An accurate bit-rate control for real-time MPEG video encoder

Tae-yong Kim*, Byeong-hee Roh, Jae-kyoon Kim

Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Yusong-gu, Taejon 305-701, South Korea

Received 10 July 1997

Abstract

In this paper, we present a novel framework for the forward bit-rate control method for a real-time MPEG video encoder. First, we propose bit-rate estimation and control algorithms based on the linear relationship between the actually generated bit count and the codeword count. Then, we also propose an encoder architecture for real-time MPEG video. Simulation results show that the proposed algorithm provides more accurate controllability and better picture quality than those of the MPEG TM5. Since the proposed buffer control algorithm can be managed efficiently even for small buffer sizes, it can be used for low-cost communication terminals or consumer-oriented products. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: MPEG; Forward bit-rate control; Buffer control; Scene change; Encoder architecture

1. Introduction

In real-time communications adopting MPEG video standards [5], some data losses may result in serious degradation of picture quality perceived by destination users. Excepting for problems by networks themselves, data losses may come from either the buffer overflows or the violation of traffic contract by inaccurate operation of the rate controller in the video encoder. One possible way to prevent these data losses is to prepare extra buffer spaces or bandwidths, but this makes the cost and unexpected delay increase. Since these problems are caused by an inaccurate bit-rate control, unless the accuracy of the bit-rate controller is improved, the problems still remain.

Various methods for improving the accuracy of bit-rate control have been proposed [1–4,6,8,11–14]. In rate-distortion sense, some rate control schemes using the method for finding the optimal solution such as dynamic programming [11] and Lagrangian techniques [1,7,8,12] have been proposed, but they require very large computational complexities. Schemes for minimizing differences between the actual bit count and the allocated bit count, such as feedback re-encoding [3] and binary tree search algorithm [14], require delays more than two frames and the corresponding memories or high-speed quantization and run-length entropy coding more than 3 times. Tiwari and Viscito [13] proposed a bit-rate control scheme based on a model for the picture complexity using the coding results of the randomly pre-selected macroblocks, which is for a software implementation of a non-real-time MPEG-2 video encoder. In [4,6],
parametric models experimentally determined from a representative set of video sequences were used for the estimation of bit count. In the case of non-stationary video sequences, these models represent a large mismatch in bit estimation. Cicalinì [2] proposed a feedback/feedforward controller for MPEG-2 video coding that dynamically tunes the quantization parameters by analyzing the image sequence from a psychovisual point of view. This method generates images with higher quality with respect to TM5 [9]; however, some basic parameters must be determined heuristically.

In this paper, we propose an efficient bit-rate control method for a real-time MPEG video encoder. For bit-rate estimation, we found out that the relationship between the actual bit count and the codeword count in a picture or slice can be approximated by linearity. The target bit count for each slice is adaptively assigned by the linear approximation according to its codeword count of the corresponding slice in the previous picture with same coding type. Our proposed method ensures one picture delay. Using the proposed bit-rate estimation model, we also propose a buffer control method which guarantees a constant image quality at receiver side even in scene changes, and efficient controllability even with small buffer sizes.

The rest of this paper is organized as follows. In Section 2, the proposed bit-rate estimation and forward bit-rate control methods based on the linear approximation on the relationship between the number of actually generated bits and the number of codewords are described. In Section 3, we explain our buffer control scheme and an encoder architecture for real-time processing. In Section 4, some experimental results are presented. Finally, conclusions are given in Section 5.

2. Accurate bit-rate estimation and control algorithm

2.1. Accurate bit-rate estimation model

In MPEG video coding, the quantized AC coefficients in a discrete cosine transform (DCT) block are scanned to a one-dimensional sequence by zig-zag scanning, and each pair of run-length of zeroes (i.e. run) and following non-zero value (i.e. level) is mapped to a variable length codeword in a variable-length coding (VLC) table [5,9].

We note that the generated bit count is proportional to the codeword count within a picture or slice. Let $C_i$ and $S_i$ be the codeword count and the generated bit count in the $i$th observation unit, respectively. In this paper, we use slice as the observation unit. In order to show the relationship between $C_i$ and $S_i$, we carried out the following experiments. For experiments, we use the MPEG encoder by the MPEG Software Simulation Group [10], and ‘Flower Garden’ and ‘Table Tennis’ sequences with 150 frames in CCIR601 NTSC format, respectively. In Fig. 1, each value of $S_i$ with respect to the corresponding $C_i$ for pictures is presented. As we can see from Fig. 1, the relationship between $C_i$ and $S_i$ can be approximated by linearity. Then, we have

$$S_i = z_i C_i + \beta_i,$$

where $z_i$ corresponds to the average length of a codeword for AC coefficients, and $\beta_i$ means the additional bits by codewords for non-AC coefficients such as DC components, motion vectors, and all the macroblock and slice overhead information, etc.

In order to estimate the bit-rate by using Eq. (1), we need the table of codeword count for every possible quantization parameter, \{\(C_i(q), 0 \leq i \leq N_{\text{slice}} - 1, 1 \leq q \leq 31\}\} for the picture to be coded, where $i$ is the slice index, $q$ is the quantization parameter, and $N_{\text{slice}}$ is the number of slices within a picture. Our algorithm is based on a look-ahead of an approximate bit count by a pre-analysis of DCT coefficients and model parameters \{\(z_i, \beta_i\}\} estimated from the coding results of the previous picture or slice.

The codeword count in each slice is equivalent to the number of non-zero quantized AC coefficients (NZC) in it. Let $\text{AC}(k,n)$ (1 $\leq k \leq K, 1 \leq n \leq 63$) be the $n$th AC coefficient of the $k$th DCT block of size $8 \times 8$, where $K$ is the number of DCT blocks in the $i$th slice. Let \(t(q,m_k) \neq 0\) be the non-zero threshold value for given quantization parameter $q$ where $m_k$ means the coding mode for intra or non-intra of the corresponding macroblock. Then, the absolute quantized value of $\text{AC}(k,n)$ is greater than
if $|\text{AC}_i(k,n)| \geq t(q,m_k)$. The codeword count for the $i$th slice, $C_i(q)$ is given by

$$C_i(q) = \sum_{k=1}^{K} \sum_{n=1}^{N} I_i(k,n,q),$$

where

$$I_i(k,n,q) = \begin{cases} 1, & |\text{AC}_i(k,n)| \geq t(q,m_k), \\ 0, & |\text{AC}_i(k,n)| < t(q,m_k). \end{cases}$$

Let $C(q)$ be the total codeword count for a picture, i.e., $C(q) = \sum_{i=0}^{N_{	ext{slice}}-1} C_i(q)$. It is noted that the codeword counting function can be easily implemented by using comparators and counters, as will be described in Section 3.

Fig. 2 shows the model parameters for each slice for eight consecutive pictures which corresponds to 240 slices. There are two main observations on these two parameters from Fig. 2. One is that $z_i$'s have the largest values for B-pictures, the smallest values for P-pictures, and the middle values for P-pictures. From this, we can get the fact that slices with larger $z_i$ generate smaller bit counts independent of the quantization parameter. The other is that the parameters for two successive pictures with same coding type show very similar values to each other and also the neighboring parameters for slices within a picture are very similar. These dependencies of model parameters imply that the model parameters for the current picture or slice can be estimated from the previous picture or slice with same coding type.

2.2. Adaptive slice target bit allocation method

Let $T$ and $T_i$ ($i = 0, 1, \ldots, N_{\text{slice}} - 1$) be the target bit count to be allocated to a picture and the $i$th slice in the picture, respectively. It is noted that $T$ is certainly given for a picture and $T_i$ are adaptively determined in each slice for matching target bit count $T$. The goal of the bit allocation method is to match the sum of $T_i$’s to $T$.

We let $C_i(q_r)$ be the reference codeword count, where $q_r$ is a fixed value (e.g. any value from 5 to 10), or the nearest integer value of averaged quantization parameters used in the previous picture with same coding type. In our test, $q_r$ is set to 5, and we define that $C_i^R(q_r)$ and $T_i^R$ are the sum of the reference codeword count for the remaining slices and the remaining picture target bit count after coding the slice $(i-1)$, respectively. That is,

$$C_i^R(q_r) = \sum_{j=i}^{N_{\text{slice}}-1} C_j(q_r),$$

$$T_i^R = T - \sum_{j=0}^{i-1} S_j(q_j),$$

where $S_j(q_j)$ is the actually generated bit count in the $j$th slice, which was quantized with $q_j$. It is noted that $C_i^R(q_r) = C_i(q_r), T_0^R = T$. To allocate a target bit
count to the current slice, we must estimate the model parameters for all the remaining slices to be coded because they are unknown. So, it is assumed that all the remaining slices have an equivalent $x_i$ and all $\beta_i$ for them are the same as the values for the slices of a previous picture with same coding type. Then, the target bit count for the $i$th slice $T_i$ is obtained from the remaining picture target bits by the ratio of reference codeword count. That is,

$$T_i = \left\{ T^R_i - \sum_{j=i}^{N_{max}} \frac{\beta_j}{\hat{\beta}_j} \frac{C_i(q_j)}{C^R_i(q_j)} + \beta_i, \right.\]$$

where $\hat{\beta}_j$ is the parameter value for the $j$th slice of a previous picture with the same coding type.

For selecting the right quantization parameter $q_i$ for the $i$th slice, brute-force exhaustive search or full-search scheme [14] may be used for exact calculation, but these are impractical due to heavy computational complexities. In our scheme, $q_i$'s are adaptively selected by comparing the actually generated codeword count $\{C_i(q_i)\}$ and the estimated codeword count $\tilde{C}_i(T_i)$. That is,

$$q_i = \arg\min_q C_i(q) - \tilde{C}_i(T_i),$$

where

$$\tilde{C}_i(T_i) = \frac{T_i - \beta_i}{x_i}.$$
codeword count, the generated bit count for AC coefficients, and the total number of generated bits, respectively. Then, from Eq. (1), we have

\[ x_0 = Z / C \quad \text{and} \quad \beta_0 = (S - Z) / N_{\text{slices}}, \]  

(9)

where \( N_{\text{slices}} \) is the number of slices within a picture.

As we have seen from Fig. 2, though the model parameters between two neighboring slices have very similar values, they have long-term variations within a picture. That is, there is no abrupt change of values of the parameters and the parameters of the current slice follow the trends of those of the previous slices. Thus, the model parameters are calculated as follows:

\[ x_i = Z_{i-1}(q_{i-1}) / C_{i-1}(q_{i-1}), \]  

(10)

\[ \beta_i = S_{i-1}(q_{i-1}) - Z_{i-1}(q_{i-1}), \]  

(11)

where \( i = 1, \ldots, N_{\text{slices}} - 1 \) and \( \{C_{i-1}(q_{i-1}), Z_{i-1}(q_{i-1}), S_{i-1}(q_{i-1})\} \) are the measured codeword count, the generated bit count by AC coefficients, and the total generated bit count by actual coding with quantization parameter \( q_{i-1} \) for the \( (i - 1) \)th slice, respectively.

The above procedure for the proposed bit-rate control algorithm is summarized as follows:

1. Through the pre-analysis of DCT coefficients, calculate the table of codeword counts \( \{C_i(q), 0 \leq i \leq N_{\text{slices}} - 1, 1 \leq q \leq 31\} \).
2. Calculate the initial model parameters \( \{x_0, \beta_0\} \) using Eq. (9) and \( C, Z \) and \( S \) of the previous picture with the same coding type.
3. Calculate the allocated bit count \( T_i \) by using Eq. (6).
4. Select the quantization parameter \( q_i \) as in Eq. (7).
5. Quantize and encode the \( i \)th slice with \( q_i \).
6. By using the coding results \( C_i(q_i), Z_i(q_i) \) and \( S_i(q_i) \), and Eqs. (4) and (5), update the remaining picture target bit count \( T_i^p \) and the remaining reference codeword count \( C_i^p(q_i) \).
7. Calculate the model parameters \( \{x_{i+1}, \beta_{i+1}\} \) for the next slice by using Eqs. (10) and (11).
8. Let \( i = i + 1 \). If \( i < N_{\text{slices}} \), go to 3, otherwise go to 1 for the next picture.

The adaptation method described above is called the ‘slice-level adaptation’ scheme because the model parameters are updated in every slice. Since model parameters \( \{x_n, \beta_n\} \) in neighboring pictures with the same coding type are very similar within a scene, in the case of sequences with the homogeneous scenes, \( \{x_0, \beta_0\} \) can be used for \( \{x_n, \beta_n\} \), which is called the ‘picture-level adaptation’ scheme. The ‘picture-level adaptation’ scheme is more simple than the ‘slice-level adaptation’ scheme, but the performances are reverse.

3. Efficient buffer control scheme and its implementation

In the TM5 rate control algorithm [9], a target bit count for the current picture is allocated based on the ratio of complexity measure of the previous coded pictures. This may not be proper for a scene change that has an abrupt change in complexity. In addition, since the quantization parameter for a macroblock in TM5 is determined by the virtual buffer fullness only, there exists a possibility not to guarantee for preventing buffer from being overflow or underflow. These result in non-uniform distortion over picture sequences, which leads to serious degradation of visual quality perceived by the destination users. In this section, we propose a buffer control algorithm for overcoming the problems, which is so accurate that it can even efficiently manage a small buffer.

3.1. Modified target bit allocation

In TM5, target bit counts \( T_p, T_r \) and \( T_b \) for I-, P- and B-pictures are determined by the ratio of complexity measures of the previously coded I-, P- and B-pictures, i.e. \( X_p, X_r \) and \( X_b \), respectively [9]. When the characteristics of subsequent pictures are significantly changed due to scene change, the target bit count may be inadequately allocated, which leads to degradation of visual quality. For example, if the generated bit count is larger than the allocated bit count, the virtual buffer becomes filled fast, which leads to continuous increase in the quantization parameter. For the next picture, the quantization parameter is too large because the buffer is filled by the previous picture. As a result,

\[ X_r = \frac{\text{picture target bit count}}{\text{picture reference codeword count}} \]
the picture quality becomes inconsistent over a picture and over a sequence.

To compensate this problem, we use the complexity estimated from the current picture for the target bit allocation. Let \( Q_{\text{avg}} \) be the average quantization parameter of the previous picture.

The number of bits estimated from the current picture with \( Q_{\text{avg}} \) is

\[
S_{\text{est}} = \alpha_0 C(Q_{\text{avg}}) + \beta_0, \tag{12}
\]

where \( C(Q_{\text{avg}}) \) is the codeword count of the current picture with non-integer \( Q_{\text{avg}} \), which can be calculated from the table of codeword count, \( C(q) \), \( q = 1, 2, \ldots, 31 \), for the current picture.

Then the estimated complexity of the current picture becomes [9]

\[
X_{\text{est}} = Q_{\text{avg}} S_{\text{est}}. \tag{13}
\]

In the equations for the target bit counts \( T_I, T_P, T_B \) and the complexity measures \( X_I, X_P, X_B \) are replaced by \( X_{\text{est}} \) and the other complexities are calculated from the previous corresponding pictures. That is,

\[
T_I = \max \left\{ \frac{R}{1 + (N_P X_P / K_P X_{\text{est}}) + N_B X_B / K_B X_{\text{est}}}, \frac{\text{bit rate}}{8 \times \text{picture rate}} \right\}, \tag{14}
\]

\[
T_P = \max \left\{ \frac{R}{N_P + N_B (K_P X_P / K_B X_{\text{est}})}, \frac{\text{bit rate}}{8 \times \text{picture rate}} \right\}, \tag{15}
\]

\[
T_B = \max \left\{ \frac{R}{N_B + N_P (K_B X_P / K_P X_{\text{est}})}, \frac{\text{bit rate}}{8 \times \text{picture rate}} \right\}, \tag{16}
\]

where \( K_P \) and \( K_B \) are constants, \( R \) is the remaining bit count allocated to the current GOP, and \( N_P \) and \( N_B \) are the number of P- and B-pictures remaining in the current GOP, respectively. In this modified scheme, the estimated complexity of the current picture is used for the target bit allocation for the current picture.

3.2. Coding for scene change

After a scene change occurs, the first picture is always the I- or P-picture. We define a scene-changed P-picture as the P-picture with the number of macroblocks with intracoding mode greater than half the total number of macroblocks [6]. Similarly, a scene-changed B-picture is defined as the B-picture with the number of macroblocks with backward or forward predictive mode greater than half the total number of macroblocks. Therefore, the scene-changed P- and B-pictures need bit counts similar to I- and P-pictures, respectively.

If a scene change is detected, the following operations are executed:

1. The remaining pictures in the current GOP and the next GOP are merged in a new GOP. As a result, the new GOP has a larger size than the normal GOP.
2. The scene-changed P-picture is coded as the I-picture.
3. The target number of bits for the scene-changed B-picture are allocated by the same method as for the P-picture.
4. The I-picture of the merged GOP is coded as the P-picture.

3.3. Prevention of buffer overflow/underflow

When the target bit counts are allocated to the I-picture or scene-changed pictures, the buffer overflow should be considered. Also in the B-picture, the buffer underflow should be considered. Let \( B_{\text{max}} \) be the encoder buffer size and \( B_{\text{cur}} \) be the current buffer state. Then, the maximum bit count \( T_{\text{upper}} \) and the minimum bit count \( T_{\text{lower}} \) for the current picture are as follows:

\[
T_{\text{upper}} = B_{\text{max}} + \text{bits per picture} - B_{\text{cur}}, \tag{17}
\]

\[
T_{\text{lower}} = \max \{0, \text{bits per picture} - B_{\text{cur}}\}, \tag{18}
\]

where \( \text{bits per picture} = \text{bit rate} / \text{picture rate} [9] \).

In order to reflect the buffer status, the target bit count \( T \) for the picture computed according to global complexity is adjusted forcibly as
if \[ T > aT_{\text{upper}}, \quad T = aT_{\text{upper}}, \] (19)
else if \[ T \leq bT_{\text{lower}}, \quad T = bT_{\text{lower}}, \] (20)

where \( a \) and \( b \) are constants reflecting the accuracy of the bit-rate control algorithm to prevent a buffer from overflowing or underflowing caused by a discrepancy between the target bit count and the actual bit count. Constants \( a \) and \( b \) are determined by the maximum deviation of I- and B-pictures, respectively. In our simulations, 0.95 and 1.2 are used for \( a \) and \( b \), respectively.

3.4. Encoder architecture for real-time processing

Fig. 3 shows an example implementation of the proposed rate control algorithm with picture memories within the coding loop such as frame reordering memory (FRM), reconstructed picture memory (RPM), FM1 (memory 1 for frame delay), FM2 and FM3. FRM is used for B-picture coding, and RPM for motion compensation. FM1, FM2 and FM3 are different from TM5. Fig. 4 shows the detailed block diagram of the NZC block used for the pre-analysis of DCT coefficients, which consists of a comparator, counter and FD (register for frame delay). It is important to note that the output bitstream is fully compliant with the MPEG standard.

An example timing diagram when a video sequence is coded without B-pictures is shown in Fig. 5. It is noted that the picture is partitioned by \( MN \) macroblocks and the motion vector search range is \( \pm S_x \) macroblocks in horizontal direction.
and $\pm S_y$ macroblocks in vertical direction. At the starting time of picture $i$ ($t_1$ in Fig. 5), FM1 has the weighted DCT coefficients of the ($i-1$)th picture and FM2 has the motion compensated data of the ($i-1$)th picture. As shown in Fig. 3, the data of the $i$th picture applied to the FRM block are delayed until $s_y$ slices of the ($i-1$)th picture are reconstructed and stored in RPM. Let the period for this processing be $s_y$-slice-delay. That is, at $t_2$ (from Fig. 5) the motion compensated data of the first macroblock of the $i$th picture can be obtained from RPM. The difference signals between the data of the $i$th picture stored at FRM and the motion-compensated data from MC are transformed in the DCT block. The transformed DCT coefficients are scaled by the weighting matrix [5] in the W block, and the results are fed into the FM1 for ensuring a certain delay and into the NZC block for pre-analysis, simultaneously. As shown in Fig. 4, the NZC block calculates the table of codeword counts 

\[ \{ C_i(q), 0 \leq i \leq N_{\text{slices}} - 1, 1 \leq q \leq 31 \} \]

Note that $C(q) = \sum C_i(q)$. The RC (rate control) block is the realization of the proposed forward bit-rate control algorithm for selecting the quantization parameter using \{ $C_i(q)$ \} and \{ $C(q)$ \} from the NZC block as described in the previous sections, and then feeds it into the Q block for the calculation of the quantization parameter.

The coding of the ($i-1$)th picture delayed in FM1 should be started at the starting point $t_3$ (in Fig. 5) of the $i$th picture. After $s_y$ slices of the ($i-1$)th pictures are coded, the processing of the $i$th picture delayed in SM1 is started. As a result, the coding of the ($i-1$)th picture delayed in FM1 is completed at $t_3$ (in Fig. 5) earlier than the starting point of the ($i+1$)th picture. The period for this processing is the same as $s_y$-slice-delay. That is, the proposed video encoder requires higher processing speed than the TM5 encoder. For example, when the picture size is $720 \times 480$ and the motion search range is $\pm 63$ pixels in horizontal and vertical directions (corresponding to four slices in $16 \times 16$ macroblock size), it requires 34/30 times higher speed than that of the TM5 MPEG encoder. If the search range is equal to the whole picture size, as an extreme case, it requires twice the processing speed of the TM5 MPEG encoder.

To encode the B-picture in TM5, at least three frame memories are required for frame reordering (corresponding to FRM) and two reconstructed picture memories are used for motion compensation (corresponding to RPM). In the proposed video encoder, additional memories for FM1 and FM3 are required. The sum of FM2 and RPM is the same as the size of the motion compensation memory for TM5. Also, the RC blocks are similar to each other and the decoder structure is the same because the proposed encoder complies with the MPEG standard. As described above, our system requires higher processing speed than TM5. These additional complexities are costs for real-time buffer control without overflow/underflow and with consistent picture quality. The stable buffer control allows to use a smaller size buffer.

4. Experimental results

4.1. Accuracy of bit-rate control algorithm

For experiments, ‘Flower Garden’ and ‘Table Tennis’ sequences with 150 frames each in CCIR601 format are used. For MPEG coding, GOP structure $(N, M) = (12, 2)$ is used. The performance of our method is compared with the results of the TM5 rate control method and the ‘full search scheme’. For TM5 rate control, the MPEG-2 encoder program provided by the MPEG Simulation Group [10] is used. The ‘full search scheme’ is as follows. First, we get the actually coded bit counts \{ $S_i(1), S_i(2), \ldots, S_i(31)$ \} by using all possible quantization parameters. Then we select the nearest $S_i(q)$ to the slice target bit count $T_i$ [10].

Fig. 6 shows the encoding results for the two test sequences, in which 440, 260 and 110 kbits are given as target bit counts for I-, P- and B-pictures, respectively, for making constant bit-rate of 6 Mbit/s. The generated bit counts by the proposed method (‘slice-level adaptation’ as in Fig. 6(a) and (c)) are much the same as the target bit counts (the dotted lines). While the bit counts by ‘TM5’ (as in Fig. 6(b) and (d)) are inconsistent with the target bit counts. In Fig. 7, we present the deviations between the target bit count and the actually generated bit count for various methods. Note that the ‘full
search scheme' shows the best accuracy, and the proposed 'slice level adaptation' and 'picture level adaptation' schemes show much better accuracy than TM5.

For the 'Flower Garden' sequence in which there is no scene change, peak-to-peak values of deviations for 'slice level adaptation' and 'picture level adaptation' are very similar to each other, while they are about 40 times of 'TM5'. However, for the 'Table Tennis' sequence that has scene changes, peak-to-peak value of deviations for 'slice level adaptation' is 1/2 times 'picture-level adaptation' and 1/50 times 'TM5'. From this, we can see that the 'slice-level adaptation' can control bit-rates more accurately than 'picture-level adaptation' and 'TM5' in case of scene changes.

The accuracy of each method can be compared by the maximum and the average of the normalized absolute differences (NAD) between the actually generated bit count and the target bit count. The NAD is defined as

\[
\text{NAD} = \frac{|S_f - T_f|}{T_f} \times 100, \tag{21}
\]

where \(S_f\) and \(T_f\) are the actual bit count and the target bit count for picture \(f\). The average and maximum NAD for Fig. 7 are summarized in Table 1. The average NAD for 'slice-level adaptation' and 'picture-level adaptation' are much less than 'TM5'. Also from the NAD point of view, 'slice-level adaptation' shows a better performance than 'picture-level adaptation'. It means that the larger NAD is as in 'TM5', the larger extra buffer or bandwidth may be required for the prevention of data loss.

4.2. Efficiency of buffer control algorithm

Two buffer sizes are applied for the experiments of the proposed buffer control algorithm; 1.8 Mbits
of the allowable maximum buffer size for MPEG-2 MP@ML [5] and 320 kbits as a small buffer size.

As test sequences, ‘Table Tennis’ of 150 frames and ‘Edited’ of 100 frames in CCIR601 format are used. The ‘Edited’ sequence is constructed by concatenating the first 30 frames of ‘Flower Garden’, first 35 frames of ‘Susi’, and first 35 frames of ‘Football’ sequentially. All the test sequences have two scene changes. The first scene change of ‘Table Tennis’ switches to a scene with similar complexity. The first scene change of ‘Edited’ switches from high complexity to low complexity, and all the second scene changes of two sequences switch from low complexity to high complexity.

Fig. 8 shows the PSNRs and the buffer fullness when the GOP structure \((N,M) = (12,2)\) and \(B_{\text{max}} = 1.8\) Mbits are used. Fig. 8(a) illustrates that the proposed algorithm shows stable PSNRs even when scene changes occur around frames 67 and 97, while the PSNR curve of TM5 fluctuates largely around the scene change points. This phenomenon can be explained by the buffer fullness curves as shown in Fig. 8(b). Since, in the proposed method, the scene-changed P-pictures (frames 67 and 97) are coded as I-pictures and the bit allocation scheme of the next B-pictures (frames 68 and 98) are the same as P-pictures, the buffer fullness curve of the proposed method shows a continuously increasing pattern around those pictures, whereas that of TM5 decreases continuously around the scene change. As a result, the PSNR of the proposed methods are maintained relatively high, over 35 dB. While, in TM5, the scene-changed pictures have relatively small bit-rates, which results in the drop of PSNRs in the scene changed and the following pictures.

In order to measure temporal quality degradations, we use the \textit{Average PSNR} and the \textit{PSNR Variation}. The \textit{Average PSNR} is the averaged value of the PSNRs of the frames within the interesting
Table 1
The comparisons of average and maximum NADs among several schemes

<table>
<thead>
<tr>
<th>Image sequences</th>
<th>Average NAD</th>
<th>Maximum NAD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>‘Slice’ &amp; ‘Picture’</td>
<td>‘TM5’</td>
</tr>
<tr>
<td>Table I</td>
<td>0.419 &amp; 0.623</td>
<td>2.773 &amp; 0.156</td>
</tr>
<tr>
<td>Tennis P</td>
<td>0.188 &amp; 0.245</td>
<td>3.936 &amp; 0.132</td>
</tr>
<tr>
<td>Tennis B</td>
<td>0.351 &amp; 0.377</td>
<td>6.216 &amp; 0.288</td>
</tr>
<tr>
<td>Flower I</td>
<td>0.161 &amp; 0.067</td>
<td>0.769 &amp; 0.039</td>
</tr>
<tr>
<td>Flower B</td>
<td>0.184 &amp; 0.271</td>
<td>4.741 &amp; 0.151</td>
</tr>
<tr>
<td>Garden P</td>
<td>0.302 &amp; 0.287</td>
<td>9.233 &amp; 0.187</td>
</tr>
<tr>
<td>Garden B</td>
<td>0.351 &amp; 0.377</td>
<td>6.216 &amp; 0.288</td>
</tr>
</tbody>
</table>


Fig. 8. Using $B_{\text{max}} = 1.8$ Mbits and GOP structure $(12,2)$, the PSNR and buffer fullness for ‘Table Tennis’ with 6 Mbits/s.

period and the PSNR Variation is defined as

$$\text{PSNR Variation} = \frac{1}{L - F + 1} \sum_{f = F}^{L} |\text{PSNR}_{f} - \text{PSNR}_{f-1}|.$$  \[22\]

where $F$ is the first frame number, $L$ is the last frame number, and PSNR$_{f}$ means the PSNR of frame $f$. The results of Fig. 8 are summarized in Table 2. The ‘Whole Sequence’ represents the frames corresponding to $F = 60$ and $L = 120$, including all scene changes. While the period of

Table 2
The Average PSNR and the PSNR Variation for the ‘Proposed’ rate control algorithm and TM5 scheme; the buffer size is 1.8 Mbits

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Average PSNR (dB)</th>
<th>6</th>
<th>12</th>
<th>PSNR Variation (dB)</th>
<th>6</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coding bit-rate (Mbits/s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Whole</td>
<td>Proposed</td>
<td>35.826</td>
<td>38.629</td>
<td>0.287</td>
<td>6</td>
<td>0.853</td>
</tr>
<tr>
<td>Sequence</td>
<td>TM5</td>
<td>35.113</td>
<td>38.053</td>
<td>0.833</td>
<td>12</td>
<td>1.782</td>
</tr>
<tr>
<td>First Scene Change</td>
<td>Proposed</td>
<td>35.988</td>
<td>38.836</td>
<td>0.390</td>
<td>6</td>
<td>1.024</td>
</tr>
<tr>
<td>Scene Change</td>
<td>TM5</td>
<td>34.865</td>
<td>37.889</td>
<td>1.026</td>
<td>12</td>
<td>2.164</td>
</tr>
<tr>
<td>Second Scene Change</td>
<td>Proposed</td>
<td>35.198</td>
<td>38.089</td>
<td>0.209</td>
<td>6</td>
<td>0.769</td>
</tr>
<tr>
<td>Scene Change</td>
<td>TM5</td>
<td>33.576</td>
<td>36.709</td>
<td>0.925</td>
<td>12</td>
<td>1.884</td>
</tr>
</tbody>
</table>
‘First Scene Change’ is the consecutive 12 frames starting from the first scene changed P-picture (i.e. \( F = 67 \) and \( L = 78 \)), and the period of ‘Second Scene Change’ is the consecutive 12 frames starting from the second scene changed P-picture (i.e. \( F = 97 \) and \( L = 108 \)). The Average PSNRs of the proposed algorithm are 0.5–0.7 dB higher for ‘Whole Sequence’, 1 dB higher for ‘First Scene Change’, and 1.3–1.5 dB higher for ‘Second Scene Change’ comparing it with TM5. For ‘Whole
Sequence’, the PSNR Variation of the proposed algorithm is about 1/3–1/2 of TM5. Especially, the PSNR Variation of the proposed method is much smaller than those of TM5 for scene changes. This means that the proposed method can maintain consistent picture quality even when the scene changes.

Fig. 9 shows the PSNR and buffer fullness when a small buffer size of 320 kbits is used. Although the buffer size becomes smaller than that in Fig. 8, the bit-rates generated by TM5 (the dotted line in Fig. 9(b)) are the same as in Fig. 8 (the dotted line in Fig. 8(b)). As a result, buffer overflows occur several times as shown in Fig. 9(b). These buffer overflows can be prevented by using a larger buffer or by lowering the bit-rate. In the proposed rate control algorithm, the buffer overflows never occur as shown in Fig. 9(b). In the scene-changed pictures and some I-pictures (i.e. frames 67, 83, 95, 97 and 107), the bit allocation is limited by the buffer status. Additionally, the proposed algorithm controls strictly the bit-rate so that the generated bit-rate becomes similar to the allocated bit-rate.

Fig. 10 shows the coding results for the ‘Edited’ sequence with GOP structure \((N, M) = (12, 2)\) and constant bit-rate 12 Mbits/s. The special points of the ‘Edited’ sequence is that the first 30 frames of the ‘Flower Garden’ have high complexity, the next 35 frames of ‘Susi’ have very low complexity, and the last 35 frames of ‘Football’ have high complexity. When the buffer size is 1.8 Mbits (Fig. 10(a) and (c)), TM5 shows very large fluctuations in PSNR around the scene changes and I-pictures (frames 30, 31, 47, 59, 65, 66). When the buffer size is as small as 320 kbits (Fig. 10(b) and (d)), buffer overflows occur in these frames, seriously. However, in the proposed rate control algorithm buffer overflows do not occur because the bit-rate is determined by considering both the buffer status and complexity.

5. Conclusions

In the bit-rate control of a video encoder, it is important to allocate the target bit count to each picture according to its complexity, as well as to encode to match the allocated number of bits as accurately as possible. In this paper, we proposed an accurate bit-rate control algorithm based on the linear relationship between the codeword count and the actual bit count. The model parameters for bit-rate estimation can be calculated in real time from the previous coding results. Also, the actual bit count by our scheme nearly matches the target bit count. In addition, we proposed an efficient buffer control algorithm, which can not only maintain a consistent visual quality even for scene changes, but also control a buffer efficiently even for small buffers. The proposed buffer control algorithm operates in real time with a constant picture delay while conforming to the MPEG standard.

The high accuracy of the proposed bit-rate control is accomplished by a pre-analysis with a picture delay. For real-time applications, the encoder requires some additional circuits, and higher processing speed than the TM5 MPEG encoder. These extra costs are only marginal for implementation of a real-time processor and worth paying for accurate rate control. The proposed methods can be used for various purposes such as a constant bit-rate MPEG video encoder with a small buffer, video encoder control for conformance of the negotiated traffic descriptors in variable bit-rate transmission, or MPEG video recording and editing.

References


