Rate Control for Multisequence Video Streaming

Joseph C. Dagher, Ali Bilgin and Michael W. Marcellin

Dept. of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85721

ABSTRACT

Streaming media over heterogeneous lossy networks and time-varying communication channels is an active area of research. Several video coders that operate under the varying constraints of such environments have been proposed recently. Scalability has become a very desirable feature in these video coders. In this paper, we make use of a leaky-bucket rate allocation method (DBRC) that provides constant quality video under buffer constraints, and extend it in two advantageous directions. First, we present a rate control mechanism for 3D wavelet video coding using DBRC. Second, we enhance the DBRC so that it can be utilized when multiple sequences are multiplexed over a single communications channel. The goal is to allocate the capacity of the channel between sequences to achieve constant quality across all sequences.

Keywords: Video streaming, JPEG2000, Rate control, Bit allocation, Buffering, Leaky-bucket, Scalable, Multisequence, 3D-Wavelet transform.

1. INTRODUCTION

With the increasing importance of heterogeneous networks and time-varying communication channels, such as packet-switched networks and wireless communications, scalability has become a highly desirable feature in both image and video coders. Besides other advantages, scalable video coders produce excellent results when they are coupled with efficient rate control algorithms. A single scalable bitstream can provide precise rate control for constant bitrate (CBR) traffic and accurate quality control for variable bitrate (VBR) traffic. Some video coders offer scalability on a coarse level, such as the MPEG-2 and H.263 coders that produce layered bitstreams. Others offer fine scalability where the bitstreams can be decoded at any bitrate up to and including a maximum. Various attempts that make use of both fine and coarse scalability for efficient rate allocation have appeared in the literature. In [1,2] the authors utilize scalable codecs to achieve constant quality video. Scalable codecs were also used in [3–5] to adaptively accommodate changing network conditions. Recently, two leaky-bucket rate allocation methods were proposed in [6]. These methods provide constant quality video under buffer constraints. Although the efficiency of the methods was presented using the finely scalable Motion JPEG2000 codestreams, these methods can be used with other finely scalable codecs.

In this paper, we further extend the work of [6]. First, we present a rate control mechanism for 3D wavelet video coding. 3D wavelet video coding has attracted considerable attention recently [7–11]. Traditionally, video compression algorithms rely on motion compensation and efficient 2D compression of the motion compensated residuals. Recently, it has been shown that 3D wavelet video coding schemes can achieve comparable performance without the complexity of motion compensation [7,8,10]. If motion compensation is utilized in 3D wavelet video coding schemes, they can outperform 2D schemes [9,11]. Furthermore, 3D wavelet video coding schemes generate finely scalable bitstreams that offer additional advantages. We apply the rate control mechanism of [6] to such bitstreams. We discuss how the wavelet transform across the time dimension can be performed to enable precise rate control without introducing substantial latency.

Second, we extend the work in [6], so that it can be utilized when multiple sequences are multiplexed over a single communications channel. Several independent video sequences are compressed and sent over a single channel sharing its capacity. The goal is to allocate the capacity of the channel between sequences to achieve constant quality across all sequences. Our results indicate that substantial decrease in variance of the quality of individual frames can be achieved using the proposed method. In our experiments, we have utilized the JPEG2000 and Motion JPEG2000 codecs since they provide finely scalable bitstreams together with state-of-the-art performance. However, the proposed methods can easily be used with other finely scalable codecs as well.

Further author information: (Send correspondence to J.C.D.) J.C.D.: joseph@ece.arizona.edu, A.B.: bilgin@ieee.org, M.W.M.: marcellin@ece.arizona.edu

This paper is organized as follows. An overview of JPEG2000 and Motion JPEG2000 is presented in Section 2. Section 3 reviews the two rate control algorithms proposed for efficient video streaming in [6]. In Section 4, we present the proposed rate controller for 3D wavelet video coders. Then, in Section 5 the algorithm for multisequence video streaming is presented. Section 6 presents concluding statements.

2. OVERVIEW OF JPEG2000 AND MOTION JPEG2000

JPEG2000 is the latest ISO/IEC image compression standard. Here, we will provide a high level description of the JPEG2000 algorithm to assist the reader in comprehending the remainder of this paper. For a more thorough description, the interested reader is referred to [12–14].

A simplified block diagram of a JPEG2000 encoder is illustrated in Figure 1. The input image is first passed through an optional component transform to achieve decorrelation across color components. The resultant components are wavelet transformed and quantized. Each subband is then divided into codeblocks. Codeblocks are compressed independently using a bitplane coder. The bitplane coder makes three passes over each bitplane of a codeblock. Each of these passes are referred to as coding passes or subbitplanes. Thus, an embedded bitstream is generated for each codeblock. The JPEG2000 encoder computes and stores the rate-distortion information corresponding to each subbitplane of every block.

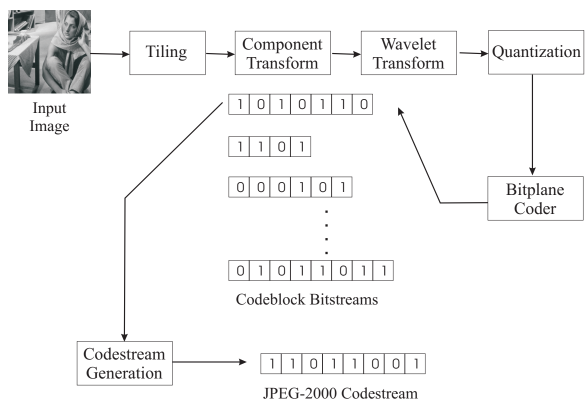

Figure 1. Block diagram of a JPEG2000 encoder.

The creation of a JPEG2000 codestream involves the inclusion of a different number of coding passes from each individual codeblock bitstream. JPEG2000 offers tremendous flexibility in this regard. The decision on how many coding passes of a particular codeblock bitstream should be included can be based on any desired criteria. For example, optimum rate-distortion performance at a given target rate is achieved when the coding passes with greatest distortion-rate slopes are included.

JPEG2000 includes the following features:

- Superior compression performance: JPEG2000 provides excellent compression performance compared to previous standards; especially at low rates.
- Multi-component image compression: JPEG2000 can handle binary and continuous tone multi -component images.
- Lossless and lossy compression can be obtained from one bitstream in the course of progressive decoding.
- Progressive transmission by pixel accuracy and resolution that allows the reconstruction of images at any rate and various resolution levels.
- Random code-stream access and processing to allow operations such as compressed domain cropping, rotation, translation, filtering, feature extraction, scaling, etc.
- Region-Of-Interest (ROI) encoding/decoding.
- Robustness to bit-errors.

The JPEG committee has decided to extend the JPEG2000 standardization effort to video coding. The result of this extension is referred to as Motion JPEG2000 (MJP2). MJP2 is essentially a file format for wrapping compressed frames generated by the JPEG2000 image coding engine [15–17]. It is intended to generate a highly scalable bitstream, which can be easily edited. Thus, MJP2 does not include motion compensation. Each frame is individually compressed and stored. The scope of MJP2 encompasses video compression for applications including Digital Still Camera (DSC) and Camcorder, remote surveillance systems, digital video recording systems, and video capture cards. Preliminary results indicate that substantial performance gain and functionality can be achieved over existing Motion-JPEG methods [17].

3. THE LEAKY-BUCKET ALGORITHM

In this section, we provide an overview of the rate control algorithms presented in [6]. Our goal is to devise an algorithm to achieve constant quality video under buffer and rate constraints. Let N denote the number of frames to be encoded and let R denote the average rate per pixel, per frame. Thus the total bit budget for encoding all N frames is NR. Let D_i and $R_i, i \in \{1, 2, \ldots, N\}$, denote the distortion and rate associated with the ith frame, respectively. Let B denote the size of the buffer that is used to hold the compressed image sequence.

For a given buffer size B, the problem is to achieve minimum average distortion under the constraint that the total bit budget is not exceeded. In other words, for a fixed B, we would like to select R_i according to

$$\underset{R_i}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} D_i \tag{1}$$

subject to the constraint that

$$\sum_{i=1}^{N} R_i = NR. \tag{2}$$

The solution to this problem is given by [18]

$$R_i = R + \frac{1}{2} \log_2 \frac{\sigma_i^2}{G}. \tag{3}$$

when the corresponding distortions are modeled by

$$D_i = G\epsilon^2 2^{-2R}, \tag{4}$$

where ϵ^2 is a constant that takes into account the performance of practical quantizers and G is the geometric mean of the variances of the frames, $\sigma_i^2, i \in \{1, 2, \ldots, N\}$, given by

$$G = \left[\prod_{i=1}^{N} \sigma_i^2\right]^{\frac{1}{N}}. \tag{5}$$

It can be seen from Equation (4) that D_i is constant, $\forall i \in \{1, 2, \ldots, N\}$. This suggests that for the simple model employed here, minimizing the average distortion should result in individual distortions being equal across all frames. In other words, minimizing average distortion should result in constant quality, as desired.

It is important to point out two extreme cases at this point. The first one is when the buffer size is equal to the size of the entire compressed sequence, i.e. $B = NR$. This will clearly yield the best result, however for large N, buffering the entire compressed sequence may not be feasible due to memory constraints. Furthermore this approach will result in very large latency. The other extreme case is when only a single compressed frame is buffered. This case will provide minimum latency. However, the quality of the decoded sequence will vary widely across frames depending on rate-distortion properties of the sequence.

The algorithm presented in [6] was motivated by the work of [19] which presents a low memory implementation of a JPEG2000 image coder for coding a single frame. That algorithm employs a sliding window wavelet transform to generate wavelet coefficients in an incremental fashion. Each time enough lines of wavelet coefficients are available, they are divided into codeblocks, quantized, and entropy coded. The resulting embedded block bitstreams are subsequently sent to an output (FIFO) buffer. Compressed data are removed from this buffer for transmission at a constant rate. Rate allocation is implicitly performed through the algorithm by which compressed data are added to the buffer.

Whenever such data are to be added to the buffer, there is a possibility that not enough buffer space is available. When this occurs, the coding passes having lowest distortion-rate slopes are discarded. In general, these discarded coding passes come from both the buffer and the newly compressed data that is to be added to the buffer.

Two different rate control (RC) algorithms were presented in [6] to provide constant decoded video quality subject to buffer constraints.

3.1. Single buffer rate controller (SBRC)

Figure 2 shows a basic block diagram of the SBRC algorithm. As shown, each frame is compressed independently using the JPEG2000 coding engine. The compression rate of each frame is somewhat greater than the target rate for the sequence. The resulting compressed bitstream is placed in a buffer awaiting transmission or storage. Then, the data is pulled out of the buffer at a constant rate. When the buffer is (or about to be) full, all bitstreams, including the ones already in the buffer along with the new bitstream to be inserted, are truncated via the embedding property to maintain constant quality across all frames in the buffer. This strategy relies on the highly scalable nature of JPEG2000. The SBRC algorithm uses a single RC buffer to achieve constant quality. The algorithm is described in more detail in Table 3.1.

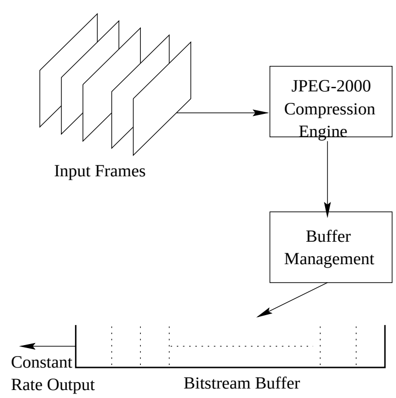

Figure 2. Basic block diagram of the SBRC algorithm.

3.2. Double buffer rate controller (DBRC)

Although the SBRC algorithm performs reasonably well under most conditions, it is possible to improve its performance. To see this, consider the scenario where we have $M-1$ frames already in the RC buffer. Furthermore, assume that the coding passes of those frames have been truncated according to a RD threshold of T_1. Suppose that the next frame to be inserted in the buffer is such that most of its coding passes have RD slopes smaller than T_2, where $T_2 \ll T_1$. As a result, the new RD threshold T_{RD} computed for all M frames will be $T_{RD} < T_1$. However, having permanently truncated the coding passes of the first $M-1$ frames with RD slopes less than T_1, we will be obliged to include the coding passes with RD slopes less than T_2 from the new frame, or allow the buffer to remain at less than full occupancy. In this situation, it is desirable to be able to “reclaim” coding passes (with RD slopes between T_2 and T_1) discarded from other frames in the buffer.

To this end, the DBRC algorithm was introduced in [6] . In DBRC, some of the coding passes that have been eliminated in previous iterations are kept in a secondary buffer of predetermined size. The DBRC algorithm allows these coding passes to be considered again at a later stage. It should be noted that once a frame is released, all of its passes residing in the secondary buffer will be permanently discarded. The DBRC algorithm is described in Table 3.2.

Given the size of the RC buffer B in bytes, determine the number of frames that will fit in the buffer, M, using $M = \frac{BD}{SR}$, where S is the size of one frame in pixels, R is the desired bit rate in bits/pixel, and D is the pixel bit-depth.

Determine an RD threshold T_{RD} such that the coding passes of the first frame with RD slopes $\geq T_{RD}$ will fit into the buffer.

Delete the coding passes of the first frame with RD slopes less than T_{RD}.

for $k = 2$ to $k = N + M$

    if $k \leq N$
        Determine T_{RD} so that coding passes of the frames currently in the buffer and those of the kth frame with slopes $\geq T_{RD}$ will fit in the buffer.

        Delete the coding passes of the frames currently in the buffer and those of the kth frame with RD slopes $< T_{RD}$.

        Insert the qualifying coding passes of the kth frame into the buffer.
    end if

    if $k > M$
        Release SR bits from the head of the buffer to the codestream.
    end if

    Set $k = k + 1$

end for

Table 1. SBRC algorithm.

4. RATE CONTROL FOR 3D WAVELET VIDEO CODING

Traditional video compression algorithms rely on motion compensation and efficient 2D compression of the motion compensated residuals. This 2D compression of the motion compensated residuals is usually achieved through a Discrete Cosine Transform (DCT) based compression scheme. In recent years, the wavelet transform has emerged as an alternative to DCT for image compression applications. It has been shown that 3D wavelet video coding schemes can achieve comparable performance without the complexity of motion compensation [7,8,10] . If motion compensation is utilized in 3D wavelet video coding schemes, they can outperform 2D schemes [9,11] . Furthermore, 3D wavelet video coding schemes generate finely scalable bitstreams that offer additional advantages.

Here, we extend the rate control mechanism of [6] to operate on 3D wavelet coded bitstreams. The basic block diagram of this scheme is illustrated in Figure 3. Here the input frames are wavelet transformed across the temporal direction first. The resulting temporal wavelet coefficient frames are fed into a JPEG2000 compression engine.

4.1. Memory Constrained Temporal Transform

An important consideration in video coding is the amount of latency introduced by the compression scheme. Large latency can be very undesirable. The goal of the RC algorithm presented in this work is to achieve constant quality under latency and memory constraints. To extend the proposed methods to 3D wavelet video coding schemes, we first need to analyze such schemes, paying close attention to their latency and memory requirements. To combat the problem of latency, common 3D wavelet video coding algorithms divide the input sequence into several groups of frames (GOF). The 3D wavelet transform is then applied to each GOF independently. Thus, the amount of latency

Given the size of the primary RC buffer B^p in bytes, determine the number of frames that will fit in the buffer, M, using $M = \frac{B^p D}{SR}$, where S is the size of one frame in pixels, R is the desired bit rate in bits/pixel, and D is the pixel bit-depth.

Determine the primary RD threshold T_{RD}^p such that the coding passes of the first frame with RD slopes $\geq T_{RD}^p$ will fit into the primary buffer.

Determine the secondary RD threshold T_{RD}^s such that the remaining coding passes of the first frame with RD slopes $\geq T_{RD}^s$ will fit into the secondary buffer.

Delete the coding passes of the first frame with RD slopes less than T_{RD}^s.

for $k = 2$ to $k = N + M$

    if $k \leq N$
        Determine T_{RD}^p so that coding passes of the frames currently in the buffer and those of the kth frame with slopes $\geq T_{RD}^p$ will fit in the primary buffer.

        Determine T_{RD}^s so that remaining coding passes of the frames currently in the buffer and those of the kth frame with slopes $\geq T_{RD}^s$ will fit in the secondary buffer.

        Delete the coding passes of the frames currently in the buffer and those of the kth frame with RD slopes $< T_{RD}^s$.

        Insert the qualifying coding passes of the kth frame into the primary and secondary buffers.

    end if

    if $k > M$
        Release SR bits from the head of the primary buffer to the codestream.
    end if

    set $k = k + 1$

end for

Table 2. DBRC algorithm.

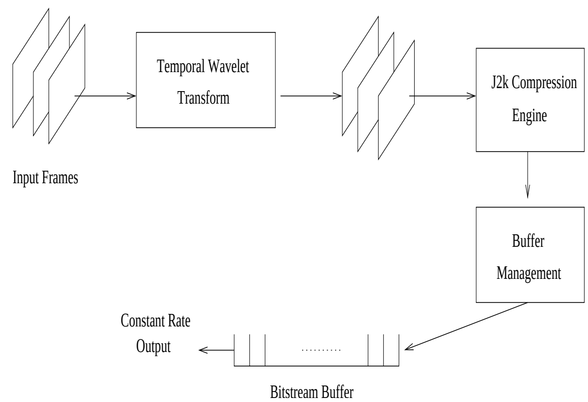

Figure 3. Block diagram of the 3D wavelet coding scheme.

can be controlled by selecting the number of frames in each GOF. Unfortunately, this approach results in subtantial performance loss, especially on GOF boundaries. The decoder still needs to wait until all of the frames in the GOF are received, before the inverse transform can be performed. This buffering increases the memory requirements of these schemes as well.

Recent research activity has concentrated on achieving low memory implementations of the wavelet transform. In [20,21], the authors have presented image compression methods that can provide excellent compression performance while requiring only a fraction of the memory of traditional implementations. In fact, the low memory wavelet transform schemes used in these works produce wavelet coefficients that are identical to those produced by the traditional schemes. Thus, no loss is incurred due to these low memory implementations of the wavelet transform. It is possible to extend the ideas of [20,21] to achieve a low-memory, low-latency 3D wavelet video coding scheme. Such an approach would not require the use of small GOF, and coupled with the RC algorithms presented here, could yield constant quality video.

The low memory implementation of the temporal wavelet transform is performed in a "sliding window". The basic idea of such an implementation is to utilize buffers to perform the transform. The input samples are placed in these buffers and the wavelet coefficients are generated as soon as all the samples that contribute to that coefficient become available. The size of the sliding window is determined by the length of the wavelet filters and the number of dyadic decomposition levels.

It is important to realize that the memory use of the reduced memory wavelet transform can be attributed to two different buffers: filtering buffer and synchronization buffer. While the filtering buffer is needed to perform the transform, the synchronization buffer ensures synchronization between the encoder and the decoder. The synchronization buffer can be implemented during either the forward or the inverse transform. It can also be divided between the forward and the inverse transform stages. This provides increased flexibility in designing a reduced memory wavelet transform for a particular application. For example, in a broadcast application, the cost of memory at the receiver may need to be reduced. Thus, in such an application the synchronization buffer can be implemented at the encoder. For a detailed analysis of low-memory implementations of the wavelet transform, the interested reader is referred to [20,21].

4.2. Experimental Results

We present here the simulation results for the DBRC-based 3-D wavelet video coding (3DWT-DBRC). Figure 4 illustrates the performance of our algorithm on the Trevor sequence. For comparison purposes, we also show in Figure 5 the performance of the DBRC algorithm on the same Trevor sequence but with the third dimensional transform turned off. From these figures, one can see that for buffer sizes corresponding to 20, 40 and 150 compressed frames [*], the average PSNR increases by 18% for each. Moreover, for buffer sizes corresponding to 20 and 40 compressed frames, the PSNR variance decreases by 41% and 21%, respectively, compared to the DBRC.

[*]Note that the concept of buffer content differ between the DBRC and the 3DWT-DBRC. In the former, the buffer content are compressed domain image frames while in the latter, it is compressed domain temporally filtered image frames.

However, for the extreme case when the buffer size corresponds to the entire compressed temporally filtered frames, the PSNR variance increases by 96% compared to the DBRC. The reason behind this is the cyclostationarity of the quantization noise in wavelet-based codecs.

The other extreme case to look at is when the buffer size corresponds to the size of one compressed frame. In this case, the third dimensional transform does not provide any gain over the DBRC for the same buffer size: The PNSR variance and average PSNR are the same in this case. This is quite expected since encoding each wavelet coefficient independently from others does not allow us to allocate rate across temporal subbands. Hence, the two cases provide similar results.

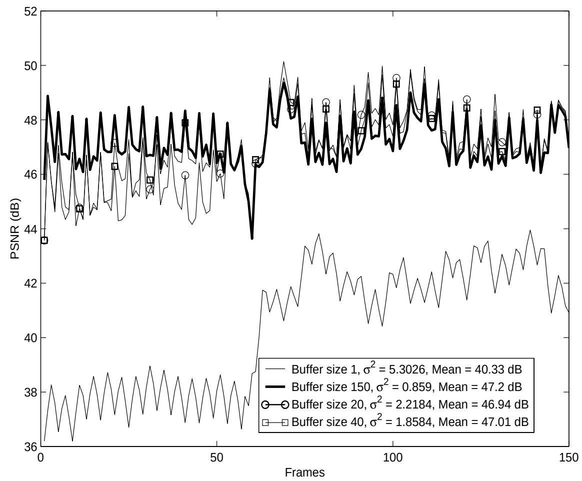

Figure 4. The performance of the DBRC algorithm with a third dimension transform on the Trevor sequence encoded at an average rate of 1.0 bpp.

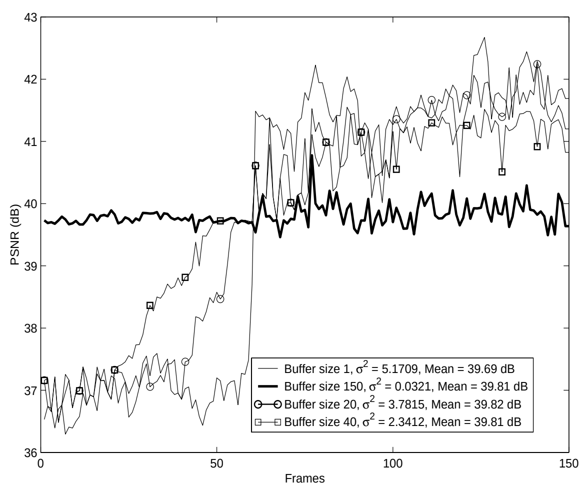

Figure 5. The performance of the DBRC algorithm without a third dimension transform on the Trevor sequence encoded at an average rate of 1.0 bpp.

5. MULTISEQUENCE VIDEO STREAMING

When a number of compressed video streams are to be transmitted through a common bandlimited channel, as in video on-demand and in digital video broadcasting applications, the simplest approach is to divide the available channel bandwidth equally among all video streams. This approach is known as Constant Bit Rate (CBR) video coding. However, there are some disadvantages associated with this approach. At any instance in time, the quality of the video streams will vary widely due to different content and the channel throughput will not be fully utilized in a rate-distortion sense. Thus, a Variable Bit Rate (VBR) video coding scheme which allows different video streams to be compressed at different rates would be beneficial [22] . Furthermore, since the video content of each stream changes over time, it would also be beneficial to change the rate of each video stream in a dynamic fashion. This will allow constant quality across all video streams. This problem is treated in [22–24] .

In this paper, we provide a solution for the above stated problem using the DBRC algorithm. The basic block diagram of our system is depicted in Figure 6 . In the figure, P different video sources are being fed into the compression engine through a multiplexer which selects the frames at an adjustable speed of L fps. We assume that, with the help of a controller, the encoder can keep track of the number of video sources being multiplexed, P, along with the channel bandwidth, C bps. Then, depending on the size of the Bitstream Buffer, M frames from the P independent video sequences are placed into the rate controller. Rate allocation is then carried out using the same DBRC algorithm described above in Table 3.2 . At a given instance in time, more bits might be allocated to one video source over the others, depending on video contents of all video sources residing at the buffer at that time. This enables the video quality to be constant over time and over the P different video sequences. The proposed algorithm offers several advantages over existing methods. First, our DBRC-based algorithm does not require additional computations for determining the content complexity of each frame [22–24] . The rate-distortion information corresponding to every coding pass of every frames is already produced by the encoder, and this information is simply passed to the rate controller. The rate controller is able to assess the importance of each coding pass without further analysis. Another advantage of the proposed scheme is that it is strictly a post-compression operation. Since the rate controller operates on the compressed bitstream, a single encoder running at a rate slightly higher than the target rate is sufficient to achieve constant quality. Unlike existing schemes, the encoder does not need continuous feedback from the rate controller.

It should also be noted that the presented scheme can dynamically accomodate conditions such as the number of bitstreams varying over time, different frame rates, etc. Some examples of these conditions are illustrated in the next section.

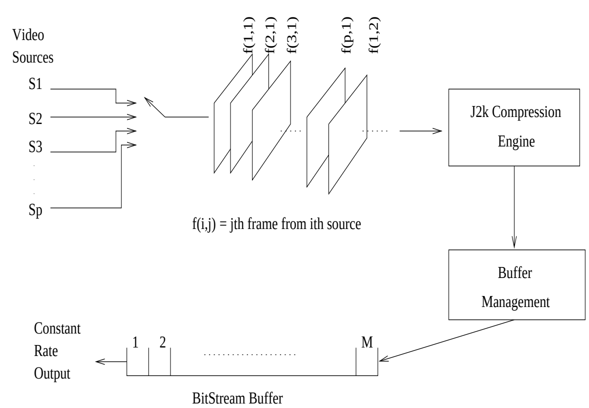

Figure 6. Block diagram of the DBRC multisequence rate controller.

5.1. Experimental Results

Here, we show the results of experiments obtained using five different sequences of 150 frames each, encoded at an average rate of 0.5 bits/pixel/sequence. Figure 7 shows the SNR performance for the five sequences, multiplexed to yield a single sequence of $5 \times 150 = 750$ frames. In this figure, the circles indicate the results for buffering a single frame. This corresponds to the fixed rate case, where each frame is allocated the same rate. The widely varying SNR values correspond to the frames from different sequences. The light dotted line indicates the performance where rate allocation is performed globally over all 750 frames. As expected, near constant quality is achieved. Finally, the heavy dots indicate the performance achieved when rate allocation is performed jointly employing a “sliding window” of five frames. Figure 7 shows that the performance of our algorithm with a buffer size corresponding to 5 frames (i.e. only 1 frame delay per sequence) and 750 frames (maximum delay) are very close.

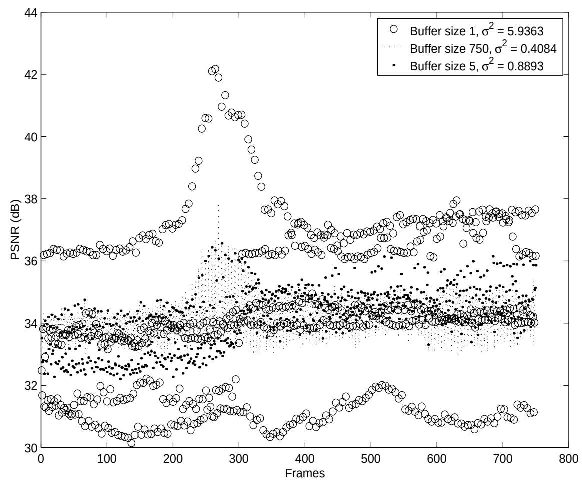

Figure 7. The performance of the DBRC algorithm on five multiplexed sequences.

Moreover, interesting results are obtained when we look at each sequence individually. Figure 8 extracts the results from Figure 7 correponding to a single sequence. The buffer size 1 case in this figure corresponds to a single frame latency if the sequence were coded in isolation. The buffer size 5 case also corresponds to a single frame of latency when rate control is performed jointly for the five sequences. It can be seen that for the same amount of latency, the variance of the PSNR values decreases by 93% under our method.

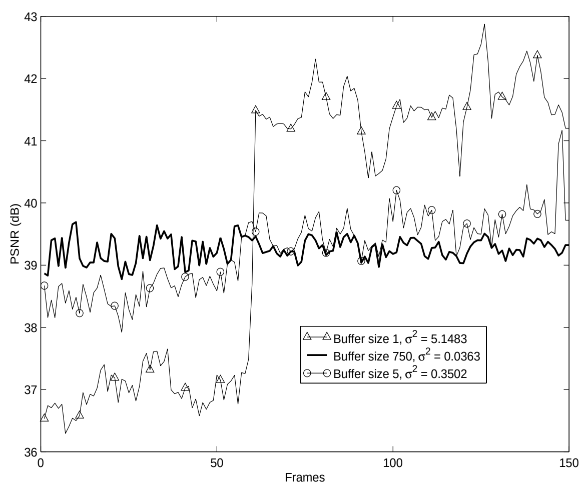

Figure 8. The performance of the DBRC algorithm on the fifth multiplexed sequence, Trevor.

Also, the simulations show that the proposed algorithm adapts well to varying conditions such as scene changes or sudden halts in video sequences. We consider an interesting scenario, where at some point in time, one of the video sequences is stopped, and the algorithm is required to allocate the resources of the channel among the remaining four sequences. In our example, the fifth sequence is halted at frame 60 which corresponds to frame number 300 in the interleaved sequence. Figure 9 illustrates the performance of our algorithm when this scenario occurs. Notice that the algorithm starts allocating more rate to the remaining four sources, the net result of which is an increase in the average PSNR.

6. CONCLUSIONS

In this paper, we present two rate control algorithms that can be applied to any compression scheme capable of fine scalability. The first rate control algorithm presented is for 3D wavelet video coding and the second rate

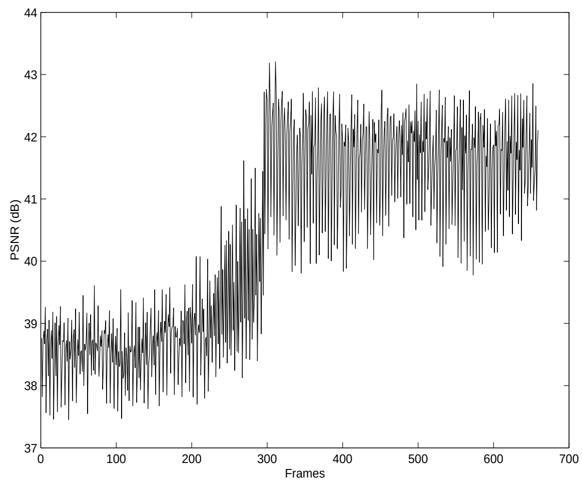

Figure 9. The performance of the DBRC algorithm on five multiplexed sequences. Sequence 5 stops at frame 60.

controller describes an algorithm for Multisequence video streaming. The algorithms significantly reduce the quality fluctuations among frames, and provide smoother video sequences. Simulations show that the proposed algorithms adapt well to varying conditions.

REFERENCES

1. H. Radha, Y. Chen, K. Parthasarathy, and R. Cohen, "Scalable internet video using MPEG-4," *Signal Processing: Image Communication* **15**, pp. 95–126, Sept. 1999.
2. M. van-der Schaar and H. Radha, "A hybrid temporal-SNR fine-granular scalability for internet video," *IEEE Transactions on Circuits and Systems for Video Technology* **3**, pp. 318–331, 2001.
3. J. Rexford, S. Sen, and A. Basso, "A smoothing proxy service for variable-bit-rate streaming video," in *Global Telecommunications Conference-GLOBECOM'99*, vol. 3, pp. 1823–1829, 1999.
4. D. Reininger, M. Ott, G. Michelitsch, and G. Welling, "Scalable QoS control for VBR video servers," in *IEEE First Workshop on Multimedia Signal Processing*, pp. 570–575, 1997.
5. E. Bommaiah, K. Guo, M. Hofmann, and S. Paul, "Design and implementation of a caching system for streaming media over the internet," in *Sixth IEEE Real-Time Technology and Applications Symposium*, pp. 111–121, 2000.
6. J. C. Dagher, A. Bilgin, and M. W. Marcellin, "Efficient rate control for video streaming," in *Applications of Digital Image Processing XXIII, Proc. of SPIE*, July 2001.
7. C. Podilchuk, N. Jayant, and N. Farvardin, "Three-dimensional subband coding of video," *IEEE Transactions on Image Processing* **4**, pp. 125–139, 1995.
8. J.-Y. Tham, S. Ranganath, and A. A. Kassim, "Highly scalable wavelet-based video codec for very low bit-rate environment," *IEEE Journal on Selected Areas in Communications* **16**, pp. 12–27, 1998.
9. A. Wang, Z. Xiong, P. A. Chou, and S. Mehrotra, "Three-dimensional wavelet coding of video with global motion compensation," in *Proceedings DCC'99 Data Compression Conference*, pp. 403–413, 1999.
10. B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)," *IEEE Transactions on Circuits and Systems for Video Technology* **10**, pp. 1374–1387, 2000.
11. J. Xu, Z. Xiong, S. Li, and Y. Zhang, "Three-dimensional embedded subband coding with optimized truncation (3-D ESCOT)," *Applied-and-Computational-Harmonic-Analysis.* **10**, pp. 290–315, 2001.
12. "JPEG 2000 Part I Final Draft International Standard," *ISO/IEC JTC 1/SC 29/ WG1, Doc. No. N1855* , Aug. 2000.
13. D. S. Taubman and M. W. Marcellin, *JPEG2000: Image Compression Fundamentals, Practice and Standards*, Kluwer Academic Publishers, Massachusetts, 2002.
14. M. Marcellin, M. Gormish, A. Bilgin, and M. Boliek, "An overview of JPEG-2000," in *Data Compression Conference*, pp. 523–541, Mar. 2000.

15. "Motion JPEG2000 (MJP2) requirements and profiles version 6.0," *ISO/IEC JTC 1/SC 29/ WG1, Doc. No. N2106* , Mar. 2001.
16. T. Fukuhara and D. Singer, "Motion JPEG2000 verification model ver.4.0 (technical description)," *ISO/IEC JTC 1/SC 29/ WG1, Doc. No. N1983* , Jan. 2001.
17. T. Fukuhara, "Presentation on Motion JPEG-2000," *ISO/IEC JTC 1/SC 29/ WG1, Doc. No. N1389* , July 1999.
18. N. S. Jayant and P. Noll, *Digital Coding of Waveforms*, Prentice-Hall, New Jersey, 1984.
19. T. Flohr, M. Marcellin, and J. Rountree, "Scan-based processing with JPEG-2000," in *Applications of Digital Image Processing XXIII, Proc. of SPIE*, vol. 4115, July 2000.
20. C. Chrysafis and A. Ortega, "Line-based, reduced memory, wavelet image compression," *IEEE Transactions on Image Processing* **9**, pp. 378–389, 2000.
21. E. Ordentlich, D. Taubman, M. Weinberger, G. Seroussi, and M. Marcellin, "Memory-efficient scalable line-based image coding," in *IEEE Data Compression Conference*, pp. 218–227, 1999.
22. M. Balakrishnan, R. Cohen, E. Fert, and G. Keesman, "Benefits of statistical multiplexing in multi-program broadcasting," in *IEEE Broadcasting Convention*, pp. 560–565, 1997.
23. D. Hoang and J. Vitter, "Multiplexing vbr video sequences onto a CBR channel with lexicographic optimization," in *IEEE International Conference on Image Processing*, pp. 369–372, 1997.
24. L. Boroczky, A. Ngai, and E. Westermann, "Joint rate control with look-ahead for multi-program video coding," *IEEE Transactions on Circuits and Systems for Video Technology* **10**, pp. 1159–1163, 2000.