IEICE TRANS. FUNDAMENTALS, VOL.E85–A, NO.6 JUNE 2002

PAPER

New Rate Control Method with Minimum Skipped Frames for Very Low Delay in H.263+ Codec

Trio ADIONO[†a)], ***Nonmember***, **Tsuyoshi ISSHIKI**[†], ***Regular Member***, **Chawalit HONSAWEK**[†], ***Nonmember***, **Kazuhito ITO**[††], **Dongju LI**[†], ***and*** **Hiroaki KUNIEDA**[†], ***Regular Members***

SUMMARY A new H.263+ rate control method that has very low encoder-decoder delay, small buffer and low computational complexity for hardware realization is proposed in this paper. This method focuses on producing low encoder-decoder delay in order to solve the lip synchronization problem. Low encoder-decoder delay is achieved by improving target bit rate achievement and reducing processing delay. The target bit rate achievement is improved by allocating an optimum frame encoding bits, and employing a new adaptive threshold of zero vector motion estimation. The processing delay is reduced by simplifying quantization parameter computation, applying a new non-zero coefficient distortion measure and utilizing previous frame information in current frame encoding. The simulation results indicate very large number skipped frames reduction in comparison with the test model TMN8. There were 80 skipped frames less than that of TMN8 within a 380 frame sequence during encoding of a very high movement video sequence. The 27 kbps target bit rate is achieved with insignificant difference for various types of video sequences. The simulation results also show that our method successfully allocates encoding bits, maintains small data at the encoder buffer and avoids buffer from overflow and underflow.

key words: *H.263+, rate control, optimum bit allocation, low encoder-decoder delay, lip synchronization*

1. Introduction

In the real-time application of H.263+ video coding, the delay between an encoder and a decoder needs to be minimized to avoid low image quality and large memory requirement. Many skipped frames occur due to a large encoder-decoder delay. The frame skip leads to a low encoding frame rate and a large frame difference. Both can decrease image quality and cause serious lip synchronization problem. In this condition, a large amount of memory is also required to buffer remaining bits at an encoder.

The encoder-decoder delay increases mainly because of false target bit rate achievement and processing delay. Most of the conventional methods in general only deal with the first cause of delay [1]–[7] and consider the delay only from remaining bits at the encoder buffer. To avoid a large delay, conventional methods [7]–[9] allocate fixed frame encoding bits from the channel bit rate and the encoding frame rate. If the remaining bits at an encoder buffer exceed a defined threshold, frame skipping will be applied. As a consequence, for a very low bit rate channel and a high encoding frame rate application such as a PSTN video phone, an encoder will produce a lot of skipped frames and low image quality.

In our proposed method, we consider both processing delay and target bit rate achievement. For target bit rate achievement, different from those conventional methods, an allocation of frame encoding bits and a frame skipping threshold are computed by taking into account various parameters causing the encoder-decoder delay in real hardware implementation. The objective is to significantly minimize the encoder-decoder delay and underflow bits. Instead of applying a fixed threshold for zero vector motion estimation, we employ an adaptive threshold, which is a function of quantization parameter, in order to control motion encoding bits. This method effectively improves target bit rate achievement by preventing an encoder buffer from a large increase of motion encoding bits, particularly, during the encoding of a high-movement video sequence.

In this new method, the complexity of macroblock quantization parameter computation is dramatically reduced by utilizing a number of non-zero DCT coefficients as a distortion measure and simple bits comparison operations. Moreover, we use previous frame image information instead of the current one to compute current frame rate control parameters. Both of these computations reduce the processing delay.

2. Overview of Video Coding System

2.1 System Block Diagram

In this proposed video coding system, the standard H.263+ encoder block diagram is modified as shown in Fig. 1. The input frame is divided into macroblocks, each of which covers an area of 16×16 pixels. Motion-compensated prediction is carried out for each mac-

Manuscript received May 2, 2001.
Manuscript revised October 10, 2001.
Final manuscript received February 26, 2002.

[†]The authors are with the Department of Communications and Integrated Systems, Tokyo Institute of Technology, Tokyo, 152-8552 Japan.
[††]The author is with the Department of Electrical and Electronic Systems Engineering, Saitama University, Saitama-shi, 338-8570 Japan.
a) E-mail: tadiono@vlsi.ss.titech.ac.jp

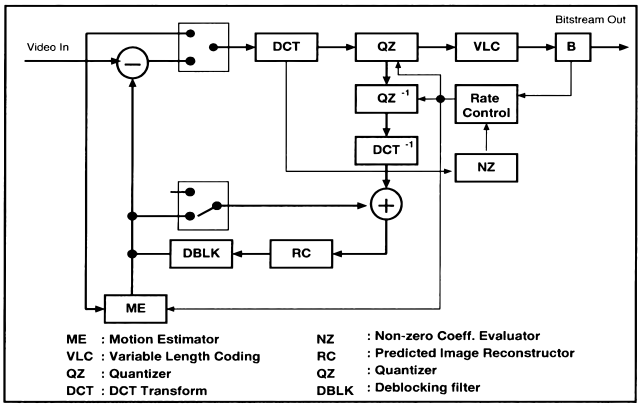


Fig. 1 Block diagram of the proposed video coding system.

roblock. Motion estimation is used to remove temporal redundancies by finding the best match for the current macroblock within a search region of the reference frame. As the matching criterion, sum of absolute difference (SAD) representation is used. Specifically, for zero motion vector, the adaptive threshold is applied.

To improve system performance, we employ full search and half-pel motion estimation (**ME**). DCT algorithm is used for residual coding. DCT coefficients are then quantized by **QZ** using quantization parameter Q, which is computed by rate control. Q computation utilizes the number of non-zero coefficients of the previous macroblock, which is measured by the non-zero coefficient evaluator **NZ**. Q is also used in **ME** to adjust the zero motion vector estimation threshold. Using residual information produced by inverse quantized $\mathbf{QZ}^{-1}$ and motion vector information generated by **ME**, the predicted image is reconstructed by **RC**. The blocking effect appearing in the reconstructed image is eliminated by applying the deblocking filter **DBLK**.

2.2 Rate Control Flow

The flow of the proposed rate control consisting of frame and macroblock level processing is described in Fig. 2. The frame level rate control is used mainly to estimate bits allocation B_{TE} of target frame encoding, to compute global quantization parameter Q_G and to determine frame skipping. The macroblock level rate control is then applied to compute macroblock quantization parameter Q precisely from the estimated Q_G in order to achieve allocated B_{TE} bits.

3. Frame Level Rate Control

The optimum value of the frame encoding bits B_{BO} is set to prevent an encoder from producing underflow bits or delay larger than the user defined maximum delay D_{max}. B_{BO} is allocated by considering various causes of delay in real-hardware implementation.

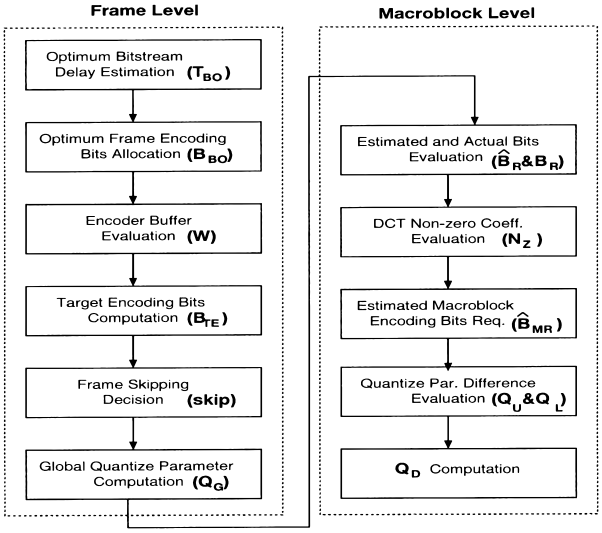


Fig. 2 Proposed low delay rate control flow.

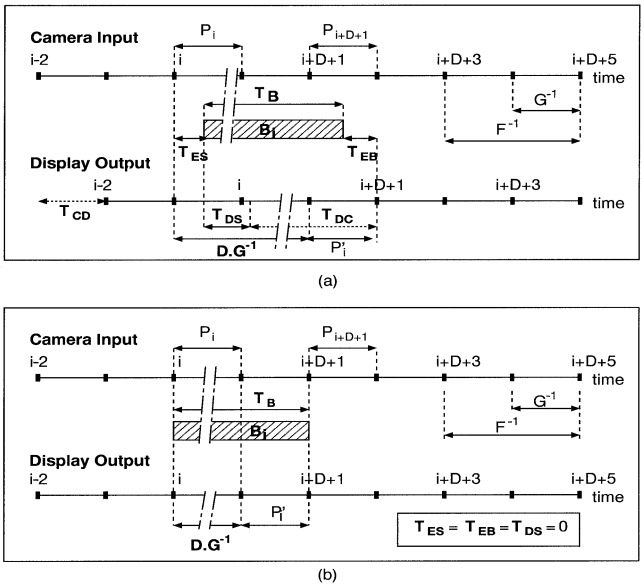


Fig. 3 Encoder-decoder delay timing diagram under the condition that the encoding frame interval is two ($G/F = 2$), and the P frame type encoding mode. D: encoder-decoder delay (in *frames*) G: camera input frame rate (in *frame/sec*), F: encoding frame rate (in *frame/sec*),i: camera input frame number which is counted in period of $(1/G)$, P_i: ith encoding frame at encoder, P_i': decoding frame of P_i, B_i: P_i's frame bitstream (in *bits*), T_{CD}: time difference between camera input and display output frame period (in *sec*), T_{DC}: decoding time of P_i' frame (in *sec*). (a) Various delays, constructing the encoder-decoder delay D. (b) Simplified encoder-decoder delay D.

3.1 Encoder Decoder Delay

There are various delays occurring between a camera's input at the encoder and its display at the decoder, as shown in Fig. 3(a).

The first delay is encoding-delay T_{ES}. T_{ES} is the

sum of the buffering time, which is time required from the first macroblock data received until it is ready to be encoded, and the encoding time, which is time required from encoding of the first macroblock until it is ready to be transferred.

The second delay is decoding-delay T_{DS}. This delay is the sum total of buffering time and decoding time. This buffering time is the time needed to buffer bitstream data from the encoder until it is ready to be decoded, while the decoding time is the time required from the decoding of the first macroblock bitstream until it is ready to be displayed.

The third delay is the amount of time to accomplish decoding of the last received bitstream data and to display T_{EB}.

The amount of time to transfer bitstream T_B, which varies according to frame's distortion, is defined as the fourth delay.

Referring to the above delays, the encoder-decoder delay D can be defined as the time difference between the time when the encoder receives image data P_i from the camera and the time when the decoder displays it P', as illustrated in Fig. 3 for P frame type encoding mode and Fig. 4 for PB frame type encoding mode. For P frame type encoding mode (number of encoding frames for each bitstream $C_c = 1$), D is defined as:

$$D = \left(T_B - \frac{1}{G} + T_{ES} + T_{DS} - T_{EB}\right) \times G \qquad (1)$$

where D is in unit of *frames*. T_{ES}, T_{DS}, T_{EB} and T_B are in *sec* and G is in *frame/sec*.

In actual conditions, T_{ES} and T_{DS} depend on the data scheduling and the processing speed of the system. To simplify the analysis of delay for bits allocation computation, we assume $T_{ES} = 0$ and $T_{DS} = 0$; consequently, $T_{EB} = 0$. Under these assumptions, the timing diagram in Fig. 3(a) is simplified to Fig. 3(b), where D becomes:

$$D \simeq (T_B \times G - 1). \qquad (2)$$

For the PB frame type encoding mode ($C_c > 1$), as shown in Fig. 4, the bitstream cannot be transferred until $C_c - 1$ frame(s) are received. Hence, D becomes:

$$D \simeq (T_B \times G - 1) + (C_c - 1) \times \frac{G}{F}. \qquad (3)$$

Both Eq. (2) and Eq. (3) will be used in B_{BO} bits allocation (see next section).

3.2 Frame Encoding Bits Allocation

The optimum number of bits of current frame encoding B_{BO}, is set by considering the earliest delay T_{BE} and the latest possible delay T_{BL} of bitstream transfer.

T_{BE} is estimated by assigning the time for bitstream transfer in such a way that it can prevent the encoder buffer from underflow. In this condition, the bitstream transfer should use available bandwidth and must be finished by the starting time of next frame's bitstream, as illustrated in Fig. 5. Therefore, T_{BE} becomes a function of current C_c (see Eq. (2) and Eq. (3)) and can be defined as:

$$T_{BE} = \frac{C_c}{G}. \qquad (4)$$

where $C_c = 1$ for P frame type and $C_c = 2$ for PB frame

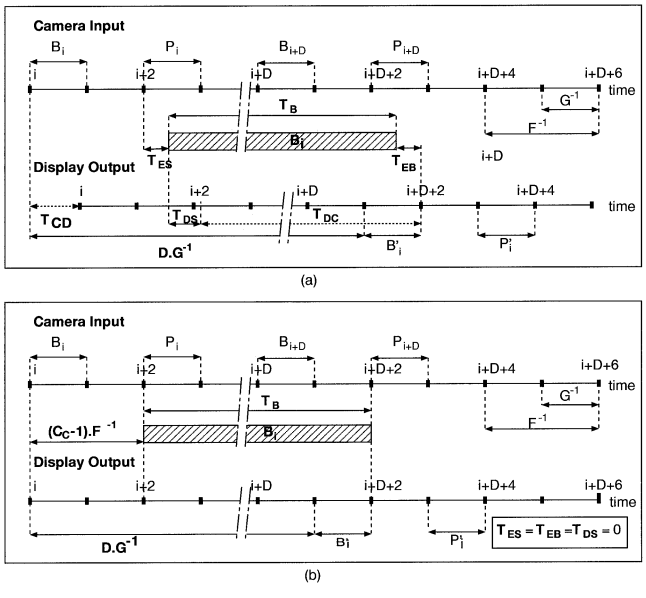


Fig. 4 Encoder-decoder delay D under the condition that the frame interval is two ($G/F = 2$) and $C_c = 2$ (PB frame type encoding mode). C_c: number of frames encoded into one bitstream B_i (in *frames*). (a) Various delays, comprising the encoder-decoder delay D. (b) Simplified encoder-decoder delay D.

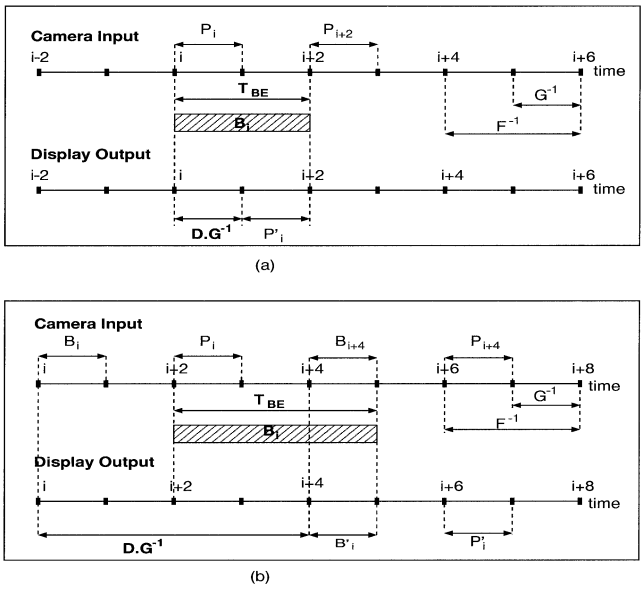


Fig. 5 Estimated required time of bitstream transfer in the earliest delay condition T_{BE} (in *sec*). The condition is $G/F = 2$. (a) P frame mode. (b) PB frame mode.

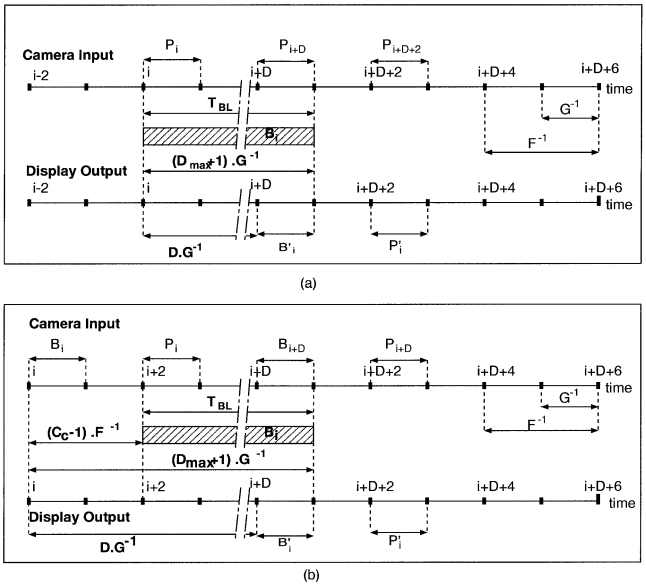


Fig. 6 Estimated required time of bitstream transfer in latest delay condition T_{BL} (in *sec*). The condition are $D = D_{max}$ and $G/F = 2$. (a) P frame mode ($C_c = 1$). (b) PB frame mode ($C_c = 2$).

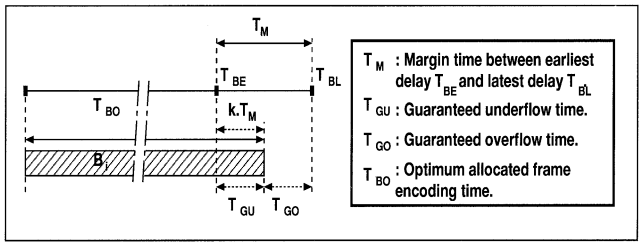


Fig. 7 Optimum allocated bitstream transfer time T_{BO}.

type (as described in annex M of H.263+ standard).

T_{BL} is estimated by assigning the time for bitstream transfer in order to handle the biggest possible bitstream, subject to delay limitation D_{max}. This can be satisfied by increasing the bitstream transfer time from T_{BE} under consideration that D is less than D_{max}. This condition is described in Fig. 6. It is shown that for displaying picture P_i at time D_{max}/G, the latest of T_{BL} is $(D_{max} + 1)/G$ (see Eq. (2) and Eq. (3)). In the case of $C_c > 1$, the starting time of bitstream transfer is $(C_c - 1)/F$. Hence, T_{BL} can be computed as:

$$T_{BL} = \frac{(D_{max} + 1)}{G} - \frac{(C_c - 1)}{F}. \tag{5}$$

Moreover, to avoid overflow and underflow conditions that depend on the target frame encoding bits achievement, the estimated optimum time for current frame bitstream T_{BO} should be longer than T_{BE} and shorter than T_{BL} as $T_{BE} \leq T_{BO} \leq T_{BL}$ (Fig. 7). Hence, T_{BO} can be computed by applying the weighting factor k to $T_{BL} - T_{BE}$, as:

$$T_{BO} = k \times (T_{BL} - T_{BE}) + T_{BE}. \tag{6}$$

where k, $0 < k < 1$.

In the rest of this section, we explore how k is determined. Recall from Eq. (4) and Eq. (5) that there is a time margin $T_M = T_{BL} - T_{BE}$ which can be adjusted using D_{max}. T_M represents time to compensate underflow and overflow resulting from a false target frame encoding bits achievement. As illustrated in Fig. 7, if the produced frame encoding bits are smaller than T_{BO}, and the difference is smaller than T_{GU}, the encoder will be guaranteed not to produce underflow bits. On the other hand, if they are larger than T_{BO}, and the difference is smaller than T_{GO}, the encoder will be guaranteed not to produce overflow bits. In the case when $T_M = 0$ (T_{BE}=T_{BL}), if produced encoding bits are smaller than T_{BO}, the underflow cannot be avoided. This condition wastes the available channel bandwidth and does not produce optimum reconstructed image quality. In other circumstances, when encoding bits are larger than the target, overflow bits at an encoder buffer cannot be allocated for transfer at the next frame bitstream transfer. This will cause the encoder to produce a number of skipped frames, and to not guarantee a defined D_{max}. To control the margin between T_{BE} and T_{BL}, the defined D_{max} should satisfy:

$$D_{max} \geq (2 \times C_c - 1) \times \frac{G}{F} - 1. \tag{7}$$

The ratio between underflow and overflow conditions that can be compensated within T_M is controlled by the weighting factor k in Eq. (6). In the case of $k < 0.5$, the rate control is more effective to compensate underflow. Conversely, $k > 0.5$ is more effective to compensate overflow and to reduce the number of skipped frames. To cover both conditions, $k = 0.5$ was adopted in this work.

From T_{BO} estimation, we can directly compute the optimum value of the encoding bits B_{BO} of each encoding frame in fixed channel bit rate as:

$$B_{BO} = T_{BO} \times \frac{R \times G}{F} \tag{8}$$

where B_{BO} is in *bit/frame* and R is in *bit/sec.*

3.3 Encoder Buffer Evaluation

In actual conditions, there is usually difference between the target and the achieved encoding bits. This condition produces remaining W bits at an encoder buffer. W is evaluated to avoid accumulated un-transmitted bits that can increase D and the number of skipped frames. At the beginning of frame encoding, W is initialized to be zero; otherwise, W is equal to its previous number of remaining bits W' and the remaining bits as a result of false target frame encoding bits achievement, as:

$$W = \max\left(W' + B'_U - \frac{R}{G} \times C'_c, 0\right). \quad (9)$$

Bits allocation should produce a very small value of W and a B_U value that is close to the channel bit rate $((R \times C'_c)/G)$.

3.4 Target Frame Encoding Bits Allocation

B_{TE} is computed as the target value for a current frame encoding bits. This value will be used in macroblock level rate control to compute Q. In the case of $W = 0$, B_{BO} can be directly used for B_{TE}. However, due to $W > 0$, the remaining bits should be transferred at the current frame, and B_{TE} can be computed as follows:

$$B_{TE} = \max(B_{BO} - W, 0). \quad (10)$$

3.5 Conditions for Frame Skip

In order to avoid unexpected delay longer than D_{max}, due to W being larger than the number of bits that can be transferred of T_M time, frame skipping is applied to the encoder. In the skip condition, the next encoding frame number is $i + 1$ (one frame skip) and its frame type is set at P ($C_c = 1$), as described in following pseudo code:

$$\begin{aligned} &\text{if } (W \geq (T_M \times R)) \\ &\quad i = i + 1,\ C_c = 1, \\ &W = \max\left(W - \frac{R}{G}, 0\right). \end{aligned} \quad (11)$$

This is different from conventional methods [7]–[9] that skip G/F (frame interval) frames. By skipping only one frame, we can minimize the number of skipped frames and reduce underflow bits, resulting from $W - (C_c \times (R/G)) < 0$.

3.6 Global Quantization Parameter

Under the assumption of a small statistical change between successive frames during encoding of a high frame rate video sequence, the global quantization parameter Q_G of the current frame can be estimated by previous average quantization parameter $\bar{Q}'$. Since there is no valid Q for the uncoded macroblock, only the coded macroblock Q (Q'_C) is considered to estimate Q_G, as:

$$\bar{Q}' = \frac{\sum_{j=0}^{N-1} Q'_C(j)}{N'_C}. \quad (12)$$

where N'_C represents the number of previous frame coded macroblocks.

To compensate Q_G, the estimation error of the previous frame, $\bar{Q}'$ from Eq. (12) is corrected by subtracting from it the ratio of error between B'_{TE} and B'_U as:

$$Q_G = \bar{Q}'\left(1 - \frac{B'_{TE} - B'_U - B'_{ST}}{B'_{TE} \times 2}\right), \quad (13)$$

where B'_{ST} is the number of previous frame stuff bits.

3.7 Bit Per Coefficient Parameter

Bit per coefficient parameter K_{BC} represents the ratio of the number of bits for encoding DCT coefficients B_{DCT} to the number of non-zero coefficients N_Z. This parameter always changes according to the distortion in every frame. This value can be computed directly by using the number of bits for encoding DCT coefficients B_{DCT} within one frame, divided by the number of non-zero coefficients N_Z. Because there is a correlation among frames, K_{BC} is computed in the form of an updating equation as

$$K_{BC} = K'_{BC} \times v + \frac{B_{DCT} \times Z \times (1 - v)}{N_Z}, \quad (14)$$

where v and Z are weighting factors for K_{BC} updating ($0 < v < 1, Z = 1..5$), and defined from the experimental results.

In our proposed rate control, K_{BC} parameters are used in macroblock level computation to estimate the macroblock encoding bits from its number of non-zero coefficients.

4. Macroblock Level Rate Control

In macroblock level rate control, the appropriate quantization parameter value of each macroblock Q is determined to meet allocated encoding bits B_{TE}. For each frame, Q_G is assigned to the first macroblock Q. The difference in Q between two successive macroblocks is denoted by Q_D, which satisfies $-2 \leq Q_D \leq 2$ (as described in H.263 standard).

4.1 Macroblock Bit Allocation

To allocate macroblock encoding bits from B_{TE}, we assume that all B_{TE} bits can be equivalently distributed to all macroblocks. Under this assumption, the estimated remaining bits at jth macroblock $\hat{B}_R(j)$ correspond to their position as:

$$\hat{B}_R(j) = \frac{(N_M - j)}{N_M} \times B_{TE}, \quad (15)$$

where j is the macroblock sequence number counted in raster scan order and N_M is the total number of macroblocks within one frame.

In fact, the actual encoding bits of each macroblock vary according to frame distortion and target frame encoding bits achievement. The actual number of current remaining bits $B_R(j)$ can be computed from previous actual remaining bits $B_R(j-1)$ and current macroblock encoding bits $B_{MU}(j)$ as:

$$B_R(j) = B_R(j-1) - (B_{MU}(j) + B_{ST}(j)). \quad (16)$$

where $B_{ST}(j)$ is stuff bits [7]. For the first macroblock, $B_R(0)$ is initialized to be equivalent to B_{TE}. By comparing $\hat{B}_R$ and B_R, we can increase Q if $\hat{B}_R > B_R$, or decrease it if $\hat{B}_R < B_R$.

To improve the precision of a macroblock bit allocation, the estimated value of the required bit for jth macroblock encoding $\hat{B}_{MU}(j)$ is computed in advance. $\hat{B}_{MU}(j)$ is correlated with the distortion measure of the current macroblock. By assuming that the statistical difference between successive macroblocks is small and macroblock encoding bits are correlated with the number of non-zero coefficient N_Z, we can compute $\hat{B}_{MU}$ from previous macroblock non-zero coefficient value $N_Z(i-1)$ as:

$$\hat{B}_{MU}(j) = \frac{N_Z(j-1) \times K_{BC}}{h} + 10. \quad (17)$$

where h is the weighting factor of K_{BC} and the constant 10 is the compensating bits of the macroblock encoding header.

To avoid over allocation of the current macroblock encoding bits, the estimated bits for encoding the remaining macroblock $\hat{B}_{MR}$ here to be computed. With the assumption that the same encoding bits are required, $\hat{B}_{MR}$ at the jth macroblock is

$$\hat{B}_{MR}(j) = \hat{B}_{MU}(j) \times (N_M - j). \quad (18)$$

4.2 Macroblock Quantization Parameter

The Q_D value of the current jth macroblock is computed by comparing parameters of encoding bits conditions (B_R, $\hat{B}_R$, B_{MR}) and the difference between previous macroblock quantization parameter $Q(j-1)$ and Q_G. The difference is computed for the upper difference Q_U and the lower difference Q_L, as:

$$Q_U = \max((Q(j-1) - Q_G), 0) \quad (19)$$

$$Q_L = \max((Q_G - Q(j-1)), 0). \quad (20)$$

Using the above parameters, Q_D can be computed as:

$$Q_D = \begin{cases} 2 & \text{if } (B_R < \hat{B}_{MR}) \cup \\ & (B_R \times (2 + Q_U) < \hat{B}_R) \cap \\ & (Q_L \neq 0) \\ -2 & \text{if } (B_R > \hat{B}_{MR}) \cap \\ & (B_R > \hat{B}_R \times (2 + Q_L)) \cap \\ & (Q(j-1) > 8) \\ -1 & \text{if } (B_R > (\hat{B}_{MR}) \cap \\ & (B_R > \hat{B}_R \times (2 + Q_L)) \cap \\ & (Q(j-1) \leq 8) \\ 0 & otherwise \end{cases} \quad (21)$$

The first case shows that if the actual remaining bits B_R are lower than the estimated bits for encoding the remaining macroblock $\hat{B}_{MR}$ or the estimated remaining bit $\hat{B}_R$ with some offset of $(2 + Q_U)$, the current macroblock Q should be increased by 2. Conversely, if B_R is larger than the estimated value $\hat{B}_R$, the current quantization parameter value can be adjusted according to the previous macroblock $Q(j-1)$. If $Q(j-1)$ is larger than 8, Q_D is decreased by 2, otherwise, Q_D is decreased by 1 if $Q(j-1)$ less than or equal to 8. The value 8 is used as a margin because there is a possibility of large encoding bits increment at very low quantization parameters.

5. Adaptive Threshold of Zero Motion Vector Estimation

In the H.263 framework, the number of bits for motion vector encoding is quite significant in comparison with the total number of encoding bits [1]. In the encoding of high-motion image sequence, those bits exceed the channel bit rate in which many skipped frames are unavoidable.

Moreover, during encoding of low-motion image sequence, the homogenous and small intensity differences between encoding frames are very sensitive to noise. A small amount of noise can produce false motion vectors that increase motion vector encoding bits. This phenomenon requires the encoder to set a threshold value for determining zero motion vectors.

Regarding the cause of the above large motion vector encoding bits, we propose an adaptive threshold for zero motion vector estimation STV. STV is defined as a function of Q as:

$$STV(Q, STF) = \max(2 \times Q \times STF, 100) \quad (22)$$

where STF is the SAD threshold factor that represents the weighting factor of Q.

In Eq. (22), the adaptive threshold is computed by measuring macroblock image distortion from the Q value. When a high Q is applied, the distortion of macroblock can increase. This condition produces a large number of false motion vectors over a stationary or homogenous intensity region. To overcome this, Eq. (22) sets a high STV value for the macroblock motion estimation threshold. This method is different from that of common H.263 encoders, e.g., Telenor (TMN8 and TMN5) [7]–[9], which use a fixed threshold for zero motion vector estimation.

The STF value is determined from the trade-off among the number of encoding bits, the number of skipped frames and the reconstructed image quality (PSNR). As can be seen in Fig. 8, the encoding bits value slightly changes for various values of STF. The number of DCT coefficients increases if we increase the STF value. Figure 8 shows that the STF value of approximately four is optimal, since the smallest encoding bits are produced. Referring to PSNR and the number

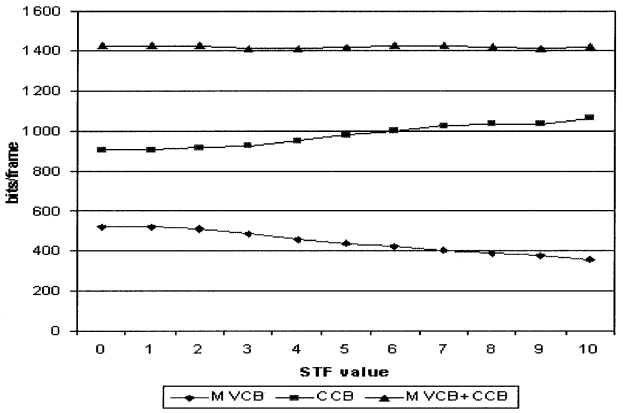


Fig. 8 Average frame encoding bits as a function of STF for image sequence *carphone*. MVCB:motion vector encoding bits, CCB: DCT coefficient encoding bits.

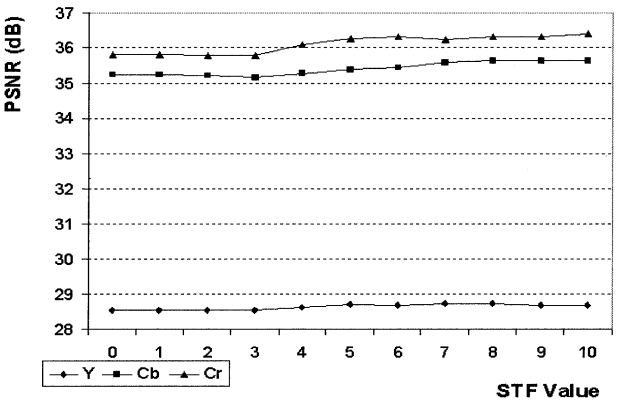


Fig. 9 PSNR as a function of STF for image sequence *carphone*.

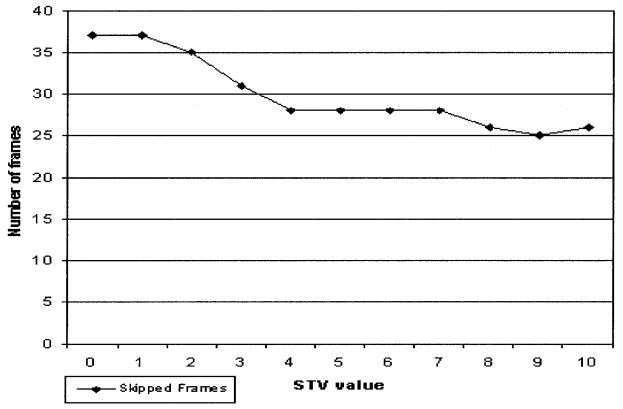


Fig. 10 Number of skipped frames as a function of STF for image sequence *carphone*.

of skipped frames, we also find a slight improvement of PSNR and many skipped frames reduction at an STF value of approximately 4, as can be confirmed in Figs. 9 and 10.

6. Experimental Results

The performance of the proposed low delay rate control (LDRC) was evaluated by comparing its encoding results of several popular QCIF size video sequences (Table 1) with the well-known Telenor H.263+ codec results (version 3.2). The Telenor codec was evaluated using TMN8 [9] rate control strategy. Regarding the target application of a high-performance encoder at very low channel bit rate and a high frame rate, all sequences were encoded at 27 kbps target bit rate and 30 frames/sec frame rate. To improve reconstructed image quality, we employed a deblocking filter and improved the PB frame type encoding mode. The first frame was intra-coded (I frame type) with $Q = 16$, while the remaining frames were all inter-coded (P frame type).

6.1 Bit Rate Achievement and Frame Skipping

Bit rate achievements in Table 1 describe the actual encoding bit rate difference from the defined target bit rate (27 kbps). It is observed that the LDRC achieved bit rates closer to the target in comparison with that of TMN8. The target bit rate can still be achieved, even for encoding of high-motion video sequences (*carphone, suzie, and foremen*). This feature confirmed that the LDRC is suitable for fixed low bit rate channel video coding application.

Frame skipping in Table 1 shows that LDRC produced a very small number of skipped frames in comparison with TMN8. For low-motion video sequences (*claire, miss_am and grandma*), the produced number of skipped frames was close to TMN8's. On the other hand, for high-motion video sequences, such as *foreman*, TMN8 produced a very large number of skipped frames.

As observed in Fig. 11, TMN8 failed to avoid many frame skips during encoding of *carphone* video sequence from frame numbers 170 to 360. TMN8 produced very large frame encoding bits (up to 7,000 bits); in contrast, LDRC produced only approximately 3,000 bits. The same phenomenon was observed in another high-motion video sequence *suzie*, as can be seen in Fig. 12. Both cases proved that LDRC successfully controlled bit allocation during encoding of a high-motion part of a video sequence.

For the low-motion video sequences, such as *salesman*, the produced frame encoding bits were almost the same for both LDRC and TMN8 (Fig. 13).

The results show that LDRC maintained a constant bit rate for both high- and low-motion video sequences.

6.2 Reconstructed Image Quality

We see from the image quality difference in Table 1

Table 1 Comparison of performance LDRC with that of TMN8. Bit rate achievements are computed by subtracting achieved encoding bit rate from 27 kbps. Frame skipping represents the number of generated skipped frames during encoding of video sequence. Image quality shows reconstructed image quality comparison, which is computed by subtracting PSNR of TMN8 and that of LDRC. Frm Num: number of frames in video sequence.

Sequence Name	Bit Rate Achievement		Skipped Frame		Image Quality			#Frm Num
	LDRC (kbps)	TMN8 (kbps)	LDRC (#Frm)	TMN8 (#Frm)	Y (dB)	Cb (dB)	Cr (dB)	
carphone	0.14	-3.82	28	125	-0.06	0.11	-0.03	382
claire	0.01	0.07	11	12	0.26	0.53	-0.11	494
grandma	-0.04	0.03	9	10	0.08	0.25	0.09	870
miss_am	0.15	0.24	6	8	0.32	-0.03	0.29	150
mthr_dotr	0.05	-0.03	11	18	0.07	0.14	0.07	961
salesman	0.05	-0.55	13	34	-0.3	0.09	-0.27	449
suzie	0.4	-2.68	11	41	-0.33	-0.02	-0.17	150
foreman	0.17	-5.4	69	203	-0.07	0.28	0.4	400

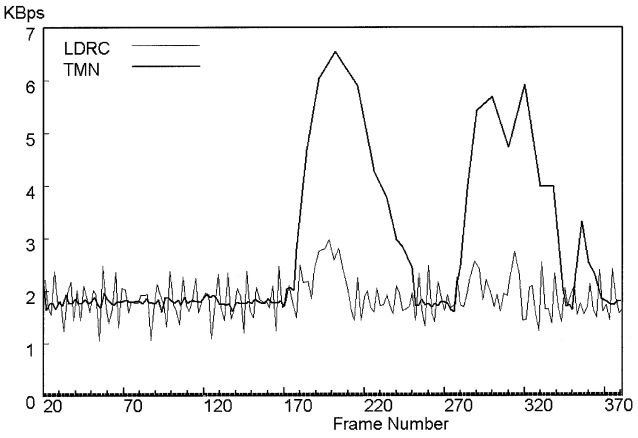


Fig. 11 Bit rate comparison between proposed algorithm (LDRC) and TMN8 for video sequence *carphone*.

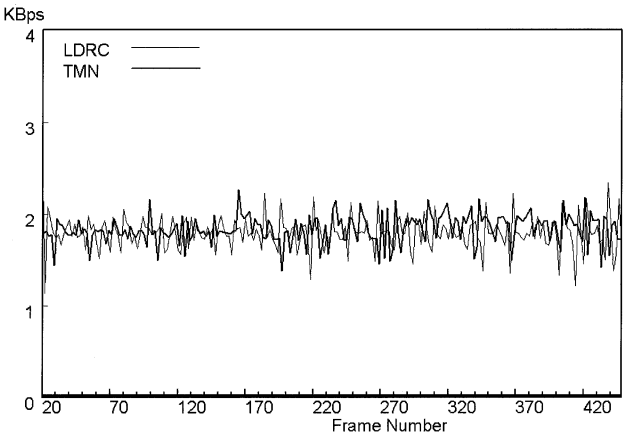


Fig. 13 Bit rate comparison between proposed algorithm (LDRC) and TMN8 for video sequence *salesman*.

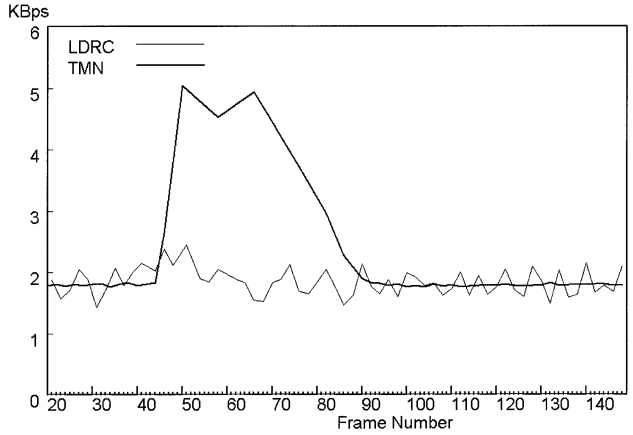


Fig. 12 Bit rate comparison between proposed algorithm (LDRC) and TMN8 for video sequence *suzie*.

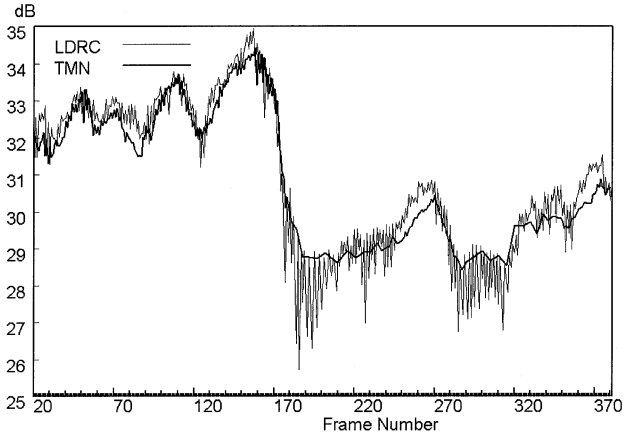


Fig. 14 PSNR comparison between proposed algorithm (LDRC) and TMN8 for image sequence *carphone*.

that there are some improvement and some decrement the average PSNR of LDRC in comparison with that of TMN8. The decrement of the average PSNR of LDRC is very small (not more than 0.58 dB). Moreover, if we observe the PSNR of each frame, the LDRC's frame PSNR was higher in most frames in Fig. 14 and as numbering 60 to 100 in Fig. 15. This condition occurred because LDRC reduced the number of skipped frames, thereby inter-frame difference and the number of frame encoding bits. This scheme leds to an improvement of image quality. The frame PSNR was improved especially after frame skipping operation had been per-

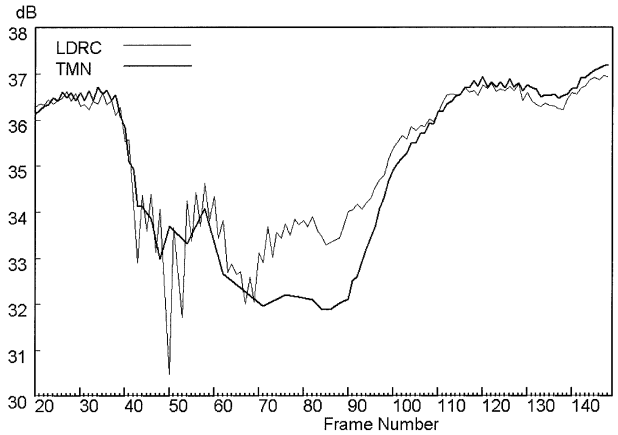


Fig. 15 PSNR comparison between proposed algorithm (LDRC) and TMN8 for image sequence *suzie*.

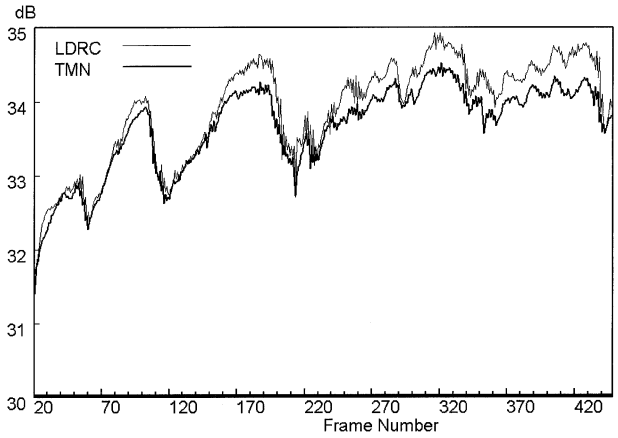


Fig. 16 PSNR comparison between proposed algorithm (LDRC) and TMN8 for video sequence *salesman*.

Fig. 17 Reconstructed image comparison between proposed algorithm **LDRC** (left) and TMN8 (right) for video sequence *carphone*.

formed in TMN8, as can be seen in the actual reconstructed image (Fig. 18). The same case was observed in the *carphone* video sequence, as shown in Fig. 17. A different phenomenon can be seen in the *salesman* video sequence, in which the skipped frames were distributed to all frames, and as a result, LDRC's PSNR became higher than TMN8's (Fig. 16). The result can be confirmed from produced high-quality reconstructed

Fig. 18 Reconstructed image comparison between proposed algorithm **LDRC** (left) and TMN8 (right) for video sequence *suzie*.

Fig. 19 Reconstructed image comparison between proposed algorithm **LDRC** (left) and TMN8 (right) for image sequence *salesman*.

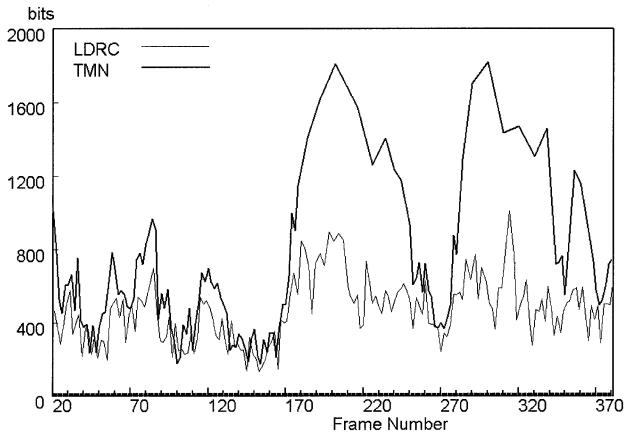


Fig. 20 MV bits comparison between proposed algorithm (LDRC) and TMN8 for video sequence *carphone*.

image as shown in Fig. 19.

6.3 Adaptive Threshold of Zero Motion Vector Estimation

At the very high-motion part of the video sequence, the number of bits for encoding motion vector is very large. It reached approximately 2,200 bits for *carphone* and *suzie* video sequences using the TMN8 (Figs. 20 and 21). As seen in those figures, the proposed *adaptive threshold of zero motion vector estimation* method with $STV = 4$ can effectively reduce the number of motion vector encoding bits fluctuation. As a result,

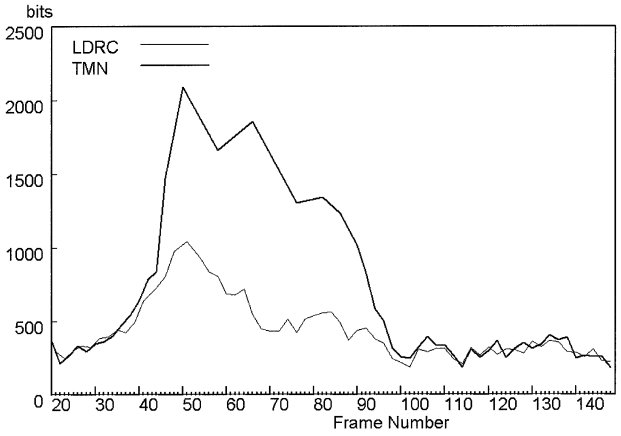


Fig. 21 MV bits comparison between proposed algorithm (LDRC) and TMN8 for video sequence *suzie*.

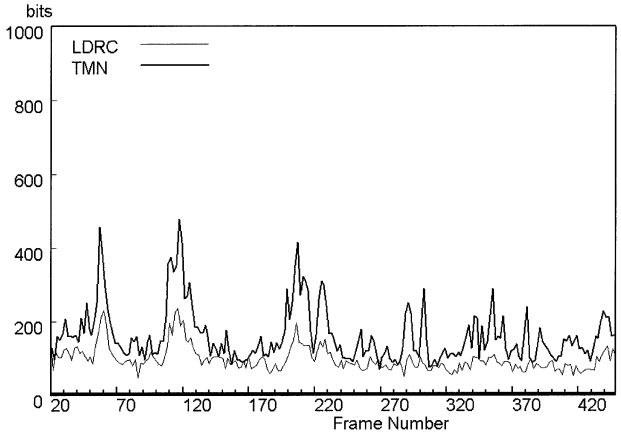


Fig. 22 MV bits comparison between proposed algorithm (LDRC) and TMN8 for video sequence *salesman*.

many skipped frames can be reduced for both video sequences.

When encoding low-motion video sequences such as *salesman*, the produced number of motion vector encoding bits was very small and the value was almost the same for both LDRC and TMN8 (Fig. 22).

This condition shows that the proposed method can effectively reduce the motion vector encoding bits during encoding of high-motion video sequences, and maintain the encoding bits in low-motion video sequences.

6.4 Encoder-Decoder Delay

Figure 23 shows the encoder-decoder delay between camera's input and display's output during the encoding of the high-motion part of the video sequence *carphone* (for input frame numbers 189 to 227). As shown in that figure, LDRC produced only five frames delay at most, but TMN8 produced 13 frames delay. As can be seen in Fig. 23(b), this was because the required bits for frame encoding reached 6,552 bits. Besides,

Table 2 Computational complexity comparison between LDRC and TMN8. M is the number of macroblocks within a frame, and P is the number of pixels within a macroblock.

Operation	LDRC	TMN8
Addition	$M+1$	$M \times (5 + P \times 4.5)$
Subtraction	$M \times 7 + 9$	$M \times (5 + P \times 4.5) + 1$
Multiplication	$M \times 3 + 3$	$M \times (13 + P \times 3) + 1$
Division	$M + 4$	$M \times 13$
Square root	0	$M \times 2$

Table 3 Computational complexity comparison obtained for QCIF size input image.

Operation	LDRC	TMN8
Addition	100	114,543
Subtraction	702	114,544
Multiplication	300	77,320
Division	103	1,287
Square root	0	198

the total underflow bits are also very large. A total of 50,980 (14.7%) underflow bits were generated using TMN8, while only 484 bits (0.14%) were generated using LDRC. The large number of total underflow bits shows that TMN8 cannot efficiently use the available channel bit rate.

6.5 Complexity Analysis

Computational complexity is measured by counting the number of operations that significantly affect the hardware implementation complexity, as described in Table 2. The results in Table 2 were obtained by assuming the largest number of possible operations in every frame and ignoring simple operations such as shifting, comparison and operation with a constant value.

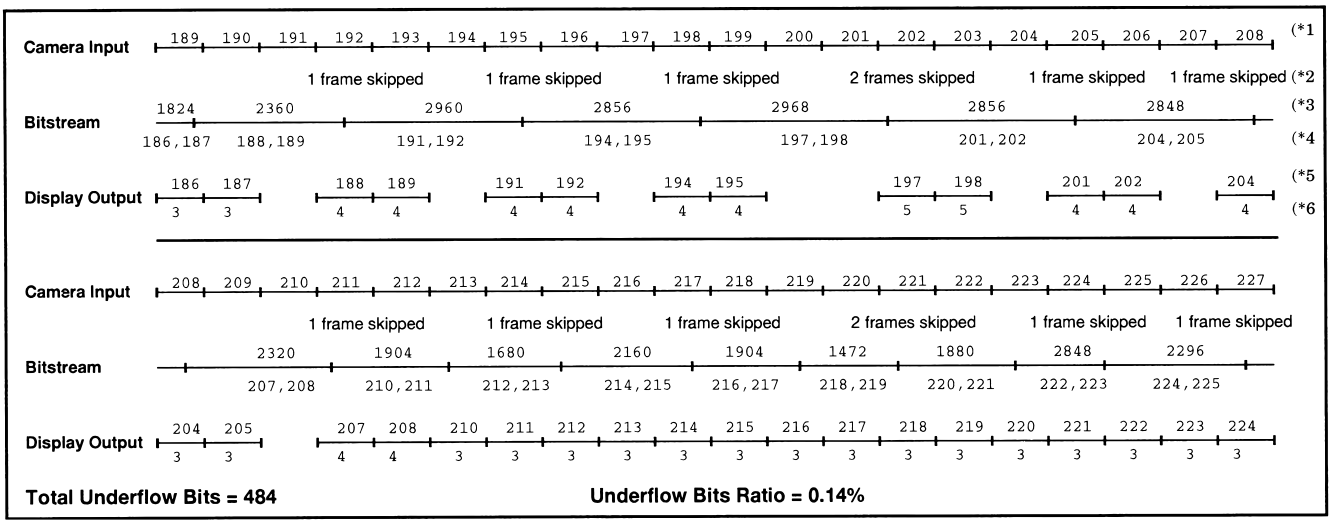
Table 2 shows a marked reduction of computational complexity, compared to that of TMN8. In the case of QCIF size input image, the quantitative comparisons are demonstrated in Table 3.

7. Conclusion

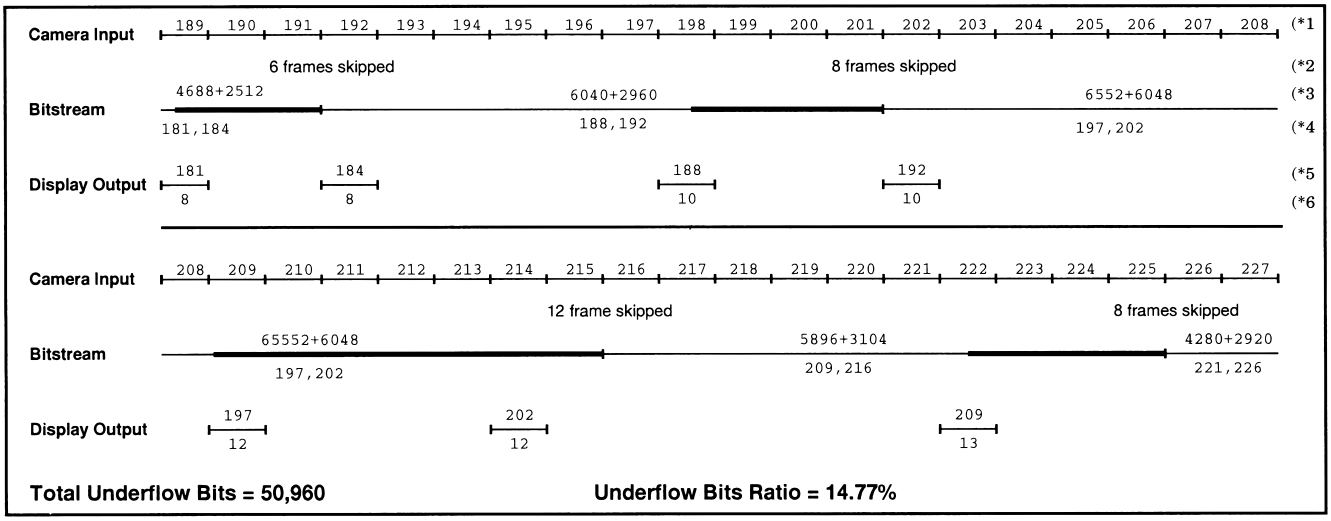
In this paper, we proposed a new rate control method that can effectively minimize encoder-decoder delay, which is very important for the elimination of the lip synchronization problem. It increases frame rate, reduces the number of skipped frames and produces high-quality image.

The simulation results confirmed that the proposed method achieved a bit rate much closer to the target bit rate than TMN8. It has only a 0.17 kbps target difference compared to the 5.4 kbps of TMN8.

This method also successfully reduced the frame buffer memory requirement for temporary data storage because it utilizes previous image information to compute quantization parameters and small remaining data at the buffer. This method utilizes simple operation and uses less number of computations for distor-



(a)



(b)

Fig. 23 Encoder-decoder delay timing diagram for image sequence *carphone*. (a) Results obtained using LDRC, and (b) results obtained using TMN8. Total underflow bits are the number of accumulated underflow bits within one video sequence encoding. Underflow bits ratio are computed by dividing the estimated underflow bits by the total number of bitstream. ($*^1$ camera input frame number, ($*^2$ number of skipped frames, ($*^3$ number of bitstreams which consist of encoding bits and underflow bits(bits), ($*^4$ bitstream frame number, ($*^5$ display output frame number, ($*^6$ encoder-decoder delay (frames).

tion measure parameters and quantization parameters computation. Both of these features are very important for hardware realization of high-quality H.263+ Video Coding.

Acknowledgments

Authors would like to thank all the members of CAD21 Research Body of Tokyo Institute of Technology for their suggestions and cooperation.

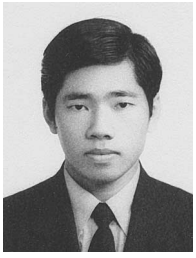
References

[1] G.J. Sullivan, and T. Wiegand, "Rate-distortion optimization for video compression," IEEE Signal Processing Mag., pp.74–90, Nov. 1998.
[2] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with application to multiresolution and MPEG video coders," IEEE Trans. Image Processing, vol.3, pp.533–545, Sept. 1994.
[3] G. Cote, B. Erol, M. Gallant, and F. Kossentini, "H.263+: Video coding at low bit rates," IEEE Trans. Circuits Syst. Video Technol., vol.8, no.7, pp.849–866, Feb. 1998.
[4] J. Ribas-Corbera and S. Lei, "Rate control in DCT video coding for low-delay communications," IEEE Trans. Circuits

Syst. Video Technol., vol.9, no.1, pp.172–185, Feb. 1999.
[5] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders," IEEE Trans. Image Processing, vol.3, no.5, pp.533–545, Sept. 1994.
[6] T. Wiegand, M. Lightstone, and D. Mukherjee, "Rate-distortion optimized for very low bit rate video coding and the emerging H.263 standard," IEEE Trans. Circuits Syst. Video Technol., vol.6, no.2, pp.182–190, April 1996.
[7] ITU-T, Study Group 16, "Rate control for low-delay communications," Document, Q15-A-20.
[8] ITU-T, Study Group 16, "Video codec test model, near-term, version 10 (TMN10) draft 1," Document, Q15-D-65d1.
[9] ITUT, Study Group 16, "Video codec test model, near-term, version 9 (TMN9)," Document, Q15-C-15.

Trio Adiono received a B.S. degree in Electrical Engineering, and an M.E. degree in Microelectronics from Bandung Institute of Technology, Indonesia, in 1994 and 1996, respectively. He is currently working toward his Ph.D. degree in Very Large Scale Integrated Circuit (VLSI) Design at Kunieda Laboratory, Tokyo Institute of Technology. His current research interests include very low bit-rate video coding, motion estimation techniques, H.263 video standard, Mpeg-4 Video Codec, VLSI Architecture for Video Applications, and Machine Vision System.

Tsuyoshi Isshiki received his Bachelor's and Master's degree in Electrical and Electronics Engineering from Tokyo Institute of Technology in 1990 and 1992, respectively. He received his Ph.D. degree in Computer Engineering from the University of California at Santa Cruz in 1996. He is currently an associate professor at the Department of Communications and Integrated Systems, Tokyo Institute of Technology. His research interests include high-level design methodology for configurable systems, bit-serial synthesis, FPGA architecture, image processing, computer graphics, and speech synthesis.



Chawalit Honsawek was born in Bangkok, Thailand in 1975. He received his B.E. degree from Chulalongkorn University, Bangkok, Thailand and his M.E. degree from Tokyo Institute of Technology, Japan, in 1996 and 1999, respectively. He is now working toward a Dr. Eng. degree at Tokyo Institute of Technology. His research interests include LSI design for digital signal processing.

Kazuhito Ito received his B.S., M.S., and Ph.D. degrees in Electrical Engineering from Tokyo Institute of Technology, Tokyo (Japan), in 1987, 1989, and 1992, respectively. He was with the Tokyo Institute of Technology from 1992 to 1995 and was with the University of Minnesota from April 1993 to July 1994. Currently, he is an associate professor of the Department of Electrical and Electronic Systems, Saitama University, Saitama (Japan). His research interests include high-level synthesis in digital signal processing, VLSI design automation, and system LSI design methodology.

Dongju Li was born in Shen-yang, Liao-ning Province, China. She received her B.S. degree from Liao-ning University and M.E. degree from Harbin Institute of Technology, China, in 1984 and 1987, respectively. She had been engaged in the VLSI design as an engineer in VLSI Design Laboratory of Northeast Microelectronics Institute, China, from 1987 to 1994. She received her Ph.D. degree in Electrical and Electronics Engineering from Tokyo Institute of Technology in 1998. She is presently a research associate in the Department of Communications and Integrated Systems, Tokyo Institute of Technology. Her current research interests include VLSI array processor architecture and its design methodology, system on chip, DSP, and VLSI design.

Hiroaki Kunieda was born in Yokohama in 1951. He received his B.E., M.E. and Dr.Eng. degrees from Tokyo Institute of Technology in 1973, 1975 and 1978, respectively. He was research associate in 1978 and associate professor in 1985, at Tokyo Institute of Technology. He is currently a professor at the Department of Communications and Integrated Systems, Tokyo Institute of Technology. He has been engaged in research on distributed circuits, switched capacitor circuits, IC circuit simulation, VLSI CAD, VLSI signal processing and VLSI design. His current research focuses on VLSI multimedia processing including video codec, design for system on chip, VLSI signal processing, VLSI architecture including reconfigurable architecture, and VLSI CAD. Dr. Kunieda is a member of IEEE CAS and SP Society and IPSJ.