PSNR Control for GOP-level Constant Quality in H.264 Video Coding

Fabio De Vito and Juan Carlos De Martin
Dipartimento di Automatica e Informatica
Politecnico di Torino, 10129, Torino, Italy
email: {fabio.devito,demartin}@polito.it

Abstract – For some multimedia applications, like storage or surveillance, it may be desirable to encode the content at constant quality level; the encoder should be aware of media content and set suitable parameters.

In this work, we analyze the case of video. We propose an algorithm to adapt the quantization, to obtain constant PSNR. This algorithm selects each frame's quantizer level by comparison with the average of PSNR over N preceding frames, and modifying the quantizer accordingly. This system proved to obtain the target average with a variance in the order of 0.3 in most of the cases.

Furthermore, the application may require a per-GOP constant PSNR, like in semantic-based video coding. To obtain this effect, we propose a method to track a given PSNR pattern. The proposed algorithm is able to achieve the target PSNR for each single shot, with a convergence time of only few frames after shot boundary. [1]

Keywords – Constant quality, PSNR-based coding, video coding, H.264 codec.

I. INTRODUCTION

Storage and transmission of multimedia signals usually involve an accurate choice of compression parameters. In the case of video, the media nature and the coding techniques require several parameters to be set properly, according to the compression ratio and the quality needed. The trade-off between compression and quality can be considered under two points of view. A long-term compromise can be set, to monitor the average required bandwidth or to meet some constraints on the space needed for storage; on the other hand, short-term tuning can be performed to obtain similar performance on consecutive or nearby frames, for example avoiding large oscillations in the bitrate, which can cause delay jitter increase or introduce losses, or wide differences in PSNR, which can create annoyance to the user.

Both compression ratio and quality are correlated to several parameters, like the GOP structure, the frame temporal and spatial resolution, coding modes for macroblocks (intra- or inter-mode), and the quantization parameter. In general, the achieved compression is inversely proportional to the quality of the stream; this proportionality is in most of the cases non linear. Both compression and quality are correlated to the choice of the quantization parameter to be used for each frame. While coding, once an overall target is set, either for bitrate or PSNR, local refinements can be driven in order to force the desired local effects.

Depending on the application, the source can encode the stream taking care of only one among bitrate and PSNR, usually to obtain constant rate or constant quality; techniques also exists to optimize the choice on both of them jointly [1].

In *rate-distortion optimization* (RDO) [2]–[4], the coding parameters (coding modes, quantization) are chosen to reach a compromise between the bitrate and the quality. Usually, at high bitrate a further increase does not lead to noticeably better quality; instead, at low bitrates, even a small variation in PSNR leads to relatively wide oscillation in the number of bits required. Joint decision can help in achieving the best choice. This optimization is particularly useful when transmitting video sequences over packet data networks, since it results both in good network utilization and PSNR, given the video characteristics.

In constant bitrate (CBR) coding, the encoder does not take care of the quality; the main constraint is achieving as precisely as possible a target bitrate [5], [6]. This approach is necessary when transmitting the video over a CBR channel, or in general when oscillation in the bitrate could result in severe information losses or late delivery due to jitter increase.

Constant quality coding is the dual of constant bitrate, and aims at encoding the sequence at a constant PSNR level, regardless of the number of bits required [7]–[10] for the entire video, or for its portions [11]. Although less suitable for information transmission due to the possible high peaks in the network utilization, this approach may be desirable in some applications like remote control of machinery or surveillance, where quality is a key issue and usually no bandwidth restriction is present. It can be also useful for data storage application, like for master copies of movies, where ensuring high fidelity with respect to the original sequence is more important than the memorization space required; this has also been made possible because the price of storage devices is nowadays decreasing.

In this paper, we propose a simple algorithm to achieve constant PSNR, and we show the results of its implementation in the H.264 video coder.

The paper is organized as follows. In Section II the framework is outlined; we describe the proposed algorithm in Section III, together with its parameters in Subsection III-A. In Section IV we present an evolution of this algorithm to modify the PSNR target during the encoding process, and to

---

[1] This work is supported by Centro Supercalcolo Piemonte (CSP), Torino, Italy.

track a given quality pattern; results are discussed in Section V and conclusions are drawn in Section VI.

II. BACKGROUND

To obtain a nearly-constant video quality (in the following, we will refer to this as *CPSNR*), a very simple approach may be considered setting a constant quantization parameter (*QP*). The main issue in achieving constant PSNR by means of a fixed QP parameter is represented by the lack of uniformity in the sequence content; the same quantizer level can lead to different PSNR's indicators if applied to different frames. Video streams are usually stationary for runs of frames, which are often referred as *shots*; this does not necessary mean that the video content remains the same or similar, but that all of the frames within a shot have some common characteristics of detail level and amount of movement. This is to say, usually a nearly constant PSNR can be obtained using a fixed QP, if this approach is restricted to each run of frames. With lower probability the same QP will produce the same effect on PSNR for different shots. If, at constant QP, the quality changes significantly (i.e., the difference is wider than a given threshold), there are high chances that the video content is changing as well, and a new shot is beginning.

Given this behavior, whenever a noticeable change in the PSNR is obtained with respect to the average of a certain number of previous frames, it can be useful to modify the quantization parameter by a given number of steps, to ensure the desired quality for the new content characteristics, therefore adapting to the video content; if the new QP is chosen wisely, the PSNR of the next frame will be closer to the target value.

Several approaches can be employed to obtain this adjustment. One of them consists in re-encoding the same frame for which the PSNR revealed to be far from the target, using a different quantization at each iteration; the operation can be repeated until no further enhancement is detected. This solution will ensure the narrowest oscillations around the desired value. Conversely, it will require in some cases several encodings for the same frame, so increasing the computational complexity, in particular at shot boundaries.

Another feasible approach is based on the observation of the past PSNR history; at each frame coding, the target value is compared with the average of the PSNR of the last N frames, and if the difference is higher than a fixed threshold, the quantization parameter should be changed. This will produce wider oscillations with respect to the previous approach, but the computational complexity of the control routine will be negligible.

The algorithm we propose in this work follows this technique. It adapts QP on a per-frame basis, so ensuring fast convergence, and it is tunable to allow different trade-offs between convergence speed and the amplitude of oscillations around the target PSNR value.

The key quantities we will provide to show the algorithm behavior are the average μ and the variance σ^2 of the PSNR's within each sequence.

III. IMPLEMENTATION OF THE PSNR CONTROL

The encoder typically saves internally some statistics on the last frame coded; at the end of each frame, we update the average of the last N PSNR values. Before starting each new frame, we compare this average with the target PSNR value. If the difference is below a given threshold Δ, then the next frame will be coded using the same quantization parameter as the previous one:

$$\left| \frac{1}{N} \sum_{i=-1}^{-N} PSNR_{j-i} - T \right| \leq \Delta \Rightarrow QP_j = QP_{j-1} \qquad (1)$$

where j is the index of the next frame that will be coded, and T is the target PSNR. In case the difference is larger than the threshold, the following frame will be coded at higher or lower QP, depending on whether the difference is positive or negative.

$$\left| \frac{1}{N} \sum_{i=-1}^{-N} PSNR_{j-i} - T \right| > \Delta \Rightarrow QP_j = QP_{j-1} + \delta; \qquad (2)$$

δ is given by:

$$\delta = \pm \min \left\{ \left[\gamma \left| \frac{1}{N} \sum_{i=-1}^{-N} PSNR_{j-i} - T \right| \right], K \right\} \qquad (3)$$

where $[x]$ denotes the integer part of x and $K \in \mathbb{N}^+$. The sign of this quantity is the same of the difference between the average PNSR of last frames and the target T. The effect of this approach is an increase of QP if the average is higher than the target (so coding at lower quality), or a decrease in the opposite situation, so increasing the PSNR. The maximum correction width is ruled by the parameter K, which limits the number of quantization steps to move up or down, preventing the algorithm from being excessively aggressive and reacting with large variations to local effects. The parameter γ is introduced as a scaling factor, and may depend on the particular sequence. The effect of parameters selection (N, Δ, γ and K) will be discussed in the following.

A. Algorithm parameters

The proposed algorithm behavior can be modified if the following parameters are changed:

- Δ, the difference threshold after, which QP is adjusted;
- *N*, the number of previous frame used to compute the average for comparison;
- γ, the slope of the dependency of QP on PSNR;
- *K*, the maximum correction in the quantization parameter.

We will describe the effects of changing each one of these parameters, showing in the results for a target PSNR of 36 dBs and sequence *Foreman*. In the following, all the sequences we will refer to are intended as QCIF format, 30 frames per second with GOP size of 30 frames and no B-frames used.

TABLE I

AVERAGE AND VARIANCE OF PSNR VERSUS N; $\Delta = 1$, $\gamma = 0.7$, $K = 3$.

Parameter	PSNR	
N	μ	σ^2
1	35.73	0.16
2	35.76	0.14
3	35.92	0.13
4	36.04	0.33
5	36.22	0.22
6	36.20	0.33

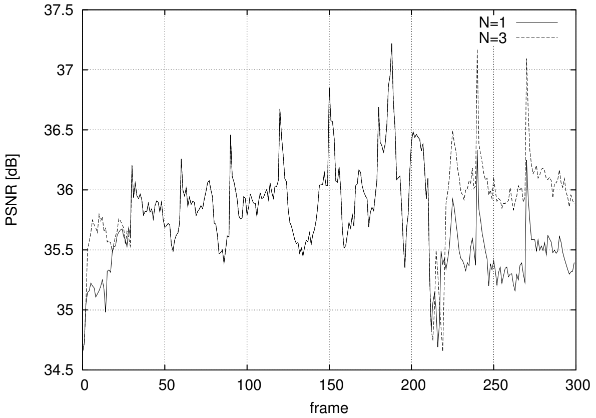

Fig. 1. Temporal evolution of the PSNR with $N = \{1, 3\}$; $\Delta = 1$, $\gamma = 0.7$, $K = 3$, sequence *Foreman*.

Anyway, the usage of B-frame is allowed without any further modification to this algorithm.

Table I shows how the average and variance of the PSNR evolve when the window length N is increased; the variance decreases up to $N = 3$, then increases again.

Increasing the number of frames on which the average is computed gives a better estimate of the sequence characteristics in the near past, hence the smaller variance. If N is small, and for any reason a frame results in a PSNR far from the target, then the algorithm will react immediately; if that frame was not the beginning of a new shot but only an isolated case, then this reaction was useless. On the other hand, if N is big, when a new shot begins, the reaction time is slow due to the time needed to replace values in the memory. If the average is far from the target, it will become closer slowly after at least a number of frames equal to the length of the window used for average computation, provided that the sequence characteristics remain stationary and do not change again in few frames. Figure 1 shows the oscillations of the algorithm for $N = 1$ and $N = 3$.

For the following experiments, we will set $N = 3$ as best choice.

Table II shows how the threshold Δ modifies the performance of the algorithm. In this case, the variance minimum is reached in the region around 1.1.

TABLE II

AVERAGE AND VARIANCE OF PSNR VERSUS Δ; $N = 3$, $\gamma = 0.7$, $K = 3$.

Parameter	PSNR	
Δ	μ	σ^2
0.1	35.87	0.18
0.3	35.87	0.18
0.5	35.87	0.18
0.7	35.87	0.18
0.9	35.98	0.28
1.1	35.77	0.14
1.3	35.36	0.15
1.5	35.53	0.28

TABLE III

AVERAGE AND VARIANCE OF PSNR VERSUS K; $N = 3$, $\gamma = 0.7$, $\Delta = 1$.

Parameter	PSNR	
K	μ	σ^2
1	35.92	0.13
2	35.92	0.13
3	35.92	0.13

When Δ parameter is small, the algorithm reacts even to small differences between the average and the target. Since the quantizer can be only moved among integer numbers and the effect of a single unit variation in QP may result in a PSNR change in the order of 0.2 to 0.3 dBs, then small values of Δ may force useless and wide oscillations. Conversely, high values of Δ push the algorithm not to react even when noticeable PSNR deviation is present. In the following, we will assume a value of $\Delta = 1$ as best choice.

Other parameters that can influence the performance of this algorithm are the ones involved in Formula (3), even if their effect is less evident as the ones previously presented.

Table III present the behavior when the parameter K is changed. Its effect is minimal, yet we leave it in our formulation to allow prevention of excessively wide reactions. The choice for further experiments in this study is $K = 3$.

Table IV shows the last parameter, γ. This parameter represents the slope of the relation between changes in QP and changes in PSNR. This scaling factor has the smallest values of σ^2 in the region between 0.7 and 1.1.

As a result of this parameter test, we indicate the setting

$$\begin{cases} N = 3 \\ \Delta = 1 \\ \gamma = 0.7 \\ K = 3 \end{cases}$$

as our choice to perform further experiments in this study.

IV. TRACKING A GIVEN PSNR PATTERN

For some particularly stationary sequences, the proposed approach shows the same performance of a constant QP

TABLE IV

AVERAGE AND VARIANCE OF PSNR VERSUS γ; $N = 3$, $K = 3$, $\Delta = 1$.

Parameter	PSNR	
γ	μ	σ^2
0.1	35.95	0.16
0.3	35.95	0.16
0.5	35.76	0.23
0.7	35.87	0.13
0.9	35.84	0.13
1.1	35.85	0.14
⋮	⋮	⋮
2	34.93	0.27

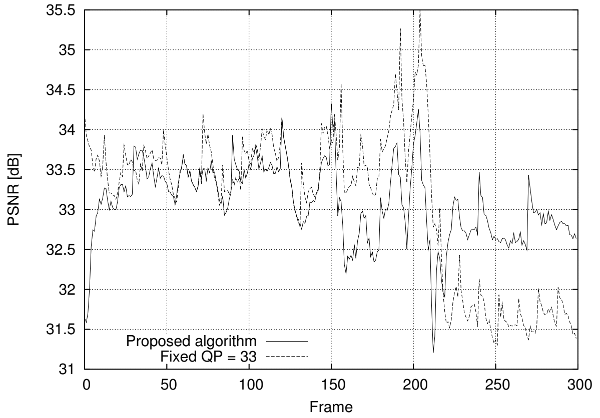

Fig. 2. Temporal evolution of PSNR for sequence *Foreman*, using different coding control approaches; target is 36 dBs.

solution; the advantage is that with our scheme we do not need to know the suitable quantizer level for the sequence, since it will converge to that value in few frames. Figure 2 and Figure 3 show the PSNR obtained using both the proposed approach and a fixed quantization parameter. For sequence *Foreman*, it is evident the advantage of the proposed algorithm over the simple constant-QP solution, since the latter would not perform well in the second half of the video. For *Tempete* the two curves are coincident; after a starting convergence time the proposed algorithm converges to the best QP and continues with narrow oscillations until the end.

Fast adaptivity can be useful if the user does not desire a uniform PSNR level for the entire sequence. If the video is segmented into shots, and each shot is assigned a value of semantic importance chosen among a set of predefined levels, then the user can map each level to a given PSNR according to his preference. For example, in sports video, it is possible to define two semantic levels, i.e., action or not action. The user can decide to receive action shots encoded at a given PSNR, say 37 dBs, and the remaining shots at a lower quality, for example at 30 dBs. This approach is general and it is possible in principle to require a different PSNR value for each shot; the convergence capability of the algorithm will take care of obtaining the desired PSNR.

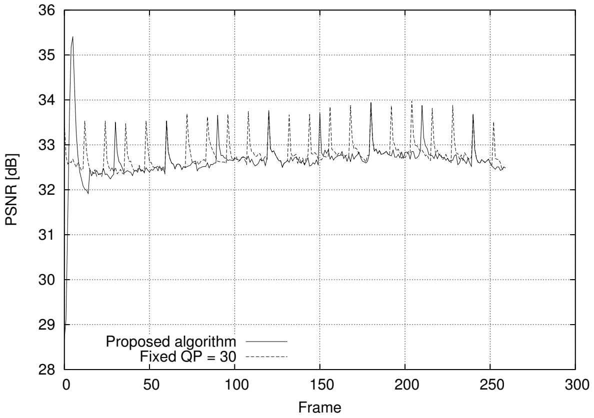

Fig. 3. Temporal evolution of PSNR for sequence *Tempete*, using different coding control approaches; target is 36 dBs.

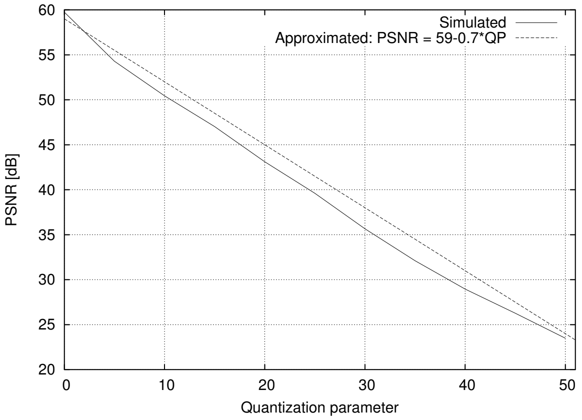

Fig. 4. PSNR versus QP used for switching capability implementation; sport video curve and proposed approximation.

At each change in the requested video quality, the convergence speed could be slow if the difference with the average of previous frames' PSNR is wide; this is because the algorithm will start encoding the first frame in the shot at the QP of the last frame in the preceding shot. To speed up this convergence, at each new request, the starting QP is recomputed to a value approximatively suitable for the new PSNR; the precision of this mapping is only relatively important, since it will also depend on the particular video content. The effect should be moving the QP in a suitable region for the PSNR required for the new shot, and then leaving further precision to be achieved by the convergence routine. Collecting statistics, a simple and linear formula to determine this working point resulted to be

$$PSNR(QP) = 59 - 0.7 \cdot QP \tag{4}$$

which is also shown in Figure 4, together with a curve computed for a soccer match video.

From Formula (4), it is possible to compute the starting QP for each shot as:

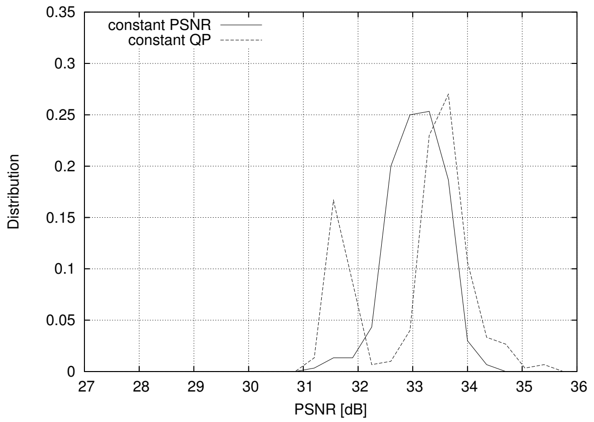

Fig. 5. Distributions of PSNR for sequence *Foreman*, using different coding control approaches.

TABLE V

PERFORMANCE OF THE PROPOSED ALGORITHM COMPARED TO CONSTANT QP APPROACH; A TARGET OF 33 DBS IS REQUESTED

Sequence	Constant parameter	value	PSNR	
			μ	σ^2
foreman	PSNR	33 dB	33.07	0.24
	QP	33	33.09	0.91
tempete	PSNR	33 dB	32.66	0.25
	QP	30	29.85	0.25

$$QP(PSNR) = \frac{59 - PSNR}{0.7}. \quad (5)$$

V. RESULTS

A. Constant PSNR

As stated in Section IV, constant quality can be easily obtained for stationary sequences, provided that the correct quantization parameter is somehow known. This is not true for non stationary videos like *Foreman*. In this case, using a constant quantization parameter leads to lower quality in the second part of the sequence, which contains high motion. In Figure 5 we show the distributions of PSNR's obtained for this sequence with a constant QP equal to 33 and with our adaptive approach.

The distribution obtained with constant QP is formed by two peaks, the one on the right belonging to the first (stationary) part, the second coming from the last high-motion frames, where the same quantizer level results in poorer quality. The distribution obtained with the proposed approach shows instead a single peak, centered in the desired target PSNR and with a variance of 0.34 as also reported in Table V.

In the same Table, results for sequence *Tempete* are also reported, showing the substantially similar performance of the two approaches for this kind of sequences. Interestingly, the requested 33 dBs are obtained with $QP = 33$ for *Foreman* and $QP = 30$ for *Tempete*, confirming that the suitable QP cannot determined a priori without knowledge of the content.

TABLE VI

PERFORMANCE (MEAN AND VARIANCE) OF THE PROPOSED ALGORITHM FOR DIFFERENT SEQUENCES AND DIFFERENT REQUESTED PSNR'S; THE OUTPUT AVERAGE BITRATE IS ALSO INDICATED.

Sequence	PSNR [dB]			Bitrate [kbps]
	Requested	Obtained		
		μ	σ^2	
foreman	30	29.93	0.22	49
	33	33.07	0.24	91
	36	35.92	0.13	166
tempete	30	29.85	0.25	136
	33	32.66	0.25	266
	36	35.84	0.23	495
paris	30	30.14	0.25	61
	33	32.52	0.11	93
	36	35.69	0.09	156
news	30	29.78	0.61	29
	33	33.31	0.39	53
	36	36.15	0.29	85
table	30	29.70	0.36	44
	33	33.15	0.22	82
	36	36.14	0.17	162

Table VI compares the performance of the proposed algorithm for five sequences at three different target PSNR values.

The main parameter of this table, the variance σ^2, is in the majority of the cases below 0.3, showing that the proposed quality control algorithm is able to achieve the desired value with limited oscillations; typically the widest one is on the first frame of each GOP.

For comparison, for sequence *Foreman*, a variance of 0.22 is achieved at nearly 30 dBs, equivalent to a standard deviation $\sigma = 0.46$; for the same sequence the approach of [10] reveals $\sigma = 1.33$, nearly three times the one obtained with our approach.

B. Tracking PSNR

The control routine structure allows changing the desired value of PSNR at shot boundaries. In principle, it is possible to change it at every I-frame. We will now present results on the ability of this algorithm to track a given per-GOP PSNR, switching among widely different values. The proposed pattern is reported in Figure 6, along with the per-frame PSNR obtained with two test sequences.

In this setting, changes occur every two GOPs. It is possible to notice the oscillations around the target value, especially on I-frames, and the high speed of convergence at shot boundaries.

When working at constant PSNR, the bitrate obtained is unconstrained. This effect is shown in Figure 7, where the bitrate obtained while tracking the quality pattern of Figure 6 is shown.

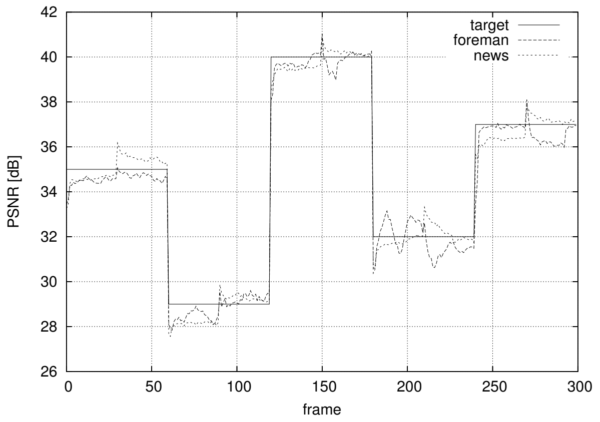

Fig. 6. Tracking a given PSNR pattern for two sequences.

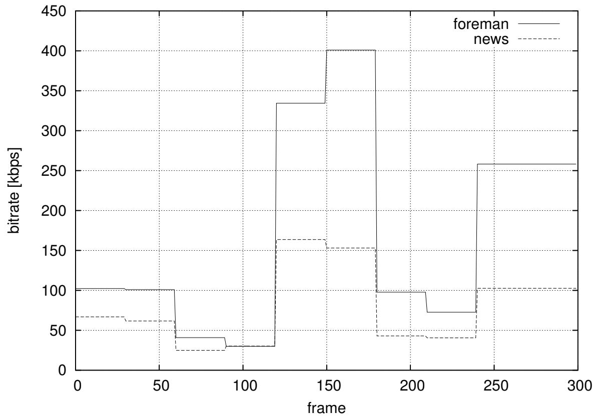

Fig. 7. Tracking a given PSNR pattern for two sequences, obtained bitrate.

For the region coded at 40 dBs, the resulting bitrate can arrive up to 400 kbps for sequence *Foreman*, which contains high motion in the second half.

VI. CONCLUSIONS

In this work, we proposed a new PSNR control algorithm to ensure constant-quality coding for H.264 compressed video sequences. This algorithm is able to converge in few frames to the desired PSNR level and can remain around that value with small oscillations even if the video content changes fastly.

The variance of the obtained PSNR is in most of the cases below 0.3. The algorithm can be tuned by a set of parameters, whose influence on performance has been shown. The routine can also converge to different values of PSNR for different GOPs, so making possible to map user preferences on the video into a target coding quality.

Since it is in principle possible to have a switch for every GOP, a method to improve convergence speed has also been presented. The proposed algorithm is able to track the PSNR for each segment of the video with few frames of convergence time.

REFERENCES

[1] S.F. Chang and A. Vetro, "Video adaptation: Concepts, technologies and open issues," *IEEProceedings of the IEEE*, vol. 93, no. 1, pp. 148–158, January 2005.

[2] T. Wiegand G.J. Sullivan, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, November 1998.

[3] T. Wiegand, M. Lightstone, D. Mukherjee, T. George, and S. K. Mitra, "Rate-distortion optimized mode selection for very low bit rate video coding and the emerging H.263 video standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 2, pp. 182–190, Apr. 1996.

[4] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 23–50, November 1998.

[5] A. Ortega, Ed., *Variable Bit-Rate Video Coding, in Compressed Video over Networks*, pp. 343–382, M.-T. Sun and A. R. Reibman, Eds, New York, NY, USA, 2000.

[6] B. Li and J. Liu, "Multirate video multicast over the internet: An overview," *IEEE Network, Special Issue on Multicasting: An Enabling Technology*, vol. 17, no. 1, pp. 24–29, January 2003.

[7] M. Ringenburg, R. Ladner, and E. Riskin, "Global minmax interframe bit allocation for embedded video coding," in *Proceedings of Data Compression Conference*, Snowbird, UT, USA, 2004, pp. 222–231.

[8] B. Xie and W. Zeng, "Sequence based rate control for constant quality video," in *Proc. IEEE Int. Conf. on Image Processing*, Rochester, NY, USA, Sept. 2002, pp. 77–80.

[9] P. Chen and J. Woods, "Video coding for digital cinema," in *Proc. IEEE Int. Conf. on Image Processing*, Rochester, NY, USA, Sept. 2002, pp. 749–752.

[10] M. Jiang, X. Yi, and N. Ling, "Frame layer bit allocation scheme for constant quality video," in *Proc. IEEE Int. Conf. on Multimedia & Expo*, Taipei, Taiwan, June 2004, pp. 1055–1058.

[11] L. Overmeire, F. Verdicchio, J. Barbarien, P. Shelkens, and L. Nachtergaele, "Constant quality rate control for video encoding based on activity segmentation," in *Proc. IEEE Int. Conf. on Image Processing*, Singapore, Oct. 2004, pp. 1133–1136.