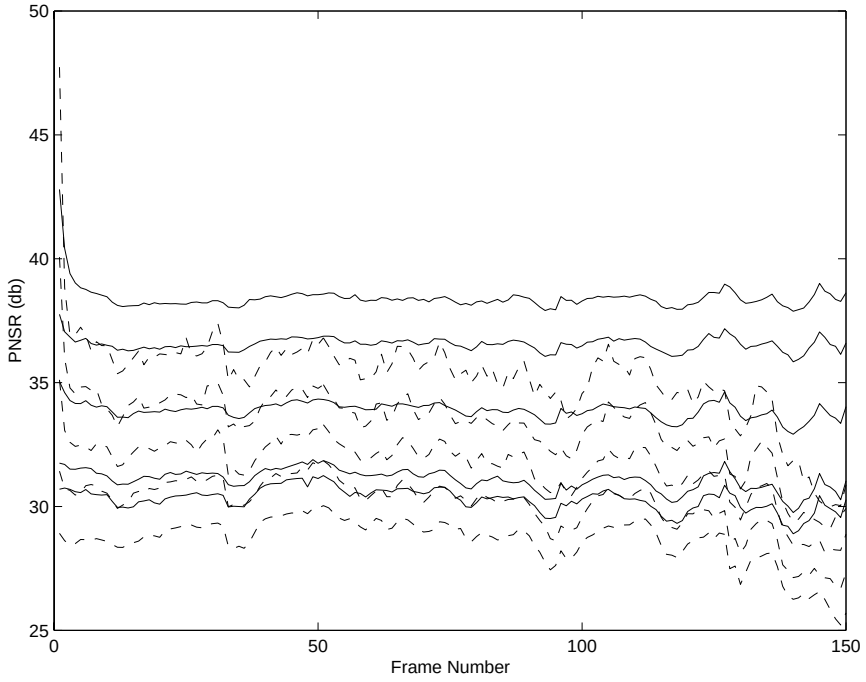
MPEG-1 | Taubman and Zakhor

Figure 13: Frame 35 (P frame) of the *football* sequence (CIF, 30 frames per second). The data rate is 1.5 Mbps

Figure 14: Frame 78 (P frame) of the *Akiyo* sequence and frame 35 (P frame) of the *Foreman* sequence, decoded at different data rates (QCIF, 10 frames per second).

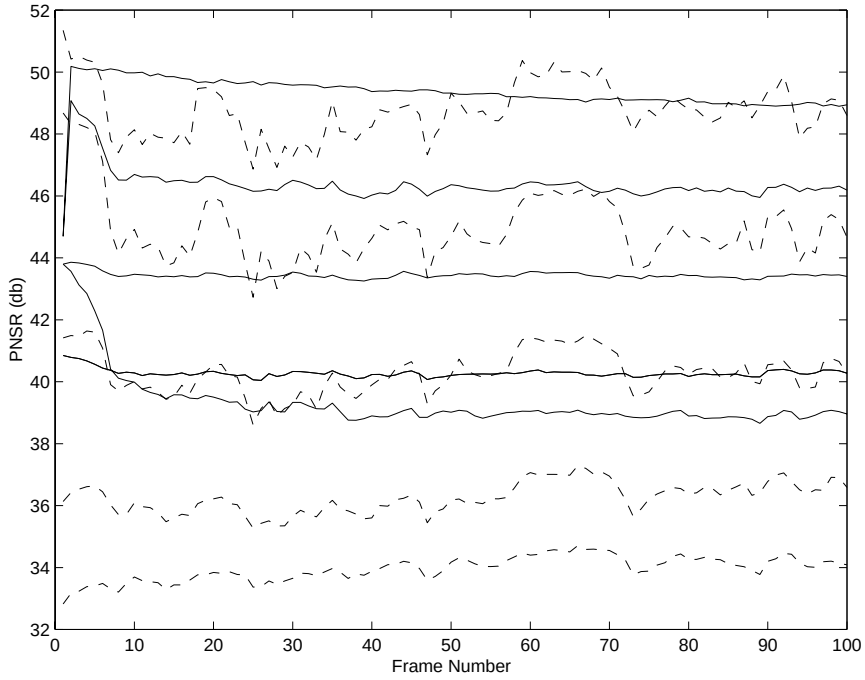


a. *akiyo*

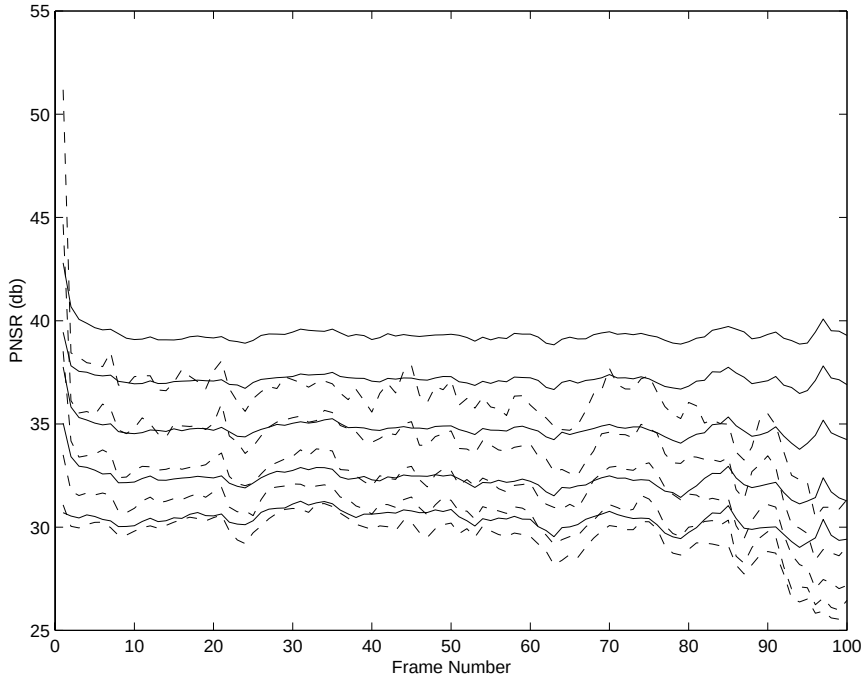


b. *foreman*

Figure 15: Comparison of the performance of *SAMCoW* and H.263. (QCIF at 15 frames per second) Dashed lines: *SAMCoW*; solid lines: H.263. The sequences are decoded at 256 kbps, 128 kbps, 64 kbps, 32 kbps and 20 kbps, which respectively correspond to the lines from top to bottom.



a. *akiyo*



b. *foreman*

Figure 16: Comparison of the performance of *SAMCoW* and H.263. (QCIF at 10 frames per second) Dashed lines: *SAMCoW*; solid lines: H.263. The sequences are decoded at 256 kbps, 128 kbps, 64 kbps, 32 kbps and 20 kbps, which respectively correspond to the lines from top to bottom.