Asynchronous Rate Control for Multi-Object Videos

Yu Sun, *Member, IEEE,* and Ishfaq Ahmad, *Senior Member, IEEE*

***Abstract*—Object-based coding can potentially achieve a higher degree of compression and better visual quality. The objects in a scene are not always synchronous, that is, they have different temporal resolutions. In some applications, it is more efficient to transmit asynchronous objects with different temporal rates so as to achieve a better tradeoff between temporal and spatial resolutions. This requires to balance the qualities of individual objects while ensuring an overall visual quality with a given bit rate. This paper proposes a rate control algorithm for multiple video object encoding, which is suitable for both synchronous and asynchronous transmissions. The algorithm aims to maximize scene quality with an accurate bit rate, while efficiently handling buffer fullness. Using an efficient bit allocation strategy, the algorithm achieves accurate target bit rates, provides good coding quality, and decreases buffer overflow/underflow. The proposed algorithm also allows flexible priority adjustment among multiple objects to ensure overall better visual perception. Designed primarily for asynchronous objects, the algorithm treats the synchronous objects as a special case. Experimental results demonstrate that, when giving suitable asynchronous video object planes rates, the proposed algorithm achieves good temporal-spatial tradeoff while yielding accurate rate regulation and effective buffer control.**

***Index Terms*—Asynchronous video objects, bit allocation, MPEG-4 video coding, object streams, priority adjustment, rate control (RC), synchronous video objects (VOs).**

I. Introduction

THE frame-based video rate control (RC) problem is well studied. Several solutions now exist for various video coding standards and applications, for example, storage media with MPEG-1 and MPEG-2 [1]–[5], video conference with H.261 and H.263 [6]–[8]. In object-based videos, such as, the one supported by MPEG-4 [9], a video object (VO) of a scene may be individually coded and may correspond to an elementary bitstream that can be individually accessed, manipulated and transmitted, while the information regarding the interobject relationship is sent in a separate stream [10]. A RC algorithm has to distribute the bits among different VOs according to their quality requirements. Exploitation of the individual characteristics of each object can also potentially improve coding efficiency, while offering additional flexibility and functions [11], [12].

How to effectively distribute bits among different objects in a scene is the core issue in RC for object-based video coding. A very straightforward way to achieve RC for object-based coding is to assign a predefined bitrate to each VO, e.g., at a constant rate, and code it independently, without sharing resources among objects. Then the output bitrate is the sum of the individual rates for various VOs [12]. However this approach is not the best way of distributing the bitrate since the characteristics of objects (e.g., the size, the complexity) are not considered.

A few RC algorithms for VO-based coding have been proposed [12]–[20]. Nunes and Pereira proposed a joint RC algorithm for scenes with multiple VOs [12]. The algorithm distributes the available bitrate among various VOs by considering their varying characteristics. Lee *et al.* have proposed an algorithm that is scalable for various bit rates, spatial and temporal resolutions, the distribution of the bit budget within a frame is proportional to the square of mean absolute difference (MAD) of each VO [13], [14]. Vetro *et al.* developed a scheme for multiple VOs in which the total target bits of a frame are distributed proportionally to the relative size, motion and variance of each object [15], [16]. Ronda and Eckert regarded multiobject RC as an optimization problem, and proposed several cost criteria as goals to be optimized [17]. These algorithms consider the coding complexity for each object for deciding the target bits among objects within a frame, but they do not take into account the total coding complexity for a frame. They make rate allocation decisions for the current frame only according to the actual bits used for previous encoded frames/[video object planes (VOPs) in MPEG-4], average bits available for the remaining frames, etc. This can lead to unsuitable bit allocation and visual quality may not be optimum. Based on the above observations, our previous work [18] estimates the bit budget of a frame according to its global coding complexity, and dynamically distributes the target bits for each object based on its coding complexity.

All of the above algorithms assume VOs are synchronous, meaning that all VOs are encoded with the same VOP rate. However, this assumption does not always hold because several VOs in a scene may have different temporal resolutions, which is called asynchronous VOs, to improve coding efficiency. For example, objects with fast movements may have higher VOP rates while objects characterized by slow movements may have lower VOP rates. Proper asynchronous RC can provide significant savings in bits, and naturally suit object-based video coding. Multiple objects may be considered equally important if the goal is to obtain the best possible global quality [17], [19]. However, when video data is transmitted over low bandwidth channels, a uniform quality among multiple objects cannot be obtained due to inadequate bits. In some applications, it may be more efficient to transmit objects with different importance specified by users. Under these cases, a RC algorithm should distribute adequate bits to enable important objects to obtain higher qualities given the available bit rate constraint, and transmit the other objects

Manuscript received February 21, 2003; revised May 21, 2004. This paper was recommended by Associate Editor Q. Tian.

Y. Sun is with the Department of Computer Science, University of Central Arkansas, Conway, AR 72035 USA (e-mail: yusun@uca.edu).

I. Ahmad is with the Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019 USA (e-mail: iahmad@cse.uta.edu).

Digital Object Identifier 10.1109/TCSVT.2005.852415

with the less qualities. For example, a foreground object in a scene may be assigned a higher priority than a background object. In some cases, the RC scheme may not even have to encode the nonimportant objects in a scene to match the output rates to the channel constraints.

A few researchers have studied the RC problem for multiple objects with different temporal resolutions. Aiming to obtain better spatial-temporal tradeoff, the work by Nunes and Pereira presented a scene level RC algorithm performing bit allocation for several VOs encoded at different VOP rates [20]. Hung and Lin [21] considered joint rate-distortion (R-D) coding of multiple videos over multiple frames for transmission over a shared channel, where the videos may have different frame rates. Lee and Vetro proposed bit-allocation algorithms that consider the tradeoff between coded quality and temporal rate [22]. However, the algorithms [21], [22] need to look ahead a number of future frames, and thus are not suitable for real-time, low-delay and low-complexity applications.

Since the building up of any mathematical models depends on the specific channel models and statistic characteristics of video signals, most R-D models are approximate models. An effective way is to find a comprehensive scheme to overcome the fundamental weakness of the inaccurate prediction of R-D models. This can be done by exploiting various feedback information to compensate the prediction deviations, in the following manner: First, estimate coding properties and predict target bit budget before encoding; Second, combine various feedback information to rapidly compensate estimated deviations after encoding, aiming to reduce the effect of random disturbance and the error caused by the variance between the real system and its statistical model.

Current MPEG-4 RC schemes for multiple objects, e.g., [20], adopt the same bit allocation strategy, that is, they first allocate target bits to a scene for each encoding time, and then distribute the allocated bits among several VOs in this scene. This kind of bit allocation method is inconsistent with MPEG-4's object-based concept. It may be suitable for synchronous RC, but is not effective when used in asynchronous case for multiple objects. Since the number of objects in a scene keeps varying along the coding time, and the VOP types of multiple objects may also different in one coding time instant, the allocation of target bits is a more complex problem.

This paper proposes a bit estimation algorithm, named asynchronous rate control (ARC), for asynchronous multiple object RC. ARC divides objects into different "object streams," such that each VO forms one *relatively independent* "object stream" along the coding time. Thus, the overall asynchronous problem is decomposed into multiple sub-problems, with each sub-problem regarded as a single-object RC problem. Similar to a single object encoding case, each VO uses its own R-D model. The major advantage of this approach, which is consistent with MPEG-4's object-based concept, is that at the top level the proposed algorithm dynamically distributes the bit budget among multiple "object streams" at each encoding time by jointly adjusting visual qualities of multiple objects. At the same time, at the lower level one can adopt efficient individual single-object or frame-level RC schemes to solve each stream's RC issue and simplify the traditional bit allocation method. Combined with an efficient bit allocation strategy and a PID buffer controller, the proposed ARC algorithm strikes a balance among the qualities of multiple objects, and allows adjusting their priorities. In addition, ARC is also equally suitable for synchronous mode. Obviously, the architecture of ARC is quite different from that of current MPEG-4 RC schemes, such as [20].

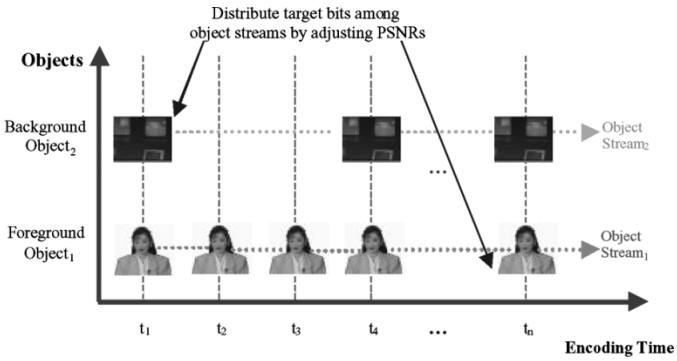

Fig. 1. Asynchronous RC for multiple VOs.

This paper is organized as follows. Section II presents the step-wise description of the ARC algorithm as well as the principles and foundations of these steps. Section III describes priority adjustment among objects. Section IV includes the experimental results demonstrating the performance of the proposed algorithm. Finally, Section V concludes the paper by providing some final remarks and observations.

II. ARC Algorithm

To obtain a better tradeoff between spatial and temporal qualities for video sequences, multiple objects in a scene may have different temporal resolutions. An example of multiple objects encoding with different VOP rates is shown in Fig. 1, Foreground Object$_1$ with fast motion has a higher VOP rate, while Background Object$_2$ with slow motion has a lower VOP rate. In this case, the number of VOPs to be encoded at each encoding time instant is different. For example, there are 2, 1, and 1 VOPs that need to be encoded at time t_1, t_2, and t_3, respectively. However, the RC scheme in VM8 [23] and the previous RC strategies for MPEG-4 [12]–[18] can only work for synchronous VOs since they assume multiple VOs have the same VOP rate, namely, the number of VOPs to be encoded at each encoding time is a constant factor. Fig. 1 presents the basic idea of asynchronous RC scheme for multiple VOs. In this section, we describe the principles and foundations of the proposed algorithm.

A. Initialization Stage

This stage includes setting up the buffer size and encoding parameters. To encode the first I-VOP for each object, an initial QP is given. Once the first VOP has been coded for each object, we can obtain initial visual quality for each object, actual bits used to encode it, etc.

B. Target Bit Estimation

The algorithm regards each object forms an "object stream" along the coding time, for example, Foreground Object$_1$ at time

$t_1, t_2, \ldots t_n$ forms an "Object Stream$_1$ " in Fig. 1. The algorithm estimates target bits and dynamically controls bit allocation among multiple "object streams".

1) Bit Ratio Computation for Object Streams: The algorithm calculates the average number of bits $\bar{A}_{i,t}$, which is used to encode per VOP_i during the previous coding time period before the current encoding time t

$$\bar{A}_{i,t} = \left(\sum_k A_{i,k} \right) / n_i \quad (k = t-1, \ldots, t-n_i-1) \tag{1}$$

here, $A_{i,k}$ represents the actual bits used to code VOP_i at time $k, t-1$ is the first previous encoding time for VOP_i before $t, t-n_i-1$ is the n_i^{th} previous encoding time, n_i represents the number of VOP_i in a given time period. For example, if this time period is set to 1 second and the encoding rate for VO_i is 15 VOP/s, then $n_i = 15$.

Considering different objects have different encoding VOP rates, the algorithm computes the previous bit ratio $L_{i,t}$ for VOP_i at time t

$$L_{i,t} = \frac{\mathrm{VR}_i \cdot \bar{A}_{i,t}}{\sum_{j=1}^{M} (\mathrm{VR}_j \cdot \bar{A}_{j,t})} \tag{2}$$

where VR_i means the encoding VOP rate for VO_i, M is the number of objects.

2) Initial Target Bit Estimation: If VO_i appears at the current encoding time t, then according to its VOP type and previous bit ratio, its initial target number of bits $\bar{T}_{i,t}$ is set to a weighted average bit count

$$\bar{T}_{i,t} = \frac{\beta_{i,t}^d}{\sum_{l=0}^{D_i} \left(\beta_{i,t}^l \cdot N_{i,t}^l \right)} \times L_{i,t} \times R_{r,t} \tag{3}$$

where $R_{r,t}$ is the remaining number of bits at time t, D_i is the number of VOP types for VO_i, $N_{i,t}^l$ is the remaining number of VOPs with VOP type l for VO_i at time t $\beta_{i,t}^l$ is the weight factor of VOP type l for VO_i at time $t, l = 0, 1, 2$ indicate I-VOP, P-VOP, and B-VOP respectively, thus $\beta_{i,t}^l$ represents $\beta_{i,t}^0, \beta_{i,t}^1$, and $\beta_{i,t}^2$, and $\beta_{i,t}^d$ is the weight corresponding to the current VOP type d for VO_i at the current time t.

3) Weight Adjustment Among Multiple Objects: To achieve comparable and balanced qualities among multiple objects, and to avoid large perceptual quality differences among them, the algorithm allows adjusting the weight for each object. The larger the weight for VO_i, the more target bits should be allocated to it. An intuitive way to adjust each object's weight is: the coding quality $Q_{i,t-1}$ of VOP_i at the previous coding time $t-1$ is compared with the average peak signal-to-noise ratio (PSNR) for all objects coded at $t-1$. If $Q_{i,t-1}$ is lower than this average PSNR, the weight of VOP_i at time $t, W_{i,t}$, is increased a little. Thus VOP_i would obtain more bits during target bit allocation to achieve a higher quality; otherwise, $W_{i,t}$ is decreased a little and achieves a lower quality. Initially, the weight for each object is set to 1.0. The weighted average PSNR $\bar{Q}_{t-1}$ considering size factor at time $t-1$ is

$$\bar{Q}_{t-1} = \frac{\sum_{j=1}^{M} (V_{j,t-1} \cdot Q_{j,t-1})}{\sum_{j=1}^{M} V_{j,t-1}} \tag{4}$$

where $V_{j,t-1}$ is the number of nontransparent *MBs* in VOP_j at time $t-1$, representing its size. The weight for VOP_i at time $t, W_{i,t}$, can be adjusted by

$$W_{i,t} = W_{i,t-1} \times \left(\frac{\bar{Q}_{t-1}}{Q_{i,t-1}} \right)^{\Gamma} \tag{5}$$

where Γ is set to 2 in the experiments. Equation (5) means if $Q_{i,t-1}$ is lower than $\bar{Q}_{t-1}, W_{i,t}$ is increased and VOP_i would get more bits during target bit allocation, thus obtain a higher PSNR; otherwise, $W_{i,t}$ is decreased a little and gets a lower PSNR.

Then the normalized weight for $\mathrm{VOP}_i, W'_{i,t}$, is calculated by

$$W'_{i,t} = W_{i,t} \Big/ \sum_{j=1}^{M} W_{j,t} \, . \tag{6}$$

4) Object-Level Coding Complexity Analysis: Since the characteristics of each VO change along time, it is very important to analysis them before allocate bits to the VO. The coding complexity indicates how many bits would be appropriate for VOPs before really encoding them. The coding complexity measure is given as

$$C_{i,t} = \sum_{m=1}^{V_{i,t}} \sqrt[k]{\sum_{j=1}^{n_m} (P_{m,j} - \bar{P}_m)^2 / n_m} \tag{7}$$

where $C_{i,t}$ is the coding complexity of VOP_i at time $t, P_{m,j}$ is the luminance value of pixel j in the mth macroblack (MB) of a motion-compensated residual $\mathrm{VOP}_i, \overline{P_m}$ is the arithmetic average of luminance residue of MB_m, n_m is the number of nontransparent pixels in MB_m, k is set to 4 in the experiments.

The coding complexity computed by (7) naturally combines the object size $(V_{i,t})$ and average variance of each *MB* in a VOP, and thus, can reflect the instantaneous characteristics of this VOP. According to it, appropriate bits can be allocated to the VOP and the encoder's performance can be improved, since it allows bits to be saved in easy scenes so that more bits can be used for difficult scenes.

A normalized coding complexity of VOP_i at time $t, C'_{i,t}$, can be obtained by

$$C'_{i,t} = W'_{i,t} \cdot C_{i,t}. \tag{8}$$

The average complexity of previous n_i time instants for VOP_i before time t can be computed by

$$\bar{C}_{i,t} = \left(\sum_k C'_{i,k} \right) / n_i \quad (k = t-1, \ldots, t-n_i-1). \tag{9}$$

5) Target Bits Adjustment: Considering the coding complexity of VOP_i, its target bits budget, $T_{i,t}$, is then estimated by

$$T_{i,t} = (C'_{i,t}/\bar{C}_{i,t}) \cdot \bar{T}_{i,t}. \tag{10}$$

The number of target bits is estimated only for P-VOP and B-VOP. We do not estimate target bits for I-VOPs [18]. Equation (10) is determined for the following reason: If the normalized complexity of VOP_i is higher than its average complexity, more bits should be allocated to it than its weighted average bits $\bar{T}_{i,t}$. On the contrary, if its normalized complexity is lower than the average complexity, fewer bits should be allocated. Therefore, proper bits can be adaptively allocated to VOP_i and coding quality can be kept more constant.

C. Buffer Control Strategy

To get a more accurate target bit estimation, the initial bit target is further fine tuned based on the buffer fullness. The goal of the buffer control is trying to keep buffer fullness in the middle level of buffer size to reduce the possibilities of buffer overflow or underflow [14]. The basic way to do this is to decrease the number of target bits a little when the buffer fullness is higher than the middle level; whereas the number of target bits is increased to some extent if the buffer fullness is lower than the middle level.

However, the VM8 solution and other algorithms exploit a simple proportional buffer control to adjust the initial estimated target bits, when the target bits plus the current buffer fullness is larger than the predefined safety margin of buffer size, the target bits must be cut down abruptly by the number of superfluous bits no matter how the current frame really needs these target bits, this may result in sudden degradation of visual quality. The control ability of this buffer technique is not effective enough, as shown in the experiment, when the complexity of a sequence changes drastically, the buffer tends to be out of control, especially in low bit rate cases.

Here, we adopt our proposed proportional-integral-derivative (PID) buffer control technique (see Fig. 2) [18]. PID technique is famous and largely exploited in automatic process control area, its popularity is mainly attributed to its simplicity and high accuracy on automatic control [24], [25]. Our goal is to continuously and adaptively adjust the initial target bits to avoid buffer overflow/underflow, thus avoid coding quality's sudden degradation or surge. Unlike VM8, we do not set any safety margins for buffer control since this PID buffer controller has more advanced control ability.

The PID buffer adjusting factor is computed as

$$\mathrm{PID}_t = K_p \times E_t + K_i \times \int E_t \cdot dt + K_d \times \frac{dE_t}{dt} \quad \text{with}$$
$$E_t = \frac{(B_s/2 - B_{f,t})}{B_s/2} \tag{11}$$

where B_s is the buffer size, the variable E_t represents the error signal at time t, which measures the deviation between the target buffer fullness $B_s/2$ and the current buffer fullness $B_{f,t}$. K_p, K_i, and K_d are the proportional, integral and derivative control parameters, respectively, and are empirically set to 1.0, 0.05, and 0.9, respectively, in the experiments. The term $K_P \times E_t$ in (11) is the proportional controller which can reduce the error between the current buffer fullness and the target buffer fullness, but cannot fully eliminate this error. The integral controller, $K_i \times \int E_t \cdot dt$, has the effect of eliminating the steady-state error, but it may make the transient response worse. The derivative controller, $K_d \times (dE_t)/(dt)$ can increase the stability of the system, reducing the overshoot, and improving the transient response. The three-mode PID controller combines the advantages of each individual controller and thus, is more smooth and effective.

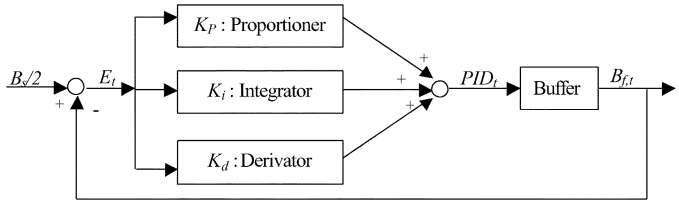

Fig. 2. PID buffer control system.

The final target bits for VOP_i after buffer adjustment is

$$T_{i,t} := T_{i,t} \cdot (1 + \mathrm{PID}_t). \tag{12}$$

To maintain a minimum acceptable visual quality for each VOP, the encoder must at least allocate the minimum number of bits: $T_{i,t} = \max\{\bar{A}_{i,t}/4, T_{i,t}\}$. For most applications, overflow is much worse than underflow, so maximum bits should be more strictly constrained than the minimum ones. To avoid buffer overflow, the maximum number of bits is given as: $T_{i,t} = \min\{2 \cdot \bar{A}_{i,t}, T_{i,t}\}$.

D. Quantization Parameter Calculation, Encoding, and Updating

Chiang and Zhang proposed a quadratic model that describes the relation between the required bits for texture coding and QP [14]. Once the number of target bits $T_{i,t}$ for VOP_i is obtained, QP for P-VOP and B-VOP can be computed based on the model of each VO [14], [15]

$$\frac{T_{i,t} - H_{i,t-1}}{\mathrm{MAD}_{i,t}} = \alpha 1_{i,t} \times \mathrm{QP}_{i,t}^{-1} + \alpha 2_{i,t} \times \mathrm{QP}_{i,t}^{-2} \tag{13}$$

where $\mathrm{MAD}_{i,t}$ is computed using motion-compensated residual for the luminance component, $\mathrm{QP}_{i,t}$ denotes quantization level used for $\mathrm{VOP}_i, \alpha 1_{i,t}$, and $\alpha 2_{i,t}$ are the first and second order model coefficients, $H_{i,t-1}$ denotes the number of bits actual used for coding the motion, shape, and header for VOP_i at its last coding time $t-1$.

The quantization parameter (QP) of I-VOP for an object is obtained as follows [18]:

$$\mathrm{QP\_}I_{i,t} = \overline{\mathrm{QP}}_{i,t} + \delta_{i,t} \tag{14}$$

where $\mathrm{QP\_}I_{i,t}$ is the QP of the current $I-\mathrm{VOP}_i$, $\overline{\mathrm{QP}}_{i,t}$ is the average QP of l intercoded VOP_i's before the current $I-\mathrm{VOP}_i$. Initially, $\delta_{i,t}$ is 1.0 and updated as

$$\delta_{i,t} := \delta_{i,t} + \frac{\mathrm{PSNR\_}I_{i,t1} - \overline{\mathrm{PSNR}_{i,t1}}}{\lambda} \tag{15}$$

where $t1$ is the coding time of the last $I-\mathrm{VOP}_i$, $\mathrm{PSNR\_}I_{i,t1}$ is the PSNR of the last $I-\mathrm{VOP}_i$ and $\overline{\mathrm{PSNR}_{i,t1}}$ is the average PSNR of l intercoded VOP_is before the last $I-\mathrm{VOP}_i$, λ is a tuning parameter. l and λ are empirically chosen to be 3 and 16, respectively, for all coding conditions, the simulation results are not very sensitive to the specific values of l and λ.

The encoder codes VOP_i when its QP is obtained. The buffer fullness is then modified after encoding, the number of bits which was used for the current VOP is added to the current buffer level, at the same time, the weighted average number of bits for VOP_i to be output from the buffer at encoding time t, $Bpp_{i,t}$, is decreased from the buffer fullness. $Bpp_{i,t}$ can be computed by

$$Bpp_{i,t} = \frac{\beta_{i,t}^{d} \cdot L_{i,t}}{\sum_{l=0}^{D_i} \left(\beta_{i,t}^{l} \cdot N_{i,t}^{l}\right)} \times R_{r,t}. \tag{16}$$

Then, the buffer fullness can be modified as follows:

$$B_{f,t} := B_{f,t} + A_{i,t} - Bpp_{i,t}. \tag{17}$$

E. VOP-Skipping Control

The encoder checks the current buffer fullness before encoding next VOP: If the buffer occupancy exceeds 80% of the buffer size, the encoder skips next VOP to be coded. When VOP skipping happens, the buffer fullness is updated by the following method.

While $(B_{f,t} \geq \mathrm{skip\_margin} \cdot B_s)$
{ //Skip encoding next VOP $(\mathrm{VOP}_{j,\tau})$;
Deciding the VOP type d of the skipped $\mathrm{VOP}_{j,\tau}$;
$N_{j,\tau}^{d}$ - -; // Decrease the remaining number of VOP_j with type d by 1
Calculate $Bpp_{j,\tau}$ for $\mathrm{VOP}_{j,\tau}$;
$B_{f,\tau} := B_{f,t} - Bpp_{j,\tau}$
}//Where $\tau = t$ or $t+1$.

F. Parameters' Updating

After encoding VOP_i, the encoder updates the R-D model for VO_i based on the encoding results of the current VOP_i as well as the past VOP_is. $\alpha 1_{i,t}$ and $\alpha 2_{i,t}$ in formula (13) are re-calculated by using linear regression technique [13], [14].

$\beta_{i,t}^{d}$ is also updated after encoding VOP_i

$$\beta_{i,t}^{d} = \frac{\mathrm{Ave\_Bit}_{i,t}^{d}}{\mathrm{Ave\_}P_{i,t}} \tag{18}$$

where $\mathrm{Ave\_}P_{i,t}$ denotes, for VO_i, the average number of bits used in coding previous n_P_i P-VOPs, and $\mathrm{Ave\_Bit}_{i,t}^{d}$ denotes the average number of bits used in coding previous n_I_i I-VOPs, or n_B_i B-VOP, corresponding to the current VOP type d. Considering the tradeoff between keeping the algorithm stability and rapidly reflecting the influence of objects' variations, we set the window size $(n\_I_i + n\_P_i + n\_B_i)$ to the number of VOPs for VO_i in one second in the experiments. For example, if VO_i's encoding rate is 15 VOP/s, $(n\_I_i + n\_P_i + n\_B_i)$ is 15.

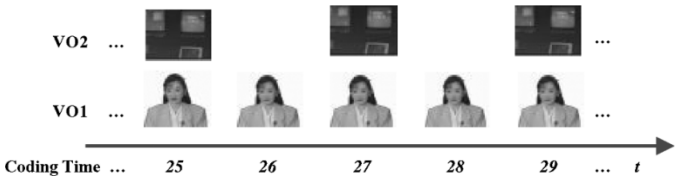

Fig. 3. Weight computation for priority adjustment.

G. Composition Problem Handling

Coding objects with different temporal resolutions may cause composition problem, in which undefined pixels or holes may appear in the reconstructed frame. This is because that the movement of one object, missing the updating of adjacent or overlapping objects, causes uncovered areas of the scene that cannot be associated with either object and for which no pixels are defined [22], [26]. Indeed, when encoding synchronous objects, there is no problem with object composition during frame reconstruction in the decoder. To overcome this problem, we simply adopt the method proposed by Lee and Vetro [22].

III. Priority Adjustment Among Multiple Objects

A simple and efficient way to express user or application requirements is to specify priorities among multiple objects. Priority can be directly used to control bit allocation in the proposed algorithm. To implement this, we can simply dynamically adjust the weight $W_{i,t}$ for VO_i by updating (4) and (5) to

$$\bar{Q}_{t-1} = \frac{\sum_{j=1}^{M} V_{j,t-1} \cdot (Q_{j,t-1} - U_j)}{\sum_{j=1}^{M} V_{j,t-1}} \tag{19}$$

$$W_{i,t} = W_{i,t-1} \times \left(\frac{\bar{Q}_{t-1}}{Q_{i,t-1} - U_i}\right)^{\Gamma} \tag{20}$$

where U_i is the "bias value" of PSNR for VO_i and represents the relative priority of VO_i.

In asynchronous environments, objects have different temporal rates and may not occur at each time slot. How to decide $Q_{i,t-1}$ so that we can compute $W_{i,t}$ by (20)? Here we always adopt the quality of the last encoding time $t-1$ for each VO. Fig. 3 shows an example to compute objects' weights, at time 29, the last encoding time for VO_1 and VO_2 is 28 and 27 respectively, so when calculating $W_{i,29}$, we use $Q_{1,28}$ directly and adopt $Q_{2,27}$ instead of $Q_{2,28}$. The size factor $V_{j,t-1}$ can be solved in the same manner.

In principle, if $Q_{i,t-1}$ is lower than $\bar{Q}_{t-1}$, we increase $W_{i,t}$, then VO_i obtains more target bits and thus achieves a higher quality; otherwise, $W_{i,t}$ is decreased and achieves a lower quality. By this way, we can adjust the coding quality of VO_i according to the requirement. When considering the priorities of objects, " $U_i > U_j$ " means VO_i has higher priority than

TABLE I
SYNCHRONOUS MULTIPLE OBJECT RATE CONTROL (IPPP . . . PPP)

Video Sequence	Algorithm	Bit Rate (kbps)				# Coded VOPs		Average PSNR (dB)	
		Target	Actual	VO1	VO2	VO1	VO2	VO1	VO2
News_1,	VM8	128	128.51	63.83	64.68	150	150	33.53	34.87
News_2	ARC	128	128.97	70.65	58.32	150	150	34.08	34.11
News_1,	VM8	256	257.83	142.94	114.89	150	150	38.69	39.09
News_2	ARC	256	257.46	147.73	109.73	150	150	38.87	38.91
Bream2_1,	VM8	128	128.24	101.51	26.73	150	150	27.08	42.34
Bream2_0	ARC	128	128.12	118.65	9.47	150	150	27.92	39.09
Bream2_1,	VM8	256	257.67	207.43	49.81	150	150	31.24	44.07
Bream2_0	ARC	256	256.24	245.18	11.06	150	150	32.35	40.27
Children2_1	VM8	384	384.29	296.08	88.21	147	147	32.44	40.54
Children2_2	ARC	384	384.06	332.53	51.53	150	150	33.54	37.61
Coastguard2	VM8	256	255.87	126.52	129.35	150	150	37.79	35.33
Coastguard3	ARC	256	256.28	104.20	152.08	150	150	36.03	36.13
Container_1,	VM8	112	111.69	73.66	38.03	150	150	31.81	47.32
Container_5	ARC	112	112.34	96.12	16.22	150	150	33.55	34.91

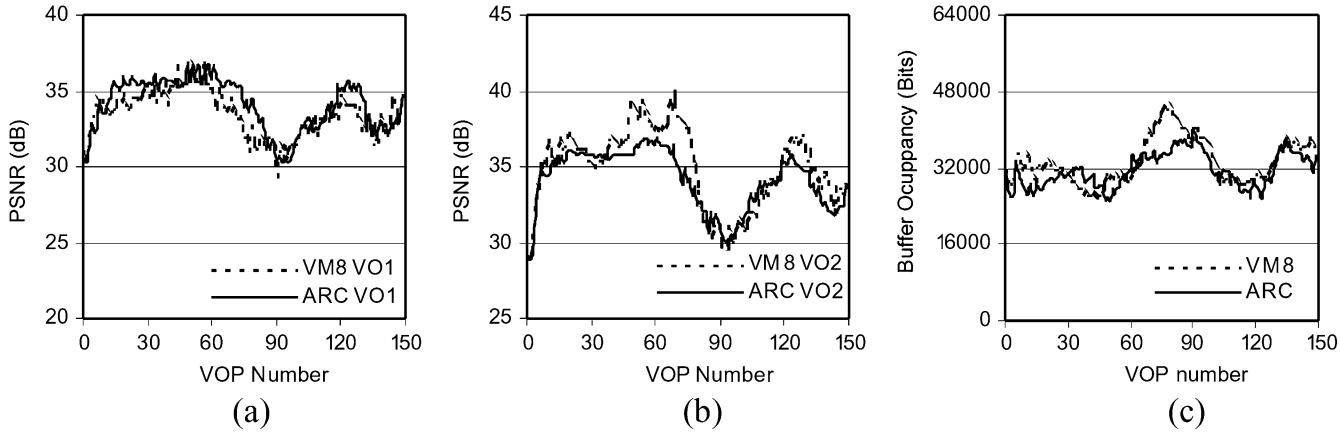

Fig. 4. *News* sequence in QCIF, 2 synchronous VOs, 30 VOP/s, 128 kb/s, IPPP . . . PPP. (a) PSNR curves of VO_1. (b) PSNR curves of VO_2. (c) Buffer occupancy.

$\mathrm{VO}_j, W_{i,t}$ should be increased and the coding quality of VO_i should be improved during encoding process; If $U_i < U_j, \mathrm{VO}_i$ has lower priority than VO_j, its weight and PSNR should be decreased while encoding; if $U_i = U_j, \mathrm{VO}_i$ and VO_j have the same priority, then the proposed algorithm tries to achieve balanced qualities among objects.

IV. SIMULATION RESULTS

To evaluate the performance of the proposed RC strategy, we have conducted three sets of experiments:

1) synchronous multiple objects encoding with the same VOP rate;
2) asynchronous multiple objects encoding with different VOP rates;
3) priority adjustment among multiple objects.

The results are compared with those achieved using the VM8 RC algorithm suggested by the MPEG-4 visual standard when possible. The initial values of $\beta_{i,0}^0$ for $I-\mathrm{VOP}_i, \beta_{i,0}^1$ for $P-\mathrm{VOP}_i$, and $\beta_{i,0}^2$ for $B-\mathrm{VOP}_i$ are 3.0, 1.0, and 0.5, respectively. $\beta_{i,t}^1$ is fixed to 1.0, $\beta_{i,t}^0$ and $\beta_{i,t}^2$ are dynamically adjusted during the encoding process. The default buffer size B_s is set to half of the target bitrate as VM8 [23]. According to MPEG-4 core experiments, the PSNR of a skipped VOP is defined by considering that a skipped VOP is represented in the decoded sequence by repeating the last coded VOP of the object [7], [17].

A. Synchronous Multiple Object Rate Control

All sequences are in QCIF format, the target number of VOPs to be encoded is 150. Multiple VOs are encoded at the same VOP rate (30 VOP/s) with different temporal prediction structures:

1) only first VOP is $I-\mathrm{VOP}$ and the remaining VOPs are all $P-\mathrm{VOPs}$ (IPPP . . . PPP);
2) both I-VOP and P-VOP are used, the intraperiod is set to 15 VOPs (IPPP . . . IPPP).

Structure (1) is the simplest case in RC since only $P-\mathrm{VOPs}$ need to be controlled, and this is the general assumption in [7], [13]–[16], [23]. Its performance results are reported in Table I. Fig. 4 shows PSNR and buffer curves for *News* sequence with two objects at the target bitrate 128 kb/s.

For structure (2), Table II presents its encoding results, Fig. 5 displays performance curves for News sequences.

By examining the results in Tables I and II, one can see that, when compared with the VM8 solution, ARC achieves more accurate target bit rates and target VOP rate (30 VOP/s) with usually higher average PSNRs or more balanced qualities among

TABLE II
SYNCHRONOUS MULTIPLE OBJECT RATE CONTROL (IPPP . . . IPPP)

Video Sequence	Algorithm	Bit Rate (kbps)				# Coded VOPs		Average PSNR (dB)	
		Target	Actual	VO1	VO2	VO1	VO2	VO1	VO2
News_1,	VM8	128	130.13	56.09	74.04	140	140	32.42	32.54
News_2	ARC	128	128.74	57.91	70.83	150	150	32.86	32.88
News_1,	VM8	256	259.36	124.07	135.29	143	143	37.24	37.48
News_2	ARC	256	257.48	125.63	131.85	150	150	37.60	37.66
Bream2_1,	VM8	128	130.38	100.03	30.35	143	143	26.34	40.82
Bream2_0	ARC	128	128.67	115.24	13.43	150	150	27.26	37.77
Bream2_1,	VM8	256	260.82	209.02	51.80	145	145	30.71	43.24
Bream2_0	ARC	256	258.14	242.82	15.32	150	150	31.86	38.26
Children2_1,	VM8	256	257.84	186.61	71.23	144	144	28.36	33.84
Children2_2	ARC	256	256.60	191.66	64.94	150	150	28.90	31.64
Children2_1,	VM8	384	384.40	291.49	92.91	146	146	31.63	34.95
Children2_2	ARC	384	385.47	295.06	90.41	150	150	32.07	34.58
Coastguard_2,	VM8	112	112.50	51.54	60.96	148	148	30.23	30.16
Coastguard_3	ARC	112	112.64	51.78	60.86	150	150	30.38	30.40
Container_1,	VM8	112	113.19	75.52	37.67	150	150	31.15	45.90
Container_5	ARC	112	112.21	96.52	15.69	150	150	32.80	34.07

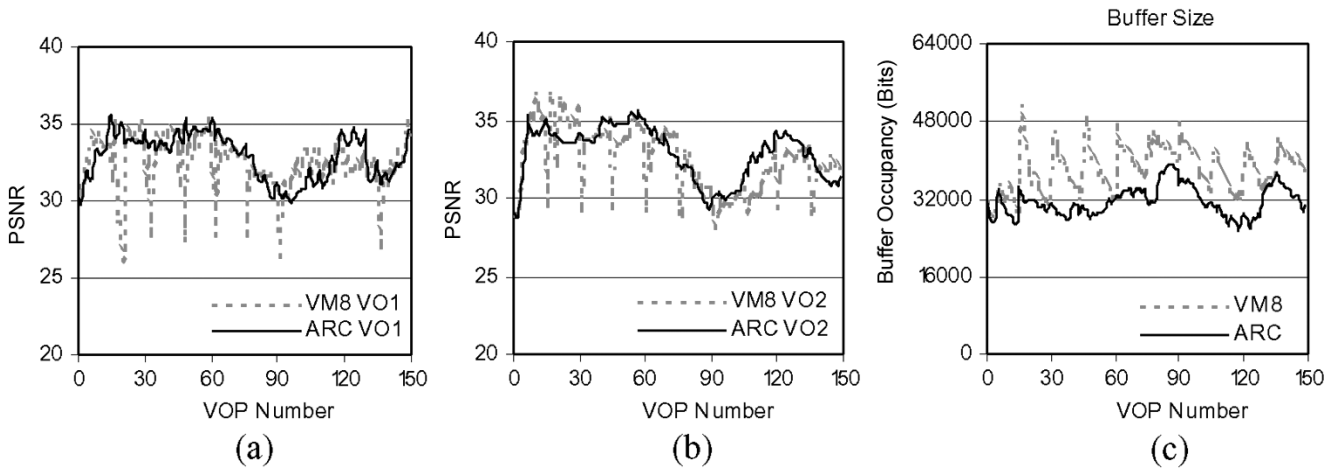

Fig. 5. *News* sequence in QCIF, 2 synchronous VOs, 30 VOP/s, 128 kb/s, IPPP . . . IPPP. (a) PSNR curves of VO_1. (b) PSNR curves of VO_2. (c) Buffer occupancy.

multiple objects. For example, when coding the *News* sequence at the target bitrate 128 kb/s (Table II), the PSNRs of VO_1 and VO_2 of ARC (32.86 and 32.88 dB) are about 0.44 and 0.34 dB higher than those of VM8 (32.42 and 32.54 dB), at the same time, VM8 skips 10 VOPs and actual bitrate is 130.13 kb/s, while ARC skips no VOP and actual bitrate is 128.74 kb/s, closer to the target 128 kb/s. Figs. 4(a)–(b) and 5(a)–(b) show that ARC obtains smoother qualities among VOPs. VM8 RC does not estimate target bits and calculate QPs for I-VOPs, its performance for sequences in IPPP . . . IPPP . . . format is not well, it has quality fluctuation between intercoded VOPs and intracoded VOPs. The proposed algorithm (ARC) also does not perform target bit allocation for I-VOPs, it just directly estimates QPs for I-VOPs using formula (14) and (15). Normally, there exists instant buffer surge when VOPs are intracoded [27]. However, the buffer occupancy curves of ARC in Figs. 4(c) and 5(c) are quite stable, as they are around half level of the buffer size with a small variation and almost avoid instant buffer surge when encoding I-VOPs [Fig. 5(c)], this indicates the proposed methods are successful.

One may note that in some cases in Tables I and II, the PSNR of one object in VM8 is much higher than that of the same object in ARC, while the other object's PSNR in VM8 is lower than the same object's in ARC. Thus, the quality difference between objects of ARC is smaller than that of VM8. For example, VO1 in the *Container* sequence is a moving big boat while VO5 is a very small moving American flag whose size is only one MB, one can see from Table I and Table II, when using VM8 RC, the quality difference between these two objects is very large; however, this quality difference of ARC has been reduced. ARC prevents the background object to have significantly better quality than that of the foreground object. This is meaningful since one pays much more attention to the foreground object than the background object.

B. Asynchronous Multiple Object Rate Control

By default, the encoding VOP rate of the object with higher activity is set to 15 VOP/s, and that of the second object with lower activity is 10 VOP/s. The total target VOPs to be encoded for VO_1 and VO_2 are 150 and 100, respectively. To compare

TABLE III
ASYNCHRONOUS MULTIPLE OBJECT RATE CONTROL (IPPP . . . PPP)

Video Sequence	Algorithm (ARC)	Bit Rate (kbps)				# Coded VOPs		Average PSNR (dB)	
		Target	Actual	VO1	VO2	VO1	VO2	VO1	VO2
News_1, News_2	Synchronous	128	128.65	74.08	54.57	150	150	38.86	38.83
	Asynchronous	128	128.24	87.12	41.12	150	100	39.98	39.65
Bream2_1, Bream2_0	Synchronous	128	127.77	121.47	6.30	150	150	32.35	41.23
	Asynchronous	128	128.28	124.42	3.86	150	100	32.51	40.94
Children2_1, Children2_2	Synchronous	256	255.69	217.95	37.74	150	150	35.96	39.21
	Asynchronous	256	255.81	232.49	23.32	150	100	36.61	38.45
Coastguard_2, Coastguard_3	Synchronous	64	63.91	29.89	34.02	150	150	31.98	32.13
	Asynchronous	64	63.86	38.28	25.58	150	100	33.78	33.52
Container_1, Container_5	Synchronous	64	63.91	54.19	9.72	150	150	34.31	36.81
	Asynchronous	64	63.98	58.60	5.38	150	100	34.89	35.08

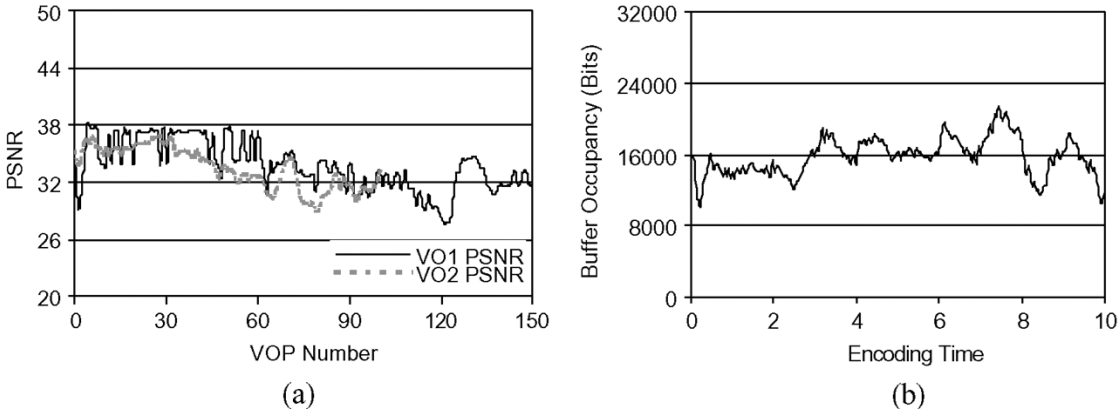

Fig. 6. PSNR and buffer curves of *Coastguard* sequence at 64 kb/s, QCIF, 2 asynchronous VOs, VO1: 15 VOP/s, VO2: 10 VOP/s, IPPP . . . PPP. (a) PSNR. (b) Buffer fullness (buffer size: 32 000).

the performance results, we also give the results of two VOs encoding at the same VOP rate (15 VOP/s) using ARC. All sequences are in QCIF format and encoded with different temporal prediction structures:

1) only first VOP is I-VOP and the remaining VOPs are all $P-$VOPs (IPPP . . . PPP);
2) both $I-$-VOP and P-VOP are used, the intraperiod is set to 15 VOPs (IPPP . . . IPPP).

Table III and Fig. 6 show encoding performance for structure (1), while Table IV, Fig. 7 and Fig. 8 demonstrate the encoding results for structure (2).

Besides accurate target bit rates have been realized without VOP skipping, the results in Table III and IV also show that ARC obtains a better tradeoff between spatial and temporal resolutions. For example, in Table IV, for the *News* sequence, VO_1 is Ballet which has faster movement, while VO_2 is Speakers with slower movement. Asynchronous scheme encodes VO_1 with a higher temporal rate than VO_2, when the target bitrate is 64 kb/s, VO_1 can obtains more bits (36.04 kb/s) than VO_1 (28.79 kb/s) in synchronous coding. Hence, VO_1 obtains a higher average PSNR (34.09 dB) when compared with its PSNR (32.87 dB) in synchronous condition. Meanwhile, although VO_2 gets fewer bits (27.97 kb/s) than its synchronous case (35.63 kb/s), since it has a slower temporal rate, the average visual quality of VO_2 still has about 1.00 dB improvement to its PSNR in synchronous case. These results indicate that a better tradeoff between spatial and temporal qualities can be achieved in asynchronous mode when specifying suitable VOP rates. This is especially useful when network bandwidths are very scarce. Fig. 6 shows PSNR and buffer curves of *Coastguard* sequence for structure (1). Fig. 7 and Fig. 8 show PSNR and buffer curves of *News* and *Bream* sequences for structure (2). In asynchronous coding environments, VO_1 and VO_2 have different encoding rates and may appear at different encoding time slot, even they occur together in one slot, the number of VOP_1 maybe different from that of VOP_2, for example, when the encoding time $t = 6$, the number of encoding VOP_1 is 90 and that of VOP_2 is 60. Based on the above reasons, we use "encoding time" as x axis instead of "VOP number" in Fig. 6(b) and Fig. 8. One can see that buffer curves in Fig. 6(b) and Fig. 8 are kept around the half level of the buffer size with small variations, this indicates our buffer policy is effective for asynchronous RC.

C. Priority Adjustment Among Multiple Objects

In addition to trying to achieve comparable and balanced qualities among multiple objects, ARC also provides priority adjustment among multiple objects. Table V and Table VI show the results of priority adjustment for synchronous and asynchronous objects for the temporal prediction structure IPPP . . . IPPP. The encoding rate of synchronous coding is 30 VOP/s, and the total VOPs to be encoded are 150 for both VO_1 and VO_2. Under asynchronous condition, the encoding

TABLE IV
ASYNCHRONOUS MULTIPLE OBJECT RATE CONTROL (IPPP . . . IPPP)

Video Sequence	Algorithm (ARC)	Bit Rate (kbps)				# Coded VOPs		Average PSNR (dB)	
		Target	Actual	VO1	VO2	VO1	VO2	VO1	VO2
News_1, News_2	Synchronous	64	64.42	28.79	35.63	150	150	32.87	32.96
	Asynchronous	64	64.01	36.04	27.97	150	100	34.09	33.96
News_1, News_2	Synchronous	128	128.97	61.89	67.08	150	150	37.54	37.82
	Asynchronous	128	127.93	77.10	50.83	150	100	39.11	38.85
Bream2_1, Bream2_0	Synchronous	64	64.51	57.81	6.70	150	150	27.28	37.77
	Asynchronous	64	64.32	60.07	4.25	150	100	27.48	38.00
Bream2_1, Bream2_0	Synchronous	128	128.62	120.52	8.10	150	150	31.82	38.38
	Asynchronous	128	128.95	124.13	4.82	150	100	32.06	38.63
Children2_1, Children2_2	Synchronous	128	128.53	98.65	29.87	150	150	29.04	32.68
	Asynchronous	128	128.52	105.98	22.54	150	100	29.60	29.83
Children2_1, Children2_2	Synchronous	256	257.16	197.51	59.65	150	150	34.60	38.04
	Asynchronous	256	257.13	223.39	33.74	150	100	35.89	36.00
Coastguard_2, Coastguard_3	Synchronous	64	64.00	28.66	35.34	150	150	31.19	31.31
	Asynchronous	64	64.05	37.01	27.04	150	100	33.06	32.86
Container_1, Container_5	Synchronous	64	64.45	54.81	9.64	150	150	33.63	36.39
	Asynchronous	64	64.03	58.54	5.49	150	100	34.09	34.66

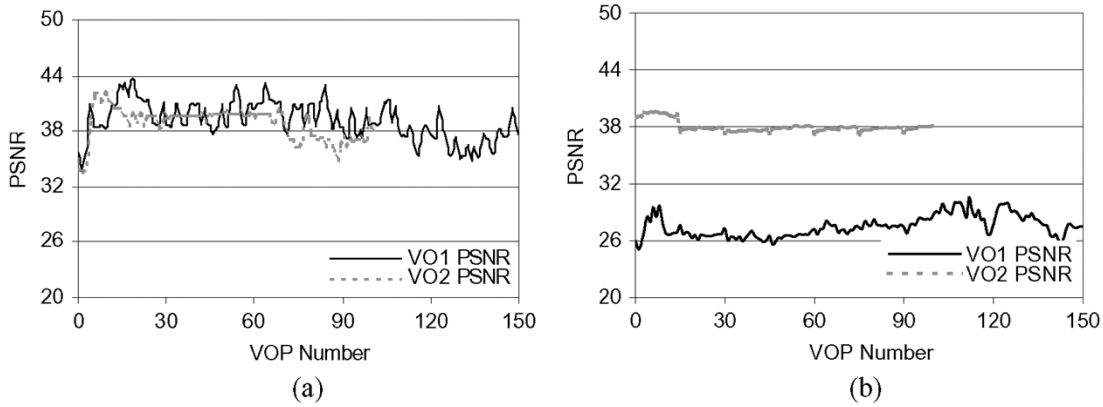

Fig. 7. PSNR Curves of *News* and *Bream* sequences in QCIF, 2 asynchronous VOs, VO1: 15 VOP/s, VO2: 10 VOP/s, IPPP . . . IPPP. (a) *News* (128 kb/s). (b) *Bream* (64 kb/s).

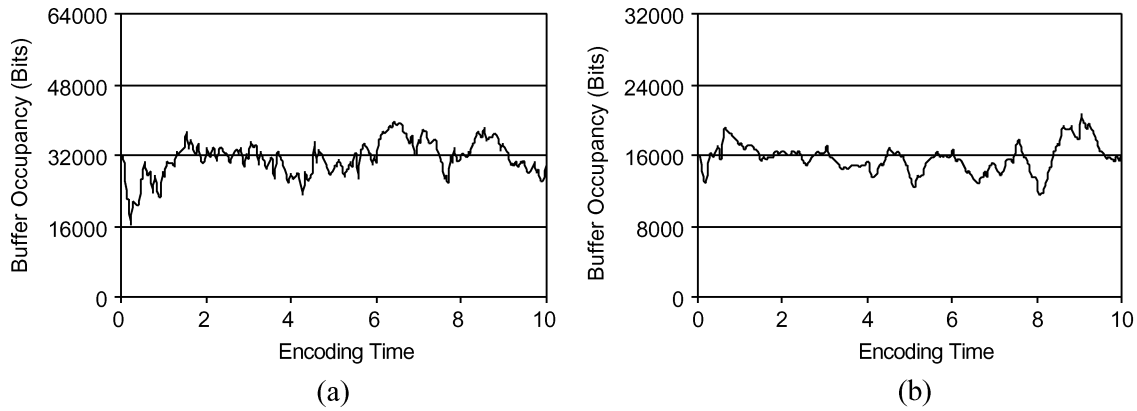

Fig. 8. Buffer fullness of *News* and *Bream* sequences in QCIF, 2 asynchronous VOs, VO1: 15 VOP/s, VO2: 10 VOP/s, IPPP . . . IPPP. (a) *News* (128 kb/s, buffer size 64 000). (b) *Bream* (64 kb/s, buffer size: 32 000).

VOP rates of VO_1 and VO_2 are 15 and 10 VOP/s, correspondingly, and the total VOPs to be encoded are 150 for VO_1 and 100 for VO_2.

By examining the results in Table V and Table VI, it is clear that ARC achieves target bitrates accurately without VOP skipping. Further, we can see that the priority adjustment is very effective when differences between coding complexities among objects are not too large. For example, for the *News* sequence, the priorities of VO_1 and VO_2, U_1 and U_2, are set to 0.0 and 0.0, respectively, at the target bitrate 128 kb/s in Table V, which means VO_1 and VO_2 are equally important, ARC has achieved comparable coding qualities between these two objects: VO_1's average PSNR is 32.86 and VO2's is 32.88. However, when U_1 and U_2 are set to 3.0 and 0.0 respectively, which means

TABLE V
PRIORITY ADJUSTMENT AMONG S{U}YNCHRONOUS OBJECTS (IPPP...IPPP)

Video Sequence	Alg.	Priority	Bit Rate (kbps)				# Coded VOPs		Average PSNR (dB)	
			Target	Actual	VO1	VO2	VO1	VO2	VO1	VO2
News_1, News_2	VM8		128	130.13	56.09	74.04	140	140	32.42	32.54
	ARC	U_1=0.0 U_2=0.0	128	128.74	57.91	70.83	150	150	32.86	32.88
		U_1=3.0 U_2=0.0	128	129.08	72.39	56.69	150	150	34.15	31.39
		U_1=4.0 U_2=0.0	128	128.93	75.74	53.19	150	150	34.46	30.88
		U_1=0.0 U_2=4.0	128	128.04	37.68	90.36	150	150	30.58	34.48
News_1, News_2	VM8		256	259.36	124.07	135.29	143	143	37.23	37.48
	ARC	U_1=0.0 U_2=0.0	256	257.48	125.63	131.85	150	150	37.60	37.66
		U_1=4.0 U_2=0.0	256	258.01	157.17	100.84	150	150	39.31	35.58
		U_1=-3.0 U_2=-1.0	256	256.46	107.22	149.24	150	150	36.62	38.51
Children2_1, Children2_2	VM8		256	257.84	186.61	71.23	144	144	28.36	33.84
	ARC	U_1=0.0 U_2=0.0	256	256.60	191.66	64.94	150	150	28.90	31.64
		U_1=-3.0 U_2=0.0	256	256.23	182.88	73.35	150	150	28.52	33.55
Children2_1, Children2_2	VM8		384	384.40	291.49	92.91	146	146	31.63	34.95
	ARC	U_1=0.0 U_2=0.0	384	385.47	295.06	90.41	150	150	32.07	34.58
		U_1=3.0 U_2=0.0	384	385.88	317.30	68.58	150	150	32.71	33.01
		U_1=0.0 U_2=3.0	384	384.27	270.88	113.39	150	150	31.31	36.11
Bream2_1, Bream2_0	VM8		256	260.82	209.02	51.80	145	145	30.71	43.24
	ARC	U_1=0.0 U_2=0.0	256	258.14	242.82	15.32	150	150	31.86	38.26
		U_1=6.0 U_2=0.0	256	258.05	243.48	14.57	150	150	31.87	38.10
		U_1=-6.0 U_2=0.0	256	258.54	228.93	29.61	150	150	31.48	40.73

VO_1 has higher priority, VO_1 can obtain higher bitrate (72.39 kb/s) during target bit allocation, therefore the average PSNR of VO_1 (34.15 dB) is almost 3 dB higher than VO_2 (31.39 dB). On the other hand, if U_1 and U_2 are 0.0 and 4.0, the average PSNR of VO_1 (30.58 dB) is about 4 dB lower than that of VO_2 (34.48 dB). Fig. 9 displays the visual effects of the 142th frame of *News* sequence for priority adjustment. One may see VO_1 (Ballet) in Fig. 9(a) is clearer than it in Fig. 9(b), while VO_2 (Speakers) in Fig. 9(a) is more obscure than Fig. 9(b). Similar results can be found in asynchronous environment, which are shown in Table VI.

When the coding complexity difference among objects is too large, the priority adjustment will become less effective. For example, there exists a large coding complexity difference between the background object Bream2_0(VO_2), and the foreground object Bream2_1(VO_1). When the target bitrate is 256 kb/s in Table V, the average coding quality of Bream2_0 is as high as to 43.24 dB, and Bream2_1's PSNR is low to 30.71 dB using VM8 RC, the quality difference between these two objects is very large (12.53 dB), and the actual bitrates of Bream2_0 and Bream2_1 are 51.80 kb/s and 209.02 kb/s, respectively. ARC has decreased the PSNR difference to 6.40 dB, meaning that it works well since it has reduced the quality difference effectively. Under this case, if we still want to increase the quality of VO_1 and decrease that of VO_2 by setting U_1 and U_2 to 6.0 and 0.0, the PSNR of VO_1 only can be increased very few, from 31.86 to 31.87 dB. In the same time, the PSNR for VO_2 has been decreased a little. There are two reasons for this: First, ARC has done its best to increase the value of PSNR for VO_1, the actual bitrate 243.48 kb/s is almost the maximum bitrate used to code VO_1, and the actual bitrate of VO_2 has been reduced successfully from 51.80 (VM8) to 14.57 kb/s, in this case, VO_2 only spends 5.65% of the total actual bitrate (258.05 kb/s). In other words, ARC cannot distribute any more bits to VO_1 to further increase its PSNR. Second, since VO_1 has a high coding complexity, it needs very large amount of bits to increase its coding quality a very small amount. However, we can increase VO_2's PSNR

TABLE VI
PRIORITY ADJUSTMENT AMONG ASYNCHRONOUS OBJECTS (IPPP . . . IPPP)

Video Sequence	Priority	Bit Rate (kbps)				# Coded VOPs		PSNR (dB)	
		Target	Actual	VO1	VO2	VO1	VO2	VO1	VO2
News_1, News_2	U_1=0.0 U_2=0.0	64	64.01	36.04	27.97	150	100	34.09	33.96
	U_1=1.0 U_2=-1.0	64	64.01	40.21	23.80	150	100	34.76	32.67
	U_1=-3.0 U_2=0.0	64	64.04	28.56	35.48	150	100	32.76	35.70
News_1, News_2	U_1=0.0 U_2=0.0	128	127.93	77.10	50.83	150	100	39.11	38.85
	U_1=2.0 U_2=0.0	128	128.64	84.42	44.22	150	100	39.83	37.80
	U_1=0.0 U_2=1.0	128	128.35	73.13	55.22	150	100	38.69	39.49
Children2_1, Children2_2	U_1=0.0 U_2=0.0	256	257.13	223.39	33.74	150	100	35.89	36.00
	U_1=3.0 U_2=0.0	256	256.04	233.21	22.83	150	100	36.42	34.31
	U_1=-2.0 U_2=0.0	256	256.13	215.43	40.70	150	100	35.54	38.62
Children2_1, Children2_2	U_1=0.0 U_2=0.0	384	384.43	331.07	53.36	150	100	39.64	40.92
	U_1=0.0 U_2=-4.0	384	382.66	348.32	34.34	150	100	40.00	36.65
	U_1=0.0 U_2=3.0	384	379.58	312.77	66.81	150	100	39.06	42.16
Bream2_1, Bream2_0	U_1=0.0 U_2=0.0	128	128.95	124.13	4.82	150	100	32.06	38.63
	U_1=-8.0 U_2=0.0	128	128.39	121.14	7.25	150	100	31.88	40.05
	U_1=8.0 U_2=0.0	128	128.85	124.47	4.38	150	100	32.07	38.38

(a) (b)

Fig. 9. Visual effects of the 142nd frame for *News* sequence, two synchronous VOs, 30 VOP/s, 128 kb/s. (a) $U_1 = 4.0; U_{2=0.0}$. (b) $U_1 = 0.0; U_2 = 4.0$.

easily, because VO_2's coding complexity is relatively lower, and thus, it can easily obtain higher PSNR improvement when taking a little more bits. Table VI also shows the similar results under asynchronous environment.

V. CONCLUSION

In this paper, we proposed a multiple-object RC algorithm for MPEG-4 video coding. Unlike traditional bit allocation methods, which first perform bit allocation among different coding time and then distribute bits among multiple objects at one coding time, the proposed algorithm regards each object as one relatively independent "object stream" along the coding time. The algorithm directly distributes target bits among multiple "object streams," which is also consistent with the original object-based concept of MPEG-4. We adopt an objective measurement, the "coding qualities," to guide the bit allocation among "object streams," which is more effective and direct than other heuristic allocation rules. Combined with a PID buffer controller and other feedback strategies, the proposed algorithm performs well not only for asynchronous multiple objects, but also for synchronous objects. Although this paper only uses the CBR application in the experiments, it is very straightforward to apply it to VBR applications as well. Regarding future works, more research is required to achieve an optimal tradeoff between visual quality and temporal resolution, such as dynamically determining VOP rates. In addition, the relation between the complexity of object and priority adjustment to further improve the performance of the algorithm needs to be investigated.

REFERENCES

[1] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. Image Process.*, vol. 2, no. 4, pp. 160–175, Apr. 1993.

[2] W. Ding and B. Liu, "Rate control of MPEG video coding and recording by rate-quantization modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 1, pp. 12–20, Feb. 1996.

[3] L.-J. Lin and A. Ortega, "Bit-rate control using piecewise approximated rate-distortion characteristics," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 446–459, Aug. 1998.

[4] B. Tao, B. W. Dickinson, and H. A. Peterson, "Adaptive model-driven bit allocation for MPEG video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 1, pp. 147–157, Feb. 2000.

[5] *MPEG Video Test Model 5*, Apr. 1993. ISO/IEC JTCI/SC29/WG11, MPEG93/457, Draft.

[6] K. Oehler and J. L. Webb, "Macroblock quantizer selection for H.263 video coding," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, Oct. 1997, pp. 365–368.

[7] J. R.-Corbera and S. Lei, "Rate control in DCT video coding for low-delay communications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 2, pp. 172–185, Feb. 1999.
[8] E. C. Reed and F. Dufux, "Constrained Bit-Rate Control for Very Low Bit-Rate Streaming-Video Applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 7, pp. 882–889, Jul. 2001.
[9] *Overview of the MPEG-4 Standard*, R. Koenen, Ed., Mar. 1999. Doc. ISO/IEC JTC1/SC29/WG11 N2725.
[10] T. Sikora, "The MPEG-4 video standard verification model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, p. 19–31, Feb. 1997.
[11] P. Nunes and F. Pereira, "Object-based rate control for the MPEG-4 visual simple profile," in *Proc. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'99)*, Berlin, Germany, May 1999, pp. 161–164.
[12] P. Nunes and F. Pereira, "Rate control for scenes with multiple arbitrarily shaped video objects," in *Proc. Picture Coding Symposium (PCS'97)*, Berlin, Germany, Sep. 1997, pp. 303–308.
[13] H.-J. Lee, T. Chiang, and Y.-Q. Zhang, "Scalable rate control for very low bitrate video," in *Proc. 1997 Int. Conf. Image Processsing*, vol. 2, Oct. 1997, pp. 768–771.
[14] H.-J. Lee, T. Chiang, and Y.-Q. Zhang, "Scalable rate control for MPEG-4 video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 6, pp. 878–894, Sep. 2000.
[15] A. Vetro, H. Sun, and Y. Wang, "MPEG-4 rate control for multiple video objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 1, pp. 186–199, Feb. 1999.
[16] *Coding of Moving Pictures and Associated Audio MPEG 97/M1631*, Feb. 1997. ISO/IEC JTC1/SC29/WG11.
[17] J. I. Ronda, M. Eckert, F. Jaureguizar, and N. Garcia, "Rate control and bit allocation for MPEG-4," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 8, pp. 1243–1258, Dec. 1999.
[18] Y. Sun and I. Ahmad, "A robust and adaptive rate control algorithm for object-based video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 10, pp. 1167–1182, Oct. 2004.
[19] J. I. Ronda, M. Eckert, S. Rieke, F. Jaureguizar, and A. Pacheco, "Advanced rate control for MPEG-4 coders," *Visual Commun. & Image Processing, Proc. SPIE*, pp. 383–394, Jan. 1998.
[20] P. Nunes and F. Pereira, "Scene level rate control algorithm for MPEG-4 video coding," *Proc. SPIE 4310Visual Communications and Image Processing*, pp. 194–205, 2001.
[21] C.-W. Hung and D. W. Lin, "Toward jointly optimal rate allocation for multiple videos with possibly different frame rates," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS 2000)*, Geneva, Switzerland, May 2000, pp. 13–16.
[22] J.-W. Lee, A. Vetro, Y. Wang, and Y4-S. Ho, "Bit allocation for MPEG-4 video coding with spatio-temporal tradeoffs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 6, pp. 488–502, Jun. 2003.
[23] *Coding of Moving Pictures and Associated Audio MPEG97/N1796*, Jul. 1997. MPEG-4 video verification model V8.0, ISO/IEC JTC1/SC29/WG11.
[24] A. F. D'Souza, *Design of Control System*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[25] C. L. Phillips and R. D. Harbor, *Basic Feedback Control Systems*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1991.
[26] A. Vetro, H. Sun, and Y. Wang, "Object-based transcoding for adaptable video content delivery," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 387–401, Mar. 2003.
[27] F. Pan, Z. Li, K. Lim, and G. Feng, "A study of MPEG-4 rate control scheme and its improvements," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 5, pp. 440–446, May 2003.

Yu Sun (S'04–M'05) received the B.S. and M.S. degrees in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 1996, and the Ph.D. degree in computer science and engineering from the University of Texas at Arlington, Arlington, TX, in 2004.

From 1996 to 1998, she was a Lecturer in the Department of Computer Science, Sichuan Normal University, Sichuan, China. Since August 2004, she has been an Assistant Professor in the Computer Science Department, University of Central Arkansas, Conway, AR. Her main research interests include video compression, multimedia communication, and image processing, etc.

Ishfaq Ahmad (S'88–M'92–SM'03) received the B.Sc. degree in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 1985, the M.S. degree in computer engineering, and the Ph.D. degree in computer science, both from Syracuse University, Syracuse, NY, in 1987 and 1992, respectively.

He is currently a Full Professor of Computer Science and engineering in the Computer Science and Engineering Department, University of Texas (UT) at Arlington. Prior to joining UT Arlington, he was an associate professor in the Computer Science Department at Hong Kong University of Science and Technology (HKUST), Hong Kong. At HKUST, he was also the Director of the Multimedia Technology Research Center, an officially recognized researh center that he conceived and built from scratch. The center was funded by various agencies of the Government of the Hong Kong Special Administrative Region as well as local and international industries. With more than 40 personnel including faculty members, postdoctoral fellows, full-time staff, and graduate students, the center engaged in numerous research and development projects with academia and industry from Hong Kong, China, and the U.S. Particular areas of focus in the center are video (and related audio) compression technologies, videotelephone and conferencing systeme. The center commercialized several of its technologies to its industrial partners world wide. His recent research focus has been on developing parallel programming tools, scheduling and mapping algorithms for scalable architectures, heterogeneous computing systems, distributed multimedia systems, video compression techniques, and web management. His research work in these areas is published in over 125 technical papers in refereed journals and conferences.

Dr. Ahmad has recevied Best Paper Awards at Supercomputing'90 (New York), Supercomputing'91 (Albuquerque), and the 2001 International Conferene Parallel Processing (Spain). He has participated in the organization of several international conferences and is an Associate Editor of *Cluster Computing, Journal of Parallel and Distributed Computing,* IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, *IEEE Concurrency*, and *IEEE Distributed Systems Online*.