Source Model for Transform Video Coder and Its Application—Part II: Variable Frame Rate Coding

Jiann-Jone Chen and Hsueh-Ming Hang, *Senior Member, IEEE*

***Abstract*— In the first part of this paper, we derive a source model describing the relationship between bits, distortion, and quantization step size for transform coders. Based on this source model, a variable frame rate coding algorithm is developed. The basic idea is to select a proper picture frame rate to ensure a minimum picture quality for every frame. Because our source model can predict approximately the number of coded bits when a certain quantization step size is used, we could predict the quality and bits of coded images without going through the entire real-coding process. Therefore, we could skip the right number of picture frames to accomplish the goal of constant image quality. Our proposed variable frame rate coding schemes are simple but quite effective as demonstrated by simulation results. The results of using another variable frame rate scheme, Test Model for H.263 (TMN-5), and the results of using a fixed frame rate coding scheme, Reference Model 8 for H.261 (RM8), are also provided for comparison.**

***Index Terms*— Image coding, rate distortion theory, source coding.**

I. Introduction

TRANSFORM coding is a very popular technique in image compression. It is one of the key components in the international video communication standards [1]–[3]. Since the channel bit rate is limited, bit-rate control or output buffer control becomes one of the essential problems in designing a video compression system because its output bit rate varies drastically. A simple rate control scheme achieves a preselected output bit rate by trial-and-error; namely, try a quantization step size, compare the produced bits to the target bit, and then adjust the step size and try again [4]. It may take several iterations before reaching an adequate quantization step size since no side information is collected to help predict the right value. Although buffer underflow and overflow can be eliminated under this multipass scheme, it is impractical for real video coding because the many iterations of quantization and variable-wordlength coding operation require a tremendous amount of hardware overhead.

A simple one-pass causal scheme was proposed by the *Reference Model 8* (RM8)—an encoder model used in the process of defining H.261 standard [5]. In this scheme, the quantization step size is assigned (linearly) proportional to the current output buffer level. When the output buffer is nearly full, a large step size is used to reduce the output bit rate. On the other hand, a small step size is used to avoid buffer underflow if the buffer is nearly empty. The effectiveness of this scheme depends on how frequently the step size is adjusted and how random the input data are. The major drawback of this scheme is the unevenness of image quality across a picture or a sequence. The buffer/quantizer controller may use a small step size at the beginning of an image if the initial buffer level is low. Then the step size is increased rapidly owning to the large number of bits generated by the small step sizes. As it proceeds, the buffer level would drop again, for fewer bits are generated by the large step sizes. The buffer/quantizer controller then again uses small step sizes. The value of quantization step may thus form an up-and-down cycle. In a hardware system, because of the inherent delay of one-pass system between the buffer status feedback and the quantization operation, the cyclic phenomenon becomes even more obvious. It may be further reinforced by the image content when one portion of the picture is smooth and the other portion has deep texture.

Several algorithms have been suggested to improve the simple one-pass RM8 buffer control scheme. One key element in a more advanced buffer control scheme is the bits model that predicts the number of coded bits when a certain quantization step size is in use. In the trial-and-error and RM8 heuristic buffer control schemes, bits model is not explicitly used, and hence, the control strategy is inefficient. The bits model could be constructed from the statistics of data ([6] and [7]). For example, Puri and Aravind [6] select the quantization step size of an image block according to its activity measure (variance) and a pretrained table. The entries in this pretrained table are computed from the statistics of training images coded using the same coders. Although quite effective, this scheme demands a large amount of training task and the coders and the processed images are limited to those similar to the training set. Another precalculated bits model example is Sun *et al*. [7]. A different activity measure, sum of the absolute values of discrete cosine transform (DCT) coefficients with DC component, is demonstrated to produce a better statistical relation.

In contrast, the bits model can also be obtained through the theoretical analysis of image and coder structures. Such a model is developed in the Part I of this paper. We like to demonstrate the use of this model by applying it to the buffer control problem. In addition to the usual quantization step size adjustment, the frame rate in our proposed scheme can also be adjusted adaptively. It is commonly believed

Manuscript received May 27, 1995; revised June 1, 1996. This paper was recommended by Associate Editor W. Li. This work was supported in part by the National Science Council of R.O.C., Taiwan, under Grant NSC83-0404-E009-021.

The authors are with the Department of Electric Engineering, National Chiao Tung University, Hsinchu, Taiwan, 300, R.O.C.

Publisher Item Identifier S 1051-8215(97)01141-5.

that when the bits number allocated to an image frame is not large enough to produce a reasonable quality image, it would be better off to skip several image frames to maintain a minimum quality of every coded frame. Takishima *et al.* [8] show that for a given sequence, the optimum frame rate can be uniquely specified through detailed off-line statistical analysis of the entire image sequence. However, an *on-line* adaptive algorithm is needed because a real-time video is changing its characteristics frequently and abruptly. A simple one-pass variable frame rate control had been proposed in the Test Model for H.263 (TMN-1.5) [9], [10]. It determines the number of skipped frames based on the buffer state while no picture coding complexity analysis is undertaken in coding process. As a result, the frame skipping is quite regular such that the performance is very similar to the fixed but fewer frame rate coding.

We like to demonstrate in this paper a methodology of using an analytic bits model (source model) to adjust the quantization step size and frame rate of a video coder to produce nearly constant quality pictures and, in the meanwhile, to limit the motion jerkiness to within a tolerable range. Although this approach can be extended to select the quantization step size for every image block to achieve the best tradeoff between bits and distortion for every single image block as shown by Lin *et al.* [11], Ortega *et al.* [12], and Ramchandran *et al.* [13], we select quantization step size only once for the entire image as an application example. Therefore, this approach is much simpler than the optimization formulation proposed in [11]–[13], and thus the real-time implementation is practical.

The organization of this paper is as follows. The source model or bits model proposed in Part I is briefly reviewed in Section II. Section III-A describes the basic encoder/decoder buffer relationship and the constraints of an H.261 video coder. The buffer/quantizer control algorithms of variable frame rate coding are developed in Sections III-B and III-C. In Section IV, the variable frame rate coding control of Test Model for H.263 is described for comparison. Simulation results are shown in Section V and Section VI summarizes our contribution in this paper.

II. Preamble—Source Model

In Part I, we derive a source model that describes the relationship between bits, distortion, and quantization step size for block transform video coders. It can be represented by the following equations:

$$\bar{b}(q_s) = -\frac{1}{\alpha(q_s)} \log_e q_s^2 + \frac{G(q_s)}{\alpha(q_s)} \tag{1}$$

where

$$G(q_s) = \frac{1}{L} \sum_{i \in A_2} \log_e \left[\frac{\beta_i(q_s) \cdot \epsilon^2 \cdot \sigma_i^2}{W_i^2}\right] + \alpha(q_s) \cdot \frac{b_{\text{EOB}}}{L} \tag{2}$$

and $A_2 = \{i | i \in \{0, 1, \cdots, L-1\}$ and $(\epsilon_i^2 \cdot \sigma_i^2) \geq W_i^2 \cdot q_s^2/\beta_i\}$. Or

$$q_s^2 = F(q_s) \cdot e^{-\alpha(q_s) \cdot \bar{b}} \tag{3}$$

with $F(q_s) = \exp\{G(q_s)\}$. In the above equations $\bar{b}(\cdot)$ represents the average coded bits per pel, q_s is the quantization step size, σ_i^2 the variance of the ith coefficient, L the number of pels (coefficients) in one block, ϵ a parameter decided by the probability distribution (assumed to be a constant if all the coefficients have similar distributions), W_i the weighting factor for the ith coefficient, usually preselected, and b_{EOB} the bits used to represent the end-of-block symbol (in JPEG, H.261, and MPEG). The model parameters $\alpha(q_s)$ and $\beta(q_s)$ were originally constants in the quantizer asymptotic model, but they are now variables adjusted to match the underlying video coder and the real picture characteristics. For low to medium bit rate coding, the values of $\beta(q_s)$ can be tabulated. As for $\alpha(q_s)$, we obtain a first-order linear approximation empirically

$$\alpha(q_s) = a_\alpha \cdot q_s + b_\alpha. \tag{4}$$

Initial values of a_α and b_α are calculated from picture data and are $a_\alpha = \sigma_{\text{ave}} \cdot 7 \times 10^{-4} + \lfloor \sigma_{\text{ave}}/10 \rfloor \cdot 0.0018 - \lfloor \sigma_{\text{ave}}/20 \rfloor \cdot 0.0096 - 0.02$ and $b_\alpha = \lfloor \sigma_{\text{ave}}/10 \rfloor \cdot 0.3 - \lfloor \sigma_{\text{ave}}/20 \rfloor \cdot 0.37 + 0.80$, where $\sigma_{\text{ave}} = \sum_{i=2}^{N} \sigma_i/(N-1)$.

Since the picture characteristics from the coding point of view are completely represented by the function $F(q_s)$, $F(q_s)$ is called "coding complexity function." For ease distinction, we denote $F(q_s)$ by $F^o(q_s)$ if it is computed by using (2) with $F(q_s) = \exp\{G(q_s)\}$. And we denote $F(q_s)$ by $F^m(q_s)$ if $\alpha(q_s)$ is set to a constant in (3) and $F(q_s)$ is allowed to vary to accommodate the variation of the originally nonconstant $\alpha(q_s)$. In Part I, we found that both $F^o(q_s)$ and $F^m(q_s)$ can be approximated by a linear function of q_s when q_s is relatively small (between 10 to 30, say). In other words

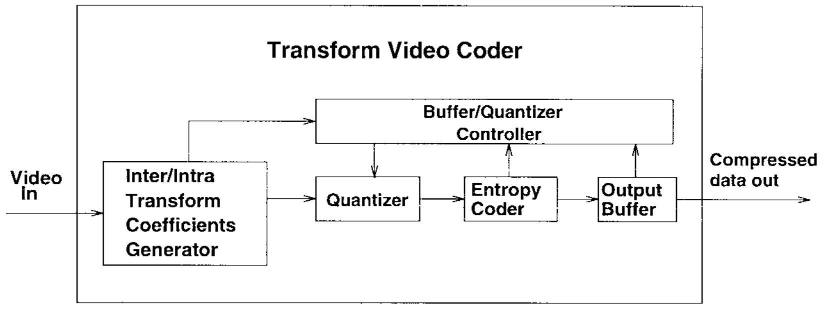
$$F(q_s) = a_F \cdot q_s + b_F \tag{5}$$

where a_F and b_F are two picture-dependent constants.

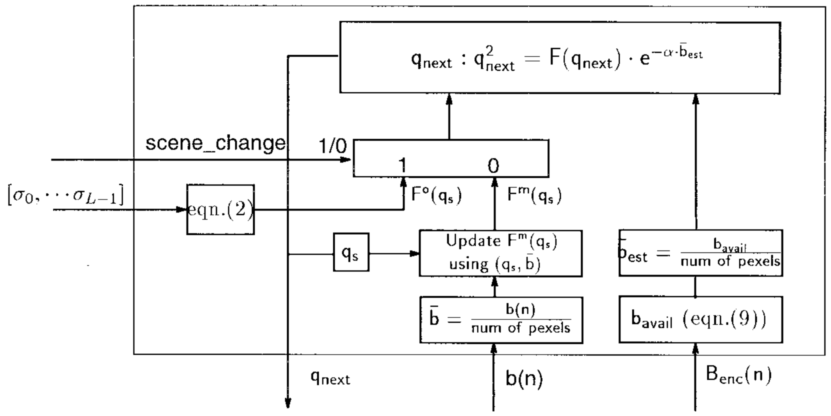
III. Variable Frame Rate Coding

As an application of our source model, a variable frame rate coder shown in Fig. 1 is described in this section. Based on the information received from the entropy coder and the output buffer, the buffer/quantizer controller in Fig. 1(a) determines the frame rate and the quantization step to be used for the next coded frame. The $P \times 64$ k standard specifies a receiver buffer model, the hypothetical reference decoder (HRD) with which all the bit streams generated by a standard-compatible encoder must comply. The well-known RM8 encoding scheme described in Section I assumes a fixed frame rate, which is improper for low bit-rate applications. At low bit rates, during periods of rapid motion, it is preferred to transmit fewer frames per second but with a better quality of each transmitted frame. There are additional drawbacks to the simple buffer/quantizer control algorithm in RM8, which selects the quantization step size linearly proportional to the buffer level. First, it ignores the overall quality of the images, and second, it costs additional overhead bits for changing the quantization step frequently to improve its stability.

In the following, we analyze the requirements imposed on a variable frame rate $P \times 64$ k codec and then construct an



(a)


$q_{next} : q_{next}^2 = F(q_{next}) \cdot e^{-\alpha \cdot \bar{b}_{est}}$
scene_change
1/0
1
0
$[\sigma_0, \cdots \sigma_{L-1}]$
eqn.(2)
$F^o(q_s)$
$F^m(q_s)$
q_s
Update $F^m(q_s)$ using $(q_s, \bar{b})$
$\bar{b}_{est} = \frac{b_{avail}}{\text{num of pexels}}$
$\bar{b} = \frac{b(n)}{\text{num of pexels}}$
b_{avail} (eqn.(9))
q_{next}
b(n)
$B_{enc}(n)$

(b)

Fig. 1. A variable frame rate coding system: (a) System diagram of variable frame rate coder and (b) operations of the VFR algorithm in the buffer/quantizer controller. Note that the number of pixels $= L \times$ (num of coded blocks).

adaptive buffer control algorithm based on our source model. The most straightforward solution to satisfy the HRD requirement is to simulate the (decoder) HRD buffer at the encoder and design a buffer control strategy to meet both the encoder buffer and the decoder buffer (HRD) requirements. But it is rather complicated to manage two buffers simultaneously. Fortunately, there is a definite relationship between the encoder buffer level and the decoder buffer level for constant rate channels. Hence, under proper choice of buffer size and target level, if the encoder buffer does not overflow or underflow, the HRD requirement is always satisfied. Also, our proposed algorithm predicts a target quantization step size which is expected to be used for the whole frame to reduce frequent adjustment.

A. Buffer Constraints

We use $b_{\text{enc}}(n)$ to denote the encoder buffer level after the nth video unit (frame) has been coded and $b_{\text{dec}}(n)$ to denote the decoder buffer level. Let d represent the delay (in video units) from the encoder buffer input to the decoder buffer output. A typical value of d, excluding the processing and transmission delays, is on the order of 10^{-1} s or slightly longer. We assume a constant channel rate and that, in the time span of one video unit, R_c bits are sent over the channel. For simplicity, we assume that the encoder and the decoder have an infinite processing power (no processing delay) and that the channel introduces no transmission delay. The processing and transmission delays can easily be incorporated into the following mathematical formulation for real systems. Let B_{enc} be the encoder buffer size and B_{dec}, the decoder buffer size. We have

$$b_{\text{enc}}(n) = b_{\text{enc}}(n-1) + b(n) - R_c \tag{6}$$

$$b_{\text{dec}}(n) = b_{\text{dec}}(n-1) + R_c - b(n-d) \tag{7}$$

where $b(n)$ are the coded bits of the nth video frame. Further derivation leads to the following propositions [11].

Proposition 1: $b_{\text{enc}}(n) + b_{\text{dec}}(n+d) = \textit{constant} \equiv K$. Proposition 1 says that the encoder and the decoder buffer levels are "complementary." Thus, the control of $b_{\text{dec}}(n)$ can be achieved by controlling only $b_{\text{enc}}(n)$.

Proposition 2: The encoder and the decoder buffer underflows and overflows can be eliminated if

$$\max(0, K - B_{\text{dec}}) \le b_{\text{enc}}(n) \le \min(B_{\text{enc}}, K).$$

The above two propositions are the constraints we adopted in designing buffer control algorithms. In the case where B_{dec} is specified ahead of time (the HRD requirement for H.261 and the VBV requirement for MPEG1), a simple choice is to set $K = B_{\text{enc}} = B_{\text{dec}}$ (if the resulting delay is acceptable). Our goal is to control the value of $b_{\text{enc}}(n)$ so as to make $b_{\text{dec}}(n)$ satisfy the HRD rules. The *target buffer fullness* (TBF), the desired buffer fullness at the end of encoding the current frame, is chosen to be inside the legal range, $w_1 \cdot \max(0, K - B_{\text{dec}}) + w_2 \cdot \min(B_{\text{enc}}, K)$, where w_1 and w_2 satisfy $w_1 + w_2 = 1$ and $w_1, w_2 \ge 0$. If frames n to

$n+fsk-1$ are skipped, (6) becomes

$$b_{\text{enc}}(n+fsk)=b_{\text{enc}}(n-1)+b(n+fsk)-(fsk+1)\cdot R_c. \quad (8)$$

Now, we wish $b_{\text{enc}}(n+fsk)$ would approach the desired buffer level, TBF. Therefore

$$\begin{aligned} b(n+fsk) =& w_1\cdot\max(0, K-B_{\text{dec}}) \\ &+ w_2\cdot\min(B_{\text{enc}}, K) \\ &+(fsk+1)\cdot R_c - b_{\text{enc}}(n-1) \end{aligned} \quad (9)$$

where fsk is the number of frames skipped and should be fewer than $b_{\text{enc}}(n-1)/R_c$ to avoid buffer underflow. We choose $w_1=1/3$ and $w_2=2/3$ in the simulations below.

B. Intraframe Coding

In a video sequence, except for scene changes, many frames would generally have similar image content. In other words, image contents usually does not change very much between nearby frames. Since the picture frame to be coded is not much different from the picture that has just been coded, the model parameter $\alpha(q_s)$ and the coding complexity function $F(q_s)$ of the current picture are similar to those of the previously coded pictures. In intraframe coding, where each picture is independently coded by the intra DCT coding, picture frames can be skipped without introducing propagation errors. Hence, we can easily employ this bits estimation model and variable frame rate (VFR) coding strategy to improve coding performance. In order to compare the VFR coding with RM8, we simply use the intra mode of H.261. Note that at the decoder, the last received picture is displayed on the monitor until the next coded frame arrives. If the number of skipped pictures is controlled within a reasonable range ($\leq fsk_{\max}$), the reconstructed picture jerkiness due to repeated frames should be acceptable. If additional delay and hardware are allowed at decoder, degradation due to frame skipping can be reduced by adding a proper temporal interpolator.

We now put together the aforementioned concepts to build a VFR coding algorithm. The general structure of our system is based on Hang [14] but with significant extensions and modifications. Our goal is to determine fsk, the number of skipped frames before the next coded frame and q_s, the quantization step size of the next coded frame. There are two ways to use (3). In the first approach, $\alpha(q_s)$ is a function of q_s (4) while $F^o(q_s)$ is assumed to be independent of q_s in a certain range of q_s. This scheme is called *VFR1*. We could also fix $\alpha(\cdot)$ (treated as a constant), but then $F^m(q_s)$ becomes a q_s-dependent variable (5). This is the second and simpler scheme—*VFR2*. The basic concept is the same in both approaches, which are illustrated by Fig. 1(b) (VFR2 is shown here). Initial $F(\cdot)$ (after scene change) is computed from the coefficient variances $[\sigma_0, \cdots, \sigma_{L-1}]$. After the first frame is coded, $F(\cdot)$ [and/or $\alpha(\cdot)$] is updated based on (3) using the previously coded q_s and $\bar{b}$. The bits budget $\bar{b}_{\text{est}}$ is derived from (9) using the current buffer level [$b_{\text{enc}}(n-1)$], the TBF, and the predicted frame skip (fsk). The q_s to be used, q_{next}, is estimated based on (3) using the bits budget $\bar{b}_{\text{est}}$ and the previous frame $F(q_s)$ [and/or $\alpha(q_s)$] under the assumption that $F(q_s)$ [and/or $\alpha(q_s)$] is not changed very much. If q_{next} is outside our desired range, fsk may be increased to meet our image spatial quality requirement. Details of these two schemes are give below.

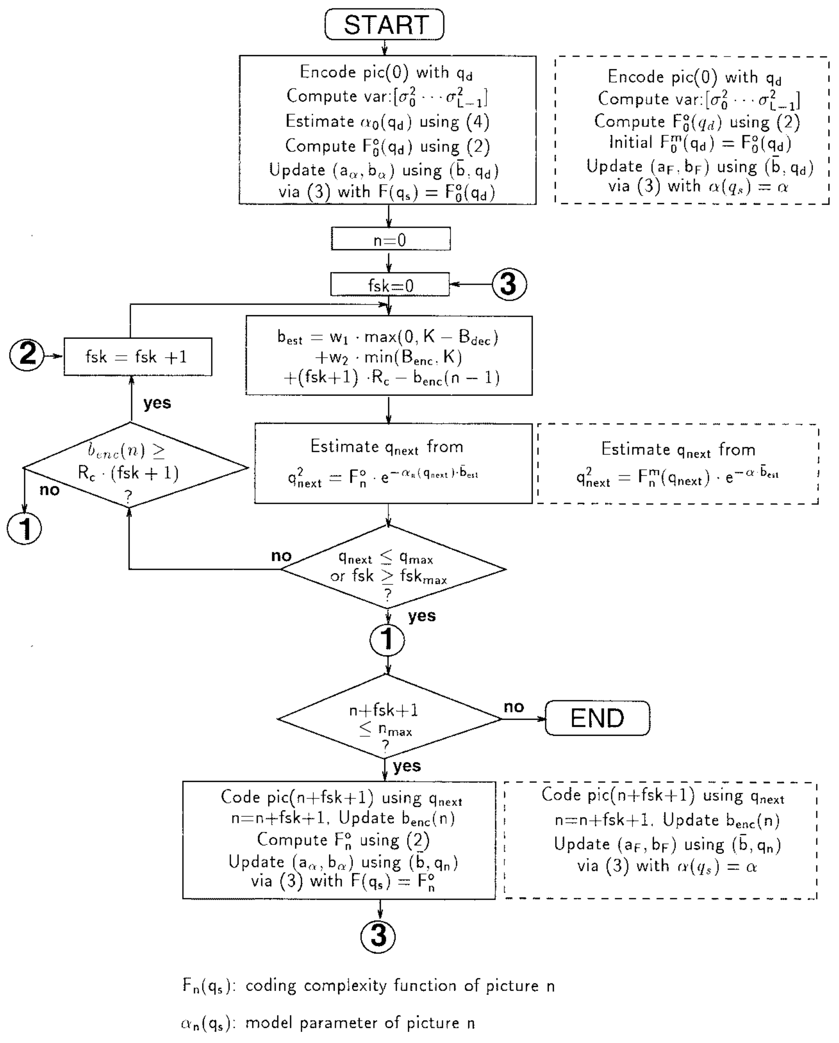
1) VFR Scheme 1—VFR1: The first algorithm (VFR1) is described by the flowchart in Fig. 2(a) with the solid-line boxes. It first encodes the first picture, $pic(0)$, with a default quantization step, q_d. The variances of the transformed coefficients, computed in the coding process, are used to guess the initial values of the a_α and b_α in (4). The $F^o(q_s)$ of $pic(0)$ is also computed from the coefficient variances using (2). Once $pic(0)$ is coded, the generated bits, $\bar{b}$, and the quantization scale used, q_d, together with the computed $F^o(\cdot)$ are used to calculate a new α value based on (3). Then, the a_α and b_α in (4) are updated based on the old α value and the new α value. The following simple, heuristic updating formulas are used:

$$a_\alpha^{\text{new}} = \delta\cdot a_\alpha^{\text{old}} + (1-\delta)\cdot\frac{\alpha^{\text{new}}-\alpha^{\text{old}}}{q_s^{\text{new}}-q_s^{\text{old}}} \quad (10)$$

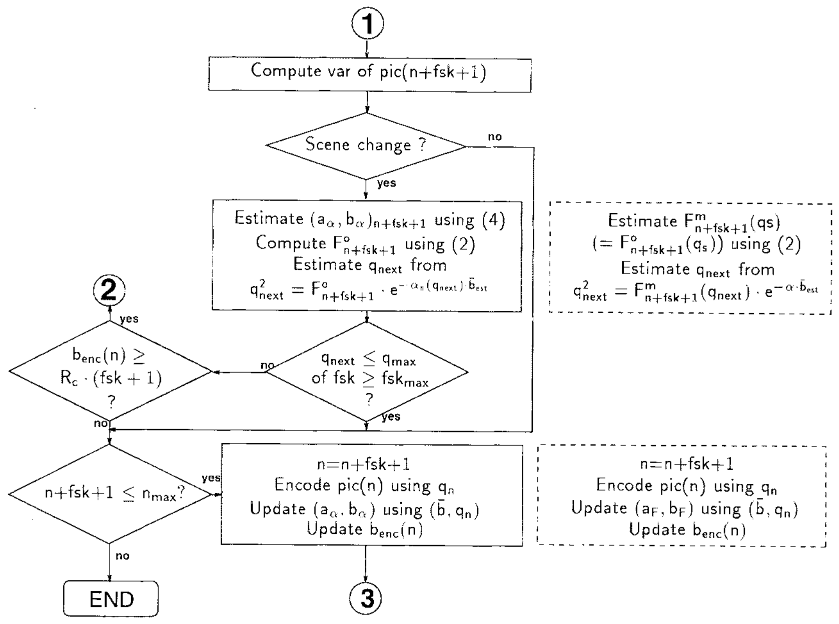
$$b_\alpha^{\text{new}} = \alpha^{\text{new}} - a_\alpha^{\text{new}}\cdot q_s^{\text{new}} \quad (11)$$

where the superscript new and old denote the previous and the current coded pictures, respectively, and δ is set to be one when $q_s^{\text{new}} = q_s^{\text{old}}$ and 0.3, otherwise. For the first frame, $pic(0)$, $q_s^{\text{new}} = q_s^{\text{old}} = q_d$. There are several other possible ways to generate the initial (a_α, b_α) value in (4). For example, $pic(0)$ is coded twice with two different values of q_s. Clearly, this would double the hardware burden in a real-time machine. Therefore, we developed an empirical formula to estimate (a_α, b_α) based on coefficient variances. And because of the simple form of our updating formula (11), b_α^{old} is not used at all. Also note that the only reason that we use the default q_d for the first picture is to match exactly what RM8 does on the first picture; otherwise, we could have even better results on $pic(0)$ if the scene change procedure described later is used instead.

Then, we start to encode the next frame under the assumption that F^o of the current frame is similar to the F^o obtained from the previously coded frame data. Note that although the value of F^o is updated from frame to frame, it is assumed to be independent of q_s for any given frame. We now enter the iteration loop of determining the number of skipped pictures. Starting with $fsk=0$, the bits budget reserved for the next picture is computed based on (9). And then the image quantization scale is estimated based on (3) using the previous frame F^o and the linear model of $\alpha(q_s)$ (4). If the estimated picture quality is not high enough; that is, the estimated quantization scale q_{next} is larger than the prechosen $q_{\max}$ and the maximum allowable number of skipped frames is not exceeded ($fsk \not\geq fsk_{\max}$), then we skip one additional frame to increase the bits budget for the next iteration. Otherwise, this q_{next} is satisfactory, and we jump out of the iteration loop. Before it proceeds to the next iteration, the buffer level is checked to prevent the buffer from underflowing. For the new fsk value, we again estimate the corresponding q_{next} value and check its associated image quality. The estimation and checking steps continue until enough bits are reserved to produce a good-quality picture. As said earlier, in determining the number of skipped frames, we use the (a_α, b_α) and F^o of the previously coded picture in the iteration loop. Once a new picture has been coded, the transform coefficient variances,


START
Encode pic(0) with q_d
Compute var:$[\sigma_0^2 \cdots \sigma_{L-1}^2]$
Estimate $\alpha_0(q_d)$ using (4)
Compute $F_0^o(q_d)$ using (2)
Update (a_α, b_α) using $(\bar{b}, q_d)$
via (3) with $F(q_s) = F_0^o(q_d)$
Encode pic(0) with q_d
Compute var:$[\sigma_0^2 \cdots \sigma_{L-1}^2]$
Compute $F_0^o(q_d)$ using (2)
Initial $F_0^m(q_d) = F_0^o(q_d)$
Update (a_F, b_F) using $(\bar{b}, q_d)$
via (3) with $\alpha(q_s) = \alpha$
n=0
fsk=0
3
2
fsk = fsk +1
$b_{est} = w_1 \cdot \max(0, K - B_{dec})$
$+w_2 \cdot \min(B_{enc}, K)$
$+(fsk+1) \cdot R_c - b_{enc}(n-1)$
yes
$b_{enc}(n) \geq$
$R_c \cdot (fsk+1)$
?
no
1
Estimate q_{next} from
$q_{next}^2 = F_n^o \cdot e^{-\alpha_n(q_{next}) \cdot \bar{b}_{est}}$
Estimate q_{next} from
$q_{next}^2 = F_n^m(q_{next}) \cdot e^{-\alpha \cdot \bar{b}_{est}}$
no
$q_{next} \leq q_{max}$
or fsk $\geq$ fsk$_{max}$
?
yes
1
n+fsk+1
$\leq n_{max}$
?
no
END
yes
Code pic(n+fsk+1) using q_{next}
n=n+fsk+1, Update $b_{enc}(n)$
Compute F_n^o using (2)
Update (a_α, b_α) using $(\bar{b}, q_n)$
via (3) with $F(q_s) = F_n^o$
Code pic(n+fsk+1) using q_{next}
n=n+fsk+1, Update $b_{enc}(n)$
Update (a_F, b_F) using $(\bar{b}, q_n)$
via (3) with $\alpha(q_s) = \alpha$
3
$F_n(q_s)$: coding complexity function of picture n
$\alpha_n(q_s)$: model parameter of picture n

(a)


1
Compute var of pic(n+fsk+1)
Scene change ?
no
yes
Estimate $(a_\alpha, b_\alpha)_{n+fsk+1}$ using (4)
Compute $F_{n+fsk+1}^o$ using (2)
Estimate q_{next} from
$q_{next}^2 = F_{n+fsk+1}^o \cdot e^{-\alpha_n(q_{next}) \cdot \bar{b}_{est}}$
Estimate $F_{n+fsk+1}^m(qs)$
$(= F_{n+fsk+1}^o(q_s))$ using (2)
Estimate q_{next} from
$q_{next}^2 = F_{n+fsk+1}^m(q_{next}) \cdot e^{-\alpha \cdot \bar{b}_{est}}$
2
yes
$b_{enc}(n) \geq$
$R_c \cdot (fsk+1)$
?
no
$q_{next} \leq q_{max}$
of fsk $\geq$ fsk$_{max}$
?
yes
no
n+fsk+1 $\leq n_{max}$?
yes
n=n+fsk+1
Encode pic(n) using q_n
Update (a_α, b_α) using $(\bar{b}, q_n)$
Update $b_{enc}(n)$
n=n+fsk+1
Encode pic(n) using q_n
Update (a_F, b_F) using $(\bar{b}, q_n)$
Update $b_{enc}(n)$
no
END
3

(b)

Fig. 2. Flowchart of variable frame rate coding: (a) VFR coding without considering delay coding and (b) additional VFR steps for detecting and handling scene changes.

$[\sigma_0^2, \sigma_1^2, \cdots, \sigma_{L-1}^2]$, and the coding result, $(\bar{b}, q_s)$, are used to update F^o and (a_α, b_α). That is, F^o is computed directly using (2) and a new α value is computed using (3), which in term is used to update the (a_α, b_α) pair using the heuristic formulas (10) and (11).

There are cases that would force the above procedure to encode the current picture without satisfying the minimum quality constraint. For example, if the encoder buffer underflow occurs or we hit the maximum skip frames $(fsk_{\max})$, then picture $(n + fsk + 1)$ is coded with a q_{next} that may be larger than $q_{\max}$. In general, this scheme tries its best to keep the coded image quality within a reasonable range while it must strictly prevent the encoder buffer from underflow and overflow. This scheme performs quite well for picture sequences without scene changes.

When scene change occurs, the F^o and $\alpha(q_s)$ of the current picture could be very different from those of the previously coded frames. Similar to the steps used for the first picture, they now have to be computed from the statistics of the current picture. Taking this into consideration, an additional scene change detection step described by the flowchart with solid-line boxes in Fig. 2(b) is developed. The entire processing procedure is almost the same as before except that a scene change detector is introduced. Now, the portion between node① and node③ in Fig. 2(a) is replaced by Fig. 2(b). Scene changes can easily be detected by comparing the variances of transform coefficients between the previous frame and the current frame. If the total absolute difference is greater than a properly selected threshold, scene change is assumed. In this case, the F^o and the (a_α, b_α) pair are all computed directly from the coefficient variances using (2) and (4), respectively.

2) Simplified Variable Frame Rate Scheme—VFR2: In the VFR1 scheme, $F^o(\cdot)$ is computed from the coefficient variances; hence, we need to accumulate the squared value of every coefficient of all frequencies. A certain amount of hardware is needed to perform this task for real-time coders. Also, the worst-case computational complexity of the iteration loop of determining fsk is $O(N)$, where N is the number of quantization scales. Further simplification can be achieved by using the modified coding complexity function $F^m(q_s)$. We call this simplified scheme "VFR2." In VFR1, the picture content variation is tracked by updating both F^o and α. In VFR2, only F^m is updated and as a matter of fact, just two coefficients a_F and b_F have to be calculated. Note also that in VFR1, the initial guess of (a_α, b_α) pair is made through *empirically determined* formulas [the equations about a_α and b_α next to (4)]. Now in VFR2, this initial guess is no longer needed and thus all the initial model parameters are estimated based on the *theoretically derived* formulas.

Assume pictures are coded with acceptable quality even at low channel rates; i.e., quantization scales are relatively small. As discussed in Part I, the value of $\alpha_s(q_s)$ would be close to one under the assumption of small quantization stepsize; hence, $F^m(q_s)$ is quite close to $F^o(q_s)$. Therefore, the initial value of $F^m(\cdot)$ can be approximated by using $F^o(\cdot)$ which is computed from the coefficient variances. After start-up, (a_F, b_F) in (5) can be updated based on the $F^m(q_s)$ computed using (3) and the coded bits $\bar{b}$ and quantization scale q_s. The updating procedure will be explained later.

The complete VFR2 algorithm is described by Fig. 2(a) with dashed-line boxes in place of the corresponding boxes in VFR1. Since α is treated as a constant, for a given bit budget $\bar{b}_{\text{est}}$, the estimation of q_{next} is equivalent to finding the roots of the following quadratic equation:

$$\begin{aligned} q_{\text{next}}^2 &= F(q_{\text{next}}) \cdot e^{-\alpha(q_{\text{next}})\cdot \bar{b}_{\text{est}}} \\ &\simeq (a_F \cdot q_{\text{next}} + b_F) \cdot e^{-\alpha \cdot \bar{b}_{\text{est}}}. \end{aligned} \tag{12}$$

Thus, the computational complexity is reduced to calculating the roots of the above equation only once. In VFR1, we update the exponent term in (3), $\alpha(q_s) = a_\alpha \cdot q_s + b_\alpha$. In VFR2, we update the multiplicative term, $F(q_s) = a_F \cdot q_s + b_F$. Both are updated through the use of (3) based on the coded data $(\bar{b}, q_s)$. But the former is more sensitive to noise because α sits on the exponent. Hence, the coding process using VFR2 is more stable as indicated by the simulations in Section V-A. The flowchart with dashed-line boxes in Fig. 2(b) is the VFR2 algorithm with scene change detector. Its operation is similar to the scene-change version of VFR1 except that F^m is computed from coefficient variances and α is held as a constant. Once a picture is coded, (a_F, b_F) rather than (a_α, b_α) is updated.

Since the values of $[F^m(q_s), q_s]$ (and $[\alpha(q_s), q_s)$ too] are often clustered in a narrow region, for stability and simplicity, we adjust (a_F, b_F) using the following heuristic formulas:

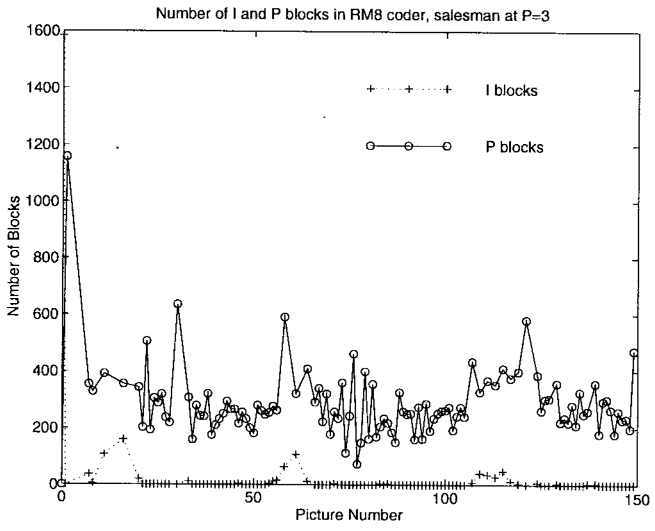
$$\begin{aligned} a_F^{\text{new}} &= \delta \cdot a_F^{\text{old}} + (1-\delta) \\ &\quad \cdot \frac{(q_s^{\text{new}} \cdot e^{\alpha \cdot \bar{b}^{\text{new}}} - q_s^{\text{old}} \cdot e^{\alpha \cdot \bar{b}^{\text{old}}})}{(q_s^{\text{new}} - q_s^{\text{old}})} \end{aligned} \tag{13}$$

$$b_F^{\text{new}} = q_s^{\text{new}} \cdot e^{\alpha \cdot \bar{b}^{\text{new}}} - a_F^{\text{new}} \cdot q_s^{\text{new}} \tag{14}$$

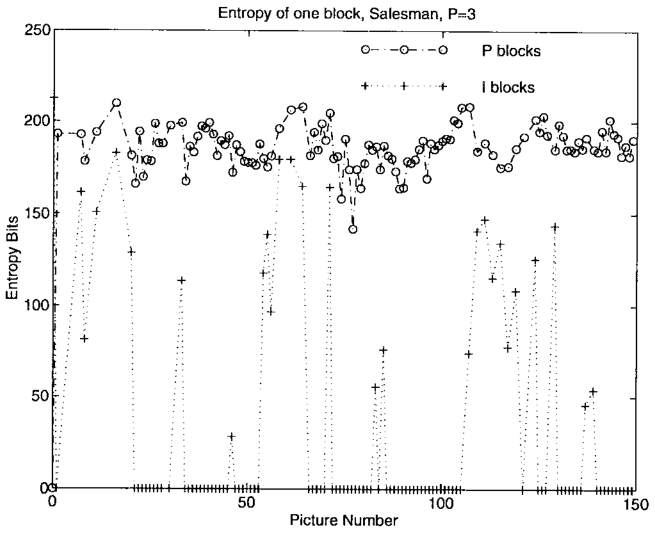
where the superscripts old and new denote the previous and the current coded pictures, respectively, and δ is set to be one when $q_s^{\text{new}} = q_s^{\text{old}}$ and 0.3, otherwise.

C. Intra/Inter Coding (H.261)

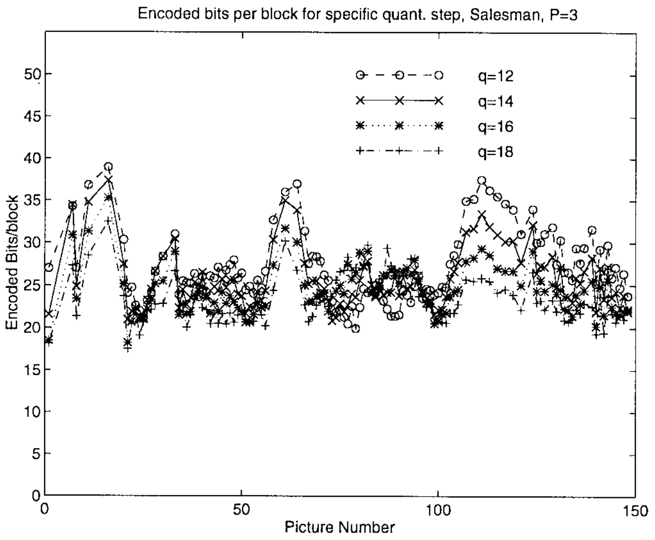
It is rather difficult to construct an accurate source model for a coder with mixed intra/inter coding modes, since the statistics of intra and inter coded pixels are very different. This becomes even more complicated when the current frame statistics depend on the previously coded frames. Nevertheless, it seems to be valid that in a video sequence, the numbers of intra and inter coding blocks and their characteristics are roughly unchanged for nearby frames. Fig. 3(a) shows the numbers of I and P blocks in each coded frame of the *Salesman* sequence coded by RM8 at a constant channel rate of 3 × 64 kb/s. It indicates that usually the I block number in an inter coded frame is quite small. Then, we compute the average entropy of the I and P blocks in each frame as shown in Fig. 3(b). The entropy of P blocks is nearly constant. Because the number of I blocks is very small in this case, its entropy estimate is not accurate. However, our concern is the bits versus q_s characteristics in the coded sequence. As shown by Fig. 3(c), the bits generated using the same q_s are similar for adjacent frames and the variations seem to be related to the number of coding blocks. Hence, the VFR algorithm described



(a)



(b)



(c)

Fig. 3. I and P block characteristics in interframe coding (RM8 on Salesman with bit rate $P = 3$). (a) Numbers of I and P blocks, (b) I and P block entropy (computed by formula), and (c) average coded bits per block.

earlier can also be applied to intra/inter coding schemes with some modifications.

1) Buffer Control Strategy: RM8 assumes half buffer fullness after the first intra picture is coded. However, this is not always doable for real images. The picture quality after scene change would degrade drastically if the buffer level is not properly controlled (the curve of RM8 in Fig. 6). This problem can be overcome through the use of bits model and VFR algorithm. However, the statistics collected from the intracoded frames cannot be applied to the intercoded frames without careful modifications. A simple solution to this problem is to encode the first inter (or more precisely, inter/intra) frame using a default quantization step size and the buffer/quantizer adjustment is active only after the first inter frame is completely coded. Hence, in addition to picture zero, which is intracoded with a default q_s, the first intercoded picture also uses a default q_s. Again, the only reason that we use default q_s for the intra picture and the first inter picture is to match what RM8 does on the first picture; otherwise, we could estimate q_s of the intra and the first inter pictures using coefficient variances as that in the scene change procedure.

Therefore, in our experiment (Section V), the first picture is intracoded with the default quantization scale and the second frame is skipped automatically for the purpose of reserving more bits for the next coded picture. The bits model for the intercoded picture is constructed using the data produced by the first (intracoded) and the third (intra/inter-coded) frames, and then the buffer control algorithm of VFR2 (in Section III-B-2) becomes active for the rest of the picture sequence. Although both intercoded and intracoded image blocks coexist in an intercoded picture, we assume they all follow the same formula (3) with different $F(\cdot)$ values. Suppose the ratio of these two types of blocks remains approximately the same for nearby frames, we only need to estimate a mixed $F(\cdot)$ value, which is a combination of originally two separate $F(\cdot)$ values. Therefore, the buffer control procedure is still similar to that described in Section III-B.

Due to the fact that the dynamic range of buffer fullness is smaller in interframe coding as compared to intraframe coding, a preferred buffer control algorithm should set two different TBF's for these two coding modes. Another reason of using a smaller $\text{TBF}_{\text{inter}}$ is to leave more space in the encoder buffer preparing for the large amount of intracoded bits during scene changes. Therefore, in our experiments, $\text{TBF}_{\text{intra}}$ and $\text{TBF}_{\text{inter}}$ are defined as two separate values to deal with intra and inter picture coding. The discontinuity between the two TBF values is smoothed out by inserting intermediate TBF's that decrease gradually from $\text{TBF}_{\text{intra}}$ to $\text{TBF}_{\text{inter}}$ after scene change; i.e.,

$$TBF_{\text{smooth}} = \frac{k}{m} \cdot (TBF_{\text{intra}} - TBF_{\text{inter}}) + TBF_{\text{inter}} \quad (15)$$

where $k = m$ for the first inter picture after scene change and then k is decreased by one for every frame until k reaches zero. The smoothing duration parameter, m, is chosen to be eight in our experiments. As we will see from Figs. 6(c) and 7(c), the end-of-picture buffer level decreases gradually after the first intra picture is coded and the quantization scale is controlled rather well during this transition period. The picture degradation after scene change is reduced by using this VFR algorithm and the new target buffer strategy.

D. Extra Cost of Using VFR

The tradeoff between quality improvement and system complexity of the proposed VFR algorithms is discussed below. These VFR schemes perform rather well in improving picture quality. However, the extra costs are: 1) a larger coder output buffer (compared to RM8) is needed, 2) additional memory is needed to store the transform coefficients for scene change handling and initial start-up, and 3) additional simple arithmetic operations are needed to execute the buffer/quantizer control algorithm (Fig. 1). In reality, the scene change handling procedure should be active all the time since the timing of scene change is unpredictable. Hence, 2) is performed for every frame. Compared with many suggested buffer control algorithms, the above extra software/hardware cost is rather mild and can easily be implemented.

IV. Variable Frame Rate Control in H.263 Test Model

The ITU-T Study Group XV has drafted H.263 for transmitting video below 64 kb/s while preserving reasonable picture quality. Because there are few well-known variable frame rate coding algorithms and the Test Model of H.263 (TMN) implements a variable frame control strategy, we thus introduce it here and will simulate it for comparison purposes later in the next section. Though H.261 and H.263 share the same basic codec structure, H.263 has a significantly enhanced performance due to improved motion estimation and the inclusion of other coding techniques, which are collected as the optional coding-modes ("options") in H.263.

The variable frame rate control strategy in TMN5 [10] is briefly described as follows. When the buffer fullness exceeds a specific threshold (for example, TBF $= 3 \times P \times$ 64k$/pic_rate$) in encoding frame n, the controller determines the number of frames to be skipped (i.e., fsk) so that the buffer fullness at $n + fsk$ would drop below TBF. Then it skips fsk frame and encodes frame $(n + fsk + 1)$.

In order to focus only on the rate control methods, we implement the aforementioned TMN5 variable frame rate control [10] on the H.261-type coder without the optional PB-mode in H.263. We call this coder "TMN" hereafter. It consists of the following steps.

1) At the beginning we need to specify a target frame rate f_{target}. Then, let picture number $n = 0$ and encode picture zero with the default q_s. The initial new target frame rate $\hat{f}_{\text{target}} = f_{\text{target}}$.
2) Set the target bit rate $b_{\text{target}} = P \times 64\,\text{kb/s}/\hat{f}_{\text{target}}$ and the target buffer fullness TBF $= 3 \times P \times 64\text{k}/pic\_rate$. The initial buffer fullness $b_{\text{enc}}(n)$ is set to be $TBF + b_{\text{target}}$ and let $b(n) = b_{\text{target}}$ after encoding frame zero.
3) Determine fsk such that $b_{\text{enc}}(n) - fsk \cdot R_c \leq TBF \leq b_{\text{enc}}(n) - (fsk - 1) \cdot R_c$, and set $n = n + fsk + 1$.
4) Set the global adjustment factor $G_{\text{adj}} = [b(n - fsk - 1) - b_{\text{target}}]/(2 \cdot b_{\text{target}})$.
Encode image blocks in frame n. The image block index is denoted by i_{MB}. The local discrepancy L_{adj} is defined to be $12 \cdot X/R_c$, where $X = [b(n, i_{MB}) - i_{MB}/N_{\text{total}MB} \cdot b_{\text{target}}]$. The quantization scale is adjusted once for every 11 MB's according to the following formula:
$$q_s(n, i_{MB} + 1) = q_s(n - fsk - 1)|_{\text{ave}} \cdot (1 + G_{\text{adj}} + L_{\text{adj}}) + 0.5 \quad (16)$$
where $q_s(n - fsk - 1)|_{\text{ave}}$ is the average q_s of frame $(n - fsk - 1)$. Note that the adjustment of quantizer step is restricted to the next two coarser or finer step sizes; i.e., $|q_s(n, i_{MB} + 1) - q_s(n, i_{MB})| \leq 2$, where $q_s \in \{1 \cdots 31\}$.
5) Complete encoding frame n, then set new target frame rate $\hat{f}_{\text{target}} = f_{\text{target}} + 4 - [q_s(n)|_{\text{ave}}/4.0] + 0.5$, $b_{\text{enc}}(n) = b_{\text{enc}}(n - 1) + b(n) - R_c$ and go to 3).

Note that the coder determines the number of skipped frames based on the "buffer fullness" to avoid buffer overflow, but the picture quality is not considered. On the other hand, the purpose of skipping frames in our model-based VFR coding method is to reserve sufficient bits to maintain a reasonable coded video quality. To sum up, the VFR coding in H.263 (TMN) is aiming at buffer control only while our VFR coding is designed for both buffer control and picture quality control.

V. Simulation Results

A. Intraframe Coding

Four image sequences, *Salesman, Miss America (Missa), Claire,* and *Swing,* are concatenated to form a test sequence to demonstrate the rapid adaptation ability of our algorithms for different types of images and at scene changes. The first three sequences are "head and shoulders" type images and the last one is a computer graphics type image sequence. In our simulations, results using RM8 are also provided for comparison. It is denoted as RM8I because every frame is intracoded. The buffer size B_{enc} is set to be 6400 $\times$ 1.5 P for VFR and 6400 $\times P$ (RM8 specification) for RM8I when the channel rate is $P\times$ 64 kb/s. Fig. 4 shows the simulation results for intra frame coding at a channel rate of 12 $\times$ 64 kb/s.

It can be seen from Fig. 4(a) that both VFR1 and VFR2 can effectively control the coding process to produce good-quality coded pictures. Because VFR schemes skip frames when necessary, the mean values of their quantization scales are smaller and particularly the quantizer variation is much lower as compared to those of RM8I. Since the quantization scale is well controlled in a predefined range, the VFR approach performs consistently better in both numerical PSNR and in subjective image quality than the simple RM8I. The PSNR improvement over RM8I is approximately 3–6 dB as shown by Fig. 4(b).

The advantage of the VFR algorithm is that the frame rate is adjusted according to the picture content, as can be seen from Fig. 4(c). In the first subsequence, *Salesman,* the bit rate needed to encode pictures with reasonable quality at 30 frames/s is approximately twice the channel rate. Consequently, almost every other frame is skipped $(fsk = 1)$ [Fig. 4(c)]. The second subsequence, *Claire,* is a relatively

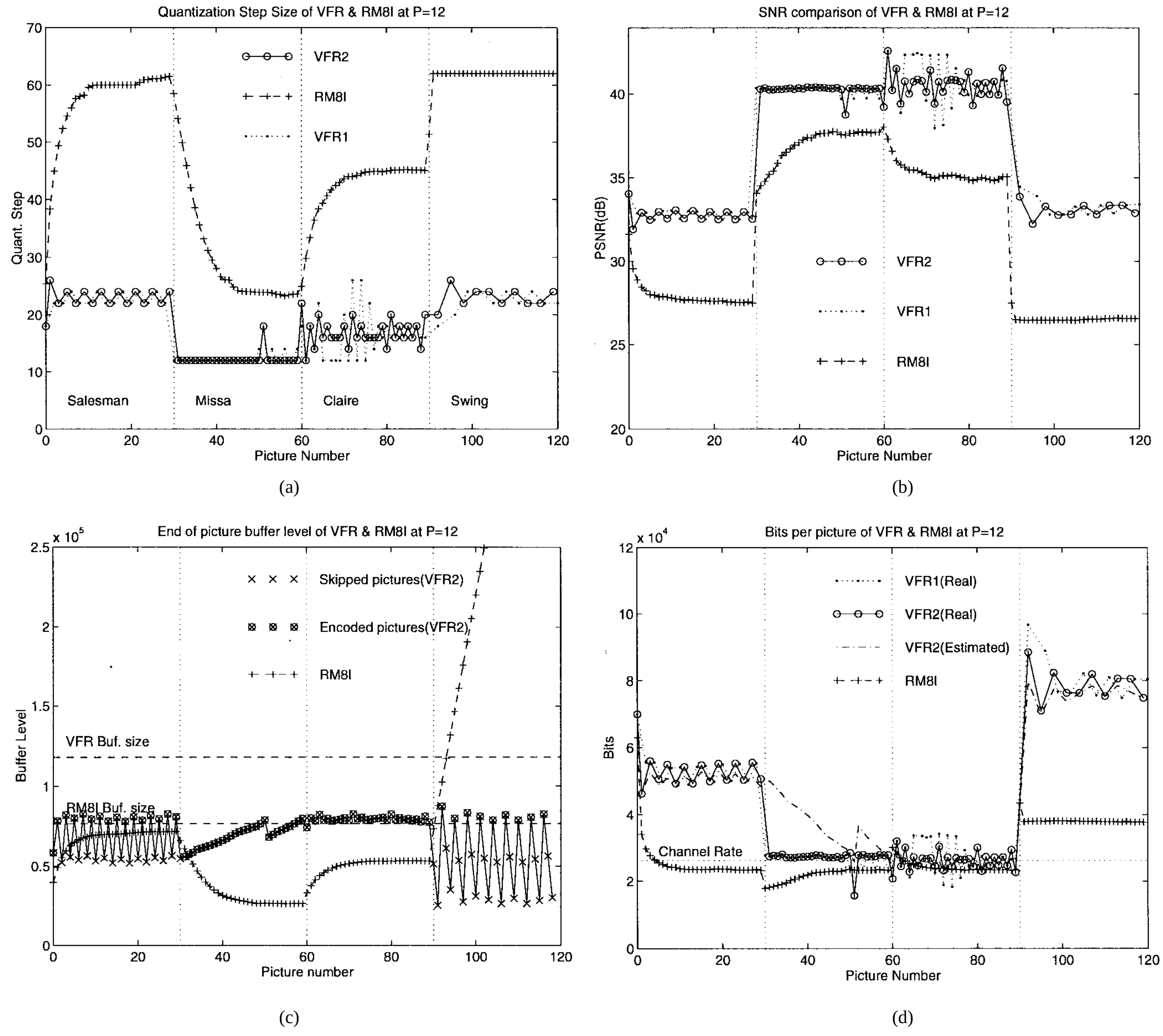


Fig. 4. Comparison of VFR2 and RM8I. (a) Quantization scales of VFR2 and VFR1 versus average q_s of RM8I, (b) PSNR, (c) end-of-picture buffer level, and (d) estimated and real coded bits per picture. Note that the differnce between the estimated and the real coded bits in the second sequence portion is not entirely the estimation error. It is mainly due to the fact that the lower limit of quantization scale is reached.

easy sequence to compress. Since the bit rate needed to produce good-quality pictures is just slightly higher than the given channel rate, the quantization scale has reached the prechosen lower limit and hence, it is running at the maximum rate of 30 frames/s. Because the accumulated bits in the buffer must comply with the HRD requirement, it forces an increase in quantization scale at the tail of this subsequence. At the beginning of the third subsequence, *Missa*, the quantization scale fluctuates slightly. The coder is adjusting the quantization scale to prevent the output buffer from underflow and overflow, and in the meanwhile, it maintains the coded image quality within a reasonable range. As the coding procedure proceeds, the variation of quantization scale (and also the range of PSNR and the coded bits per picture) is reduced and it becomes more stable. The fourth subsequence, *Swing*, pushes the quantization scale of RM8I to the upper limit at this channel rate. Even at the maximum quantization scale, it still produces many more bits than the channel can take; hence, the buffer overflows immediately. On the other hand, the VFR controller, though transmitting fewer pictures per second, can still encode this image sequence steadily. On the average, it skips two frames out of every three frames to satisfy the high bits demand of this image sequence.

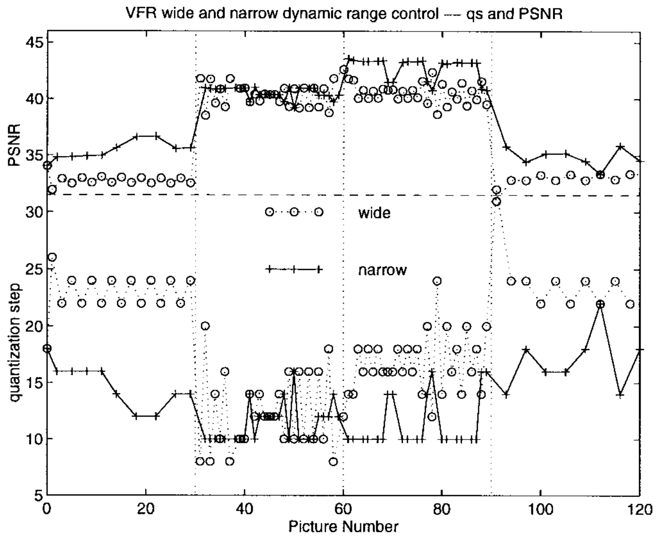
We next examine the accuracy of our source model, excluding the cases where the lower bound of quantization scale is hit. Although VFR2 is simpler in calculation, simulation results show that it performs well in bits estimation. The estimation error between the predicted bits and the real coded bits is found to be within 4.5% in the VFR2 simulations. As mentioned in Section III-B2, the bits estimation procedure in VFR2 is in fact more stable because the adjustable terms a_F and b_F in (12) are multiplicative factors. This can be observed from the fact that, at the beginning of the third subsequence, the quantization scale in VFR2 fluctuates less as compared to that of VFR1 [Fig. 4(a) and (b)]. This phenomenon is also reflected on the peak signal-to-noise ratio (PSNR) and the coded bits per picture. Although the pictures produced by VFR are slightly more jerky, they are subjectively better looking because of the higher and more uniform quality of every coded picture.

Another advantage of the VFR scheme is that one can select the admissible range of quantization scales to match the picture quality request. The admissible range can be either wide or narrow. As can be seen from Fig. 5, the quantization scales are controlled well for two selected ranges. In the case of wide admissible range, the coding quality vibrates more frequently and the variation is larger in magnitude. In contrast, a narrow admissible range forces a nearly constant image spatial quality but may skip more frames when necessary [Fig. 5(b)]; that is, less uniform temporal quality. Although quantization scales are expected to stay in a predefined range, there are occasions where the quantization step is driven out of the desired range due to buffer underflow constraint or the maximum skipped frame constraint. This happens more often when the admissible range is very tight. A tight range also results in higher buffer level variation.

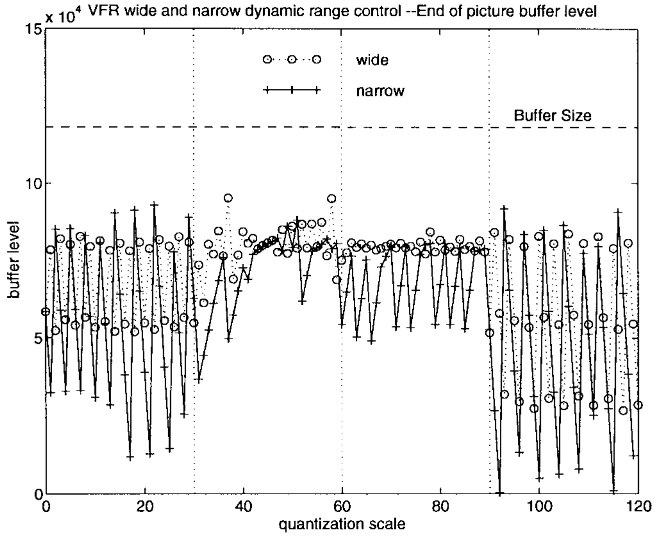
B. Intra/Inter Coding

The intra/inter coding performance of RM8, TMN, and our VFR algorithm is compared in this section.

1) Single Sequence: We apply RM8, TMN, and VFR2 algorithms to the *Salesman* sequence at a channel rate $P = 3$ and the results are shown in Fig. 6. For VFR2, the buffer size is set to $(P + 4) \times 6400 \times 2.5$, its $\mathrm{TBF}_{\mathrm{intra}}$ is two thirds of the buffer size and the $\mathrm{TBF}_{\mathrm{inter}}$, 40% of $\mathrm{TBF}_{\mathrm{intra}}$. These numbers are chosen based partially on the expected average bits per frame and partially on our experiences. They are not optimized. The target frame rate of TMN is 23, which approximately equals the frame rate of VFR2, and TBF$= P\times$ 6400, equal to the buffer size of RM8. At a constant frame rate, 30 frames/s, the coded bits per frame in RM8 is approximately equal to 6.4 k. Since every frame has roughly the same number of bits, the reconstructed picture quality is loosely inversely proportional to the picture coding complexity. As can be seen from Fig. 6(a), the quantization scales of RM8 for pictures 1–25, 50–70, and 110–130 are coarser; hence, the picture quality degrades during these periods. For TMN coding, because of its variable frame rate control, the coded picture quality is improved (about 1 dB over RM8). However, since the TMN coder determines the number of skipped frames based only on buffer state and the picture coding complexity is not considered in bit allocation, the number of skipped frames is quite regular and thus the shape of PSNR of TMN and RM8 are quite similar except for an offset of about 1 dB as shown in Fig. 6(b). In our VFR, the quantization scales are controlled to stay in a narrower range—12–18. During periods of rapid motion, it transmits fewer pictures (Fig. 6(b): frame skipped) and bits are allocated according to the coding complexity [Fig. 6(d)]; therefore, the coding quality of the



(a)

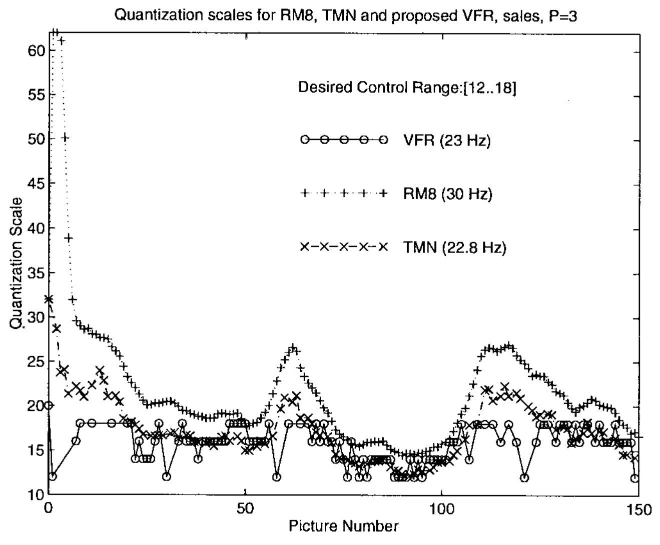


(b)

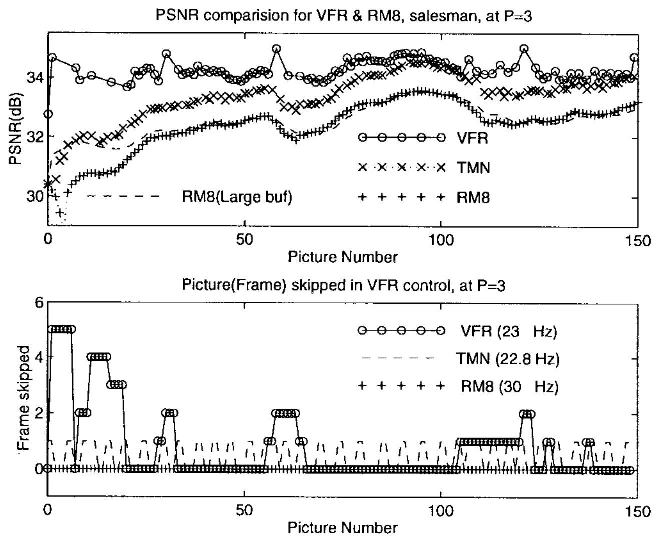
Fig. 5. Comparison of wide and narrow dynamic range control. (a) Quantization scales and PSNR and (b) end-of-picture buffer level. For clarity and ease of understanding, buffer fullness at each time interval is displayed and marked by circles and plus sign.

entire sequence is controlled rather well and is kept at a higher level as compared to that of TMN and RM8 (Fig. 6(b): PSNR). The tradeoff between these coding algorithms can easily be observed from Fig. 6(a) and (d). As Fig. 6(a) shows, approximately equal bit allocation to every picture in RM8 and TMN results in variation of coding quality. Our VFR scheme can achieve constant coding quality, because the variation of picture content is compensated by proper bit allocation [Fig. 6(d)].

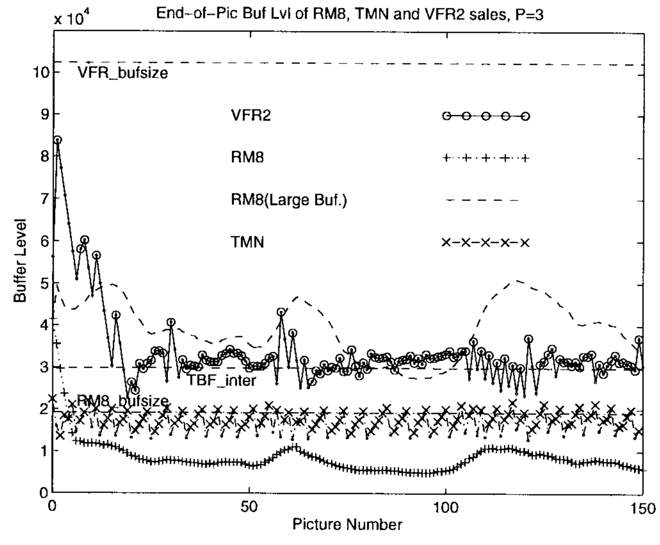
Another advantage of the VFR algorithm is the ability to handle scene changes. In our RM8 simulation, the quantization scale is adjusted to the maximum allowable value (=62) when the buffer overflows. This would occur after an intra picture is coded. In this situation, it needs a longer time (about 25 pictures) for the coder to reach a stable state—pictures coded with reasonable quality. For TMN, since frames are skipped so



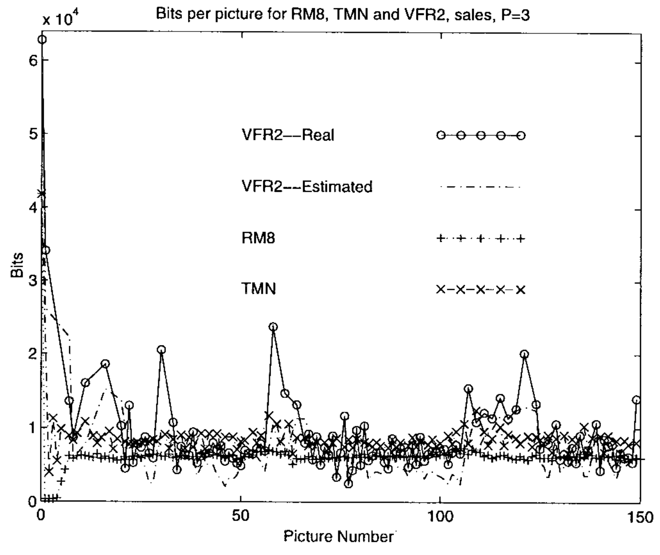
(a)



(b)



(c)



(d)

Fig. 6. Comparison of RM8, VFR, and TMN5 rate control performance for channel rate $P = 3$ on sequence "Salesman." (a) Quantization scales and (b) PSNR and frame skipped. Note that the value denotes the number of skipped frames starting at a certain picture number. For example, in VFR, five pictures (frames) are skipped between picture numbers 1 and 7. Hence, a value of five (marked by circles) is plotted for pictures 1 to 6. (c) End-of-picture buffer level and (d) estimated and real coded bits per picture

that the buffer fullness is kept near TBF and the quantization scale is adjusted according to (16), no abrupt change in quantization scale is observed. On the other hand, due to lack of coding complexity analysis in TMN, the resultant regular frame skipping [Fig. 6(b)] makes its PSNR curve almost the same the same as that of RM8; i.e., long adaptation time before reaching stable state. In our VFR, many pictures are skipped after encoding the first intracoded picture [Fig. 6(b) and (c)]. Though fewer pictures are transmitted, they are coded with good-quality right from the beginning when scene change occurs. Note that a good start-up very much helps the coder to easily reach and stay at a good coded quality thereafter. Many subjective tests report that a slightly jerky but better spatial resolution video is often preferred when compared to a faster refresh rate but poor spatial quality video.

The above simulations demonstrate that the bits estimation model together with the VFR control algorithm can track the variation of coding complexity functions rather well. The degradation of picture quality after scene change can thus be improved by this VFR control algorithm. In the periods of drastic image contents variation, the bits estimation error in the intra/inter coding mode can be up to 58%; however, the bits estimator still provides valuable information and the VFR encoder improves the coded picture PSNR over RM8 and TMN by about 1.9 and 0.9 dB, respectively. The variance and the mean of PSNR are listed in Table I. Most noticeable is that the PSNR variation of our VFR algorithm is much smaller than that the other algorithms.

2) Multiple Sequences: The significance and efficiency of our source model in handling scene changes can be further

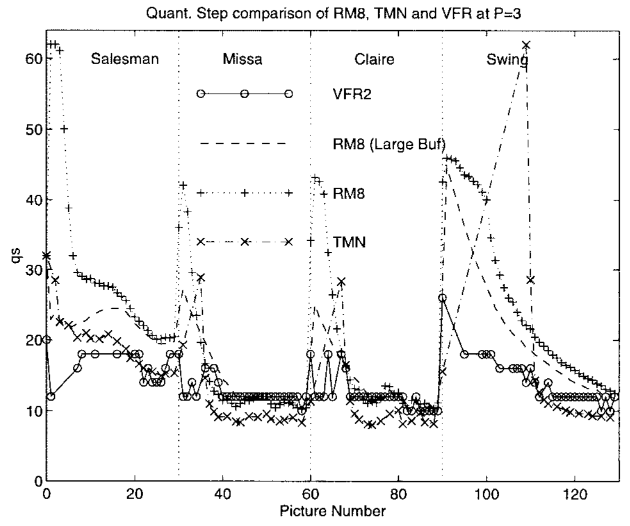
TABLE I
VARIANCE AND MEAN PSNR OF RM8, TMN, AND THE PROPOSED VFR ALGORITHM PERFORMED ON THE SALESMAN SEQUENCE

	RM8 (30 Hz)	TMN (22.8 Hz)	VFR (23 Hz)
Variance	0.81	0.67	0.11
Mean PSNR	32.36	33.34	34.24

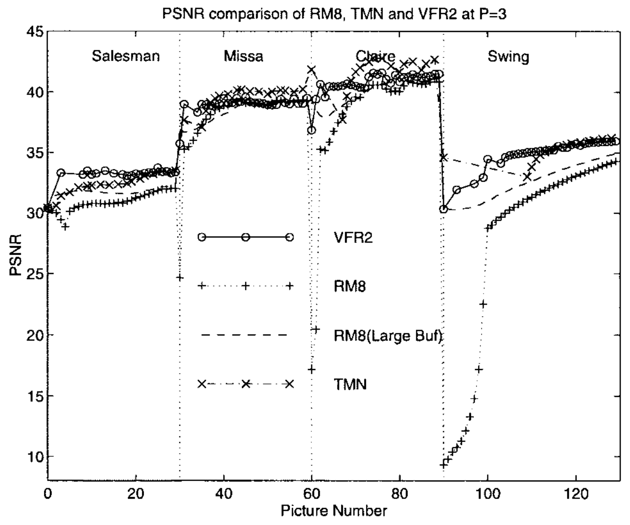
demonstrated by encoding the concatenated test sequence as shown in Fig. 7. The interframe VFR buffer control strategy has been described in Section III-C1. In our VFR algorithm, the quantization scales are controlled to stay inside a selected narrow range, and thus four different subsequences can all be well coded without being affected by scene changes [Fig. 7(a)]. In TMN, there is no buffer size specification, and no buffer state feedback information is used to adjust quantization scales; hence, its buffer fullness can become very high as can be seen from Fig. 7(c). Also, the skipped picture number and the quantization scale variation may be unreasonably high after a scene change. The PSNR produced by our VFR is maintained at a nearly constant level even right after scene changes. In contrast, it takes a significantly longer period of time for the TMN and the RM8 coders to re-enter the normal image quality state [Fig. 7(b)].

Note that the buffer size used by VFR is larger than that used by RM8. It may be speculated that the coding performance of RM8 is limited by its small buffer size. From our simulation, the degradation due to scene changes can be improved to some extent if the RM8 buffer size is enlarged. For example, in Fig. 7(b) when the RM8 buffer is five times its normal size (so that it is comparable to the VFR buffer size), the degradation of scene changes is reduced (another example is in Fig. 6(b): PSNR). However, the average coding quality with large buffer size is not much improved as compared with that with normal buffer size [Figs. 6(b) and 7(b)]. This is because although the range of buffer dynamic is enlarged, the coding quality is still constrained by the average bits per frame when the frame rate is fixed. The VFR algorithm together with the new target buffer fullness control strategy (15) can utilize the large buffer space to eliminate the scene change degradation, and at the same time, the HRD buffer requirements are satisfied. Fig. 7(c) shows the buffer behavior that helps explain the effect of various buffer control schemes.

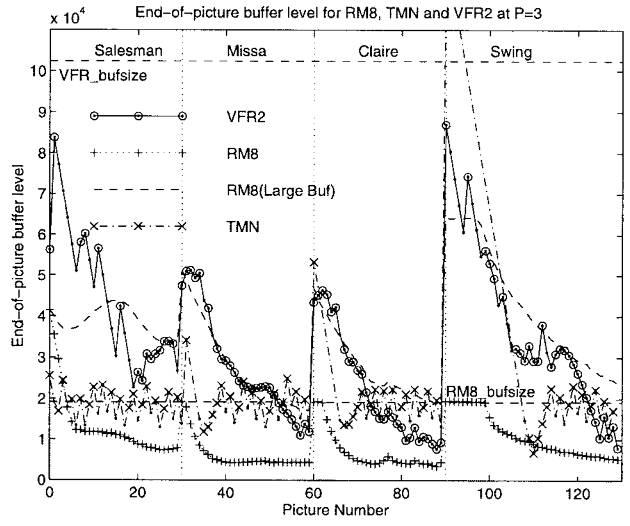
At very low rates, precise bit rate control becomes very critical. This is illustrated by comparing the coding performance of VFR2 and RM8 (with 30 frames/s) on the concatenated sequence for $P = 2$ and $P = 3$. The advantage of VFR is particularly noticeable at low bit rate coding, as can be seen from Table II where the PSNR improvement of VFR over RM8 and TMN for channel rate at $P = 2$ is higher than that at $P = 3$. More importantly, the PSNR of VFR coding is maintained at almost a constant level for a particular image sequence for a given channel rate. The only exception appears at the beginning of the entire sequence, where the default q_s is used to intra-encode the whole picture. Consequently, the PSNR of the first frame is significantly lower than that of the other frames. Therefore, the PSNR variance and peak difference are much larger in the first subsequence (*Salesman*). This start-up situation can be improved if the initial quantizer



(a)



(b)



(c)

Fig. 7. Comparison of RM8, VFR, and TMN5 rate control performance for channel rate $P = 3$ on multiple sequences. (a) Quantization scales, (b) PSNR, and (c) end-of-picture buffer level.

TABLE II
MEAN, VARIANCE, AND PEAK DIFFERENCE OF PSNR FOR THE CONCATENATED SEQUENCE AT $P = 2$ AND $P = 3$

A: RM8, B: RM8 (large buffer), C: TMN and D: Our VFR

		Mean (dB)				Variance (dB)				Peak diff.(dB)			
		A	B	C	D	A	B	C	D	A	B	C	D
Sales	P=2	29.79	30.66	31.38	33.33	1.30	0.28	0.23	2.47	4.80	0.57	1.74	4.67
	P=3	30.88	31.72	32.34	34.01	0.55	0.11	0.75	0.15	3.17	1.83	3.14	1.84
Claire	P=2	35.78	37.26	38.62	38.93	29.92	0.87	1.15	0.06	20.45	2.79	3.52	0.78
	P=3	38.04	38.38	39.54	39.35	7.50	0.79	0.82	0.08	14.7	2.54	3.12	1.62
Missa	P=2	34.26	38.07	39.79	39.70	61.66	0.70	1.86	0.34	24.74	2.70	4.92	2.16
	P=3	38.13	39.88	41.71	41.09	29.60	1.04	1.63	0.28	23.88	3.35	5.07	2.49
Swing	P=2	23.34	27.49	33.12	33.72	81.11	43.20	0.75	1.68	23.17	21.01	2.81	3.33
	P=3	27.26	32.51	35.22	35.16	72.14	2.42	0.81	0.41	25.01	4.66	3.21	2.64

scale is selected based on the computed image variances as stated in Section III-B. For easy sequences such as *Claire* and *Missa*, the channel rates are sufficient for nearly maximum frame rate transmission; hence, the lower limit of q_s is frequently encountered, which leads to a rather high average PSNR. On the other hand, *Salesman* and *Swing* are difficult to compress. In order to skip fewer frames, the higher limit of q_s is often used, which leads to a lower average PSNR. As a result, it is quite interesting that there exist nearly two PSNR values, 34 and 40 dB, in VFR coding. In the case of RM8 and TMN, the PSNR is controlled mostly by the average bits per picture and is thus strongly picture dependent. Its PSNR values are unpredictable.

VI. CONCLUSIONS

In the first part of this paper, we derive a source model that describes the relationship between bits, distortion, and quantization step size for transform coders. The coding behavior can thus be predicted if all the component variances are available. This model enables us to estimate the bits needed to encode a picture at a given distortion or to adjust the quantization scale of a coder at a given bit rate.

The second part of this paper develops a variable frame rate coding algorithm based on the proposed bits model. The almost constant picture quality is ensured by skipping frames to preserve enough bits for every coded picture. Unlike the previous study [8] that looks for an appropriate constant frame rate for a particular image sequence or that skips frames based only on buffer fullness (TMN), the frame rate in our scheme is self-adjusted according to both the current picture content and the buffer status. This coding control algorithm is quite simple and cost-effective for hardware realization. Compared to the well-known RM8 (an H.261 video coder) an improved PSNR performance at about 3–6 dB is demonstrated using the variable frame rate coding/decoding. However, the most distinct feature of VFR coding is its ability to produce a uniform quality coded picture (sequence) that neither RM8 nor TMN can achieve.

Analysis of the real coded bits and the estimated bits reveals that bits estimation errors are due to the facts that: 1) the stationarity and ergodicity assumptions of signal source are not completely satisfied for real pictures, and 2) our VFR system is a one-pass system, and hence, the uncertainty in prediction cannot all be removed. In addition, we adjust our quantization scale only once per frame. The best quantization scale to achieve the bits budget could be a fractional number. However, the quantization scale in H.261 is restricted to even integers. Therefore, it is, under this limitation, not possible to control the bit rate exactly. This situation can be improved by adjusting the quantization scales several times in the middle of a picture. The concept and approach of variable frame coding proposed here may be further extended to control multiple video sources over a shared channel. This topic is now under development.

REFERENCES

[1] CCITT, Working Party XV/1, “Draft of recommendation H.261: video codec for audiovisual services at $P \times 64$k bits/s,” July 1990.

[2] W. B. Pennebaker and J. L. Mitchell, *JPEG—Still Image Data Compression Standard.* New York, NY: Van Nostrand Reinhold, 1993.

[3] MPEG (Motion Picture Experts Group, ISO–IEC JTC1/SC29/WG11), “Coding of Moving Pictures and Associated Audio,” CD 11172-2 Nov. 1991.

[4] L. Wang, “Bit rate control for hybrid DPCM/DCT video codec,” *IEEE Trans. Circuits Syst. Video Technol.,* vol. 4, pp. 509–517, Oct. 1994.

[5] CCITT, Working Party XV/4, “Description of Reference Model 8 (RM8),” June 9, 1989.

[6] A. Puri and R. Aravind, “Motion-compensated video coding with adaptive perceptual quantization,” *IEEE Trans. Circuits Syst. Video Technol.,* vol. 1, pp. 351–361, Dec. 1991.

[7] W.-Y. Sun, H.-M. Hang, and C. B. Fong, “Scene adaptive parameters selection for MPEG syntax based HDTV coding,” in *Signal Processing of HDTV,* L. Stenger *et al.*, Eds. Elsevier Science, 1994, vol. V.

[8] Y. Takishima, M. Wada, and H. Murakami, “An analysis of optimal frame rate in low bit rate video coding,” *IEIEC Trans. Commun.,* vol. E76-B, no. 11, pp. 1389–1397, Nov. 1993.

[9] ITU-T, Study Group 15, “Recommendation H.263,” Nov. 1995.

[10] Video Codec Test Model for H.263 (TMN5), Telenor Research, Jan. 1995.

[11] D. W. Lin, M. H. Wang, and J. J. Chen, “Optimal delayed-coding of video sequences subject to a buffer-size constraint,” in *Symp. Visual Communication Image Processing ’93,* Cambridge, MA, Nov. 1993, vol. 2094, pp. 223–234.

[12] A. Ortega, K. Ramchandran, and M. Vetterli, “Optimal trellis-based buffered compression and fast approximation,” *IEEE Trans. Image Processing,* vol. 3, pp. 26–40, Jan. 1994.

[13] 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, pp. 533–545, Sept. 1994.

[14] H.-M. Hang, “Adaptive buffer/quantizer control for transform video coders,” U.S. Patent: 5 038 209, Aug. 1991.

Jiann-Jone Chen, for a photograph and biography, see this issue, p. 298.

Hsueh-Ming Hang (S’79–M’84–SM’91), for a photograph and biography, see this issue, p. 298.