11

Interactive Video Communications

11.1 Video Conferencing and Telephony

Video conferencing and telephony are typical examples of interactive video communications, as nowadays mobile phones and portable devices are very popular. The major difference between video conference and telephony with other multimedia applications, such as multimedia messaging (MMS) and multimedia streaming is that the video conferencing is delay sensitive while others are not.

In the past decade, point-to-point protocols (PPP) have been in common use for TCP/IP communications (the protocol used by the Internet) over a telephone line. Using the Internet, multimedia communication can be achieved without incurring any long distance charges. On the other hand, the usage of a cable modem along with DSL technology has enabled broadband Internet access, where the cable modem is used to deliver broadband Internet access taking advantage of the unused bandwidth of the cable television network. The bandwidth of the cable connection varies from 3 Mbits/s to 30 Mbits/s and the upstream speed ranges from 384 Kbits/s to 6 Mbits/s. The DSL modem, on the other hand, takes advantage of the unused frequencies in the telephone line and varies in speed from hundreds of kbits/s to few Mbits/s. With the development of the 3G and 4G, the constraint on bandwidth for a wireless system to carry video content has been lifted, in addition, more sophisticated access protocols such as HSPA, EDGE, etc. hit the market and give a strong impetus to the mobile telephony business.

11.1.1 IP and Broadband Video Telephony

Compared to the very low-bit rate (around 40 kbit/s) of video telephony over the PSTN (Public Switched Telephone Network), the cable modem and Direct Subscriber Line (DSL) connections offer Internet connections at much higher bit rates. Thus, IP video telephony in conjunction with DSL and cable modems can now offer video communications at a much higher quality than before. IP Video Telephony uses the H.323 standard which can be used over any packet data network, such as those using the Internet protocol (IP). Owing to the interest in video telephony over IP, many of the existing commercial implementations use the H.323 standard.

4G Wireless Video Communications Haohong Wang, Lisimachos P. Kondi, Ajay Luthra and Song Ci
 ISBN: 978-0-470-77307-9

The Digital Subscriber Line (DSL) connection and cable Internet connections are both referred to as broadband since they use different channels to send the digital information simultaneously with the audio signal or the cable television signal. A DSL connection uses a dedicated line from the subscriber to the telephone company, while the cable Internet service is provided to a neighborhood by a single coaxial cable line. Hence, connection speed varies depending on how many people are using the service. Cable modems send the data signal over the cable television infrastructure. They are used to deliver broadband Internet access by taking advantage of the unused cable network bandwidth. DSL, on the other hand, uses a conventional twisted wire pair for data transmission. ADSL [1] uses two frequency bands known as upstream and downstream bands. The upstream band is used for communications from the end user to the telephone central office while the downstream band is used for communicating from the central office to the end user. ADSL provides dedicated local bandwidth in contrast to the cable modem which gives shared bandwidth. Hence, the upstream and downlink speed varies depending on the distance of the end user from the telephone office. Conventional ADSL has a downstream speed of approximately 8 Mbits/s and an upstream speed around 1 Mbits/s. Thus, acceptable quality video telephony is achievable with the advances in modem technology and audio and video compression.

11.1.2 Wireless Video Telephony

Video Telephony is offered in 3G networks in both the circuit-switch mobile core network and packet network. The former provides 64 kbits/s per circuit switched path, while the latter may provide greater bandwidth but the bandwidth is not guaranteed during the call as no dedicated circuit is reserved. 3GPP uses the 3G-324M protocol to support video telephony services. In January 2004, NTT DoCoMo (a Japanese operator) announced that its FOMA (freedom of mobile multimedia access) 3G video telephony service had passed the milestone of 2 million customers. Recently almost all mobile phones supporting UMTS networks can make videophone conversations with other UMTS users, and it was estimated that there were more than 130 million UMTS users in mid-2007. In the next section we provide more detail of the 3G-324M protocol that is widely used in videophone applications.

11.1.3 3G-324M Protocol

3G-324M [2] is the 3GPP umbrella protocol for video telephony in 3G mobile networks, which is based on the H.324 (a standard for low bit rate GSTN networks) specification for multimedia conferencing over circuit switched networks. 3G-324M is comprised of the following sub-protocols:

- H.245 for call control.
- H.223 for bitstreams to data packets multiplexer/demultiplexer.
- H.223 Annex A and B for error handling of low and medium BER detection, correction and concealment.
- Adaptation layers.

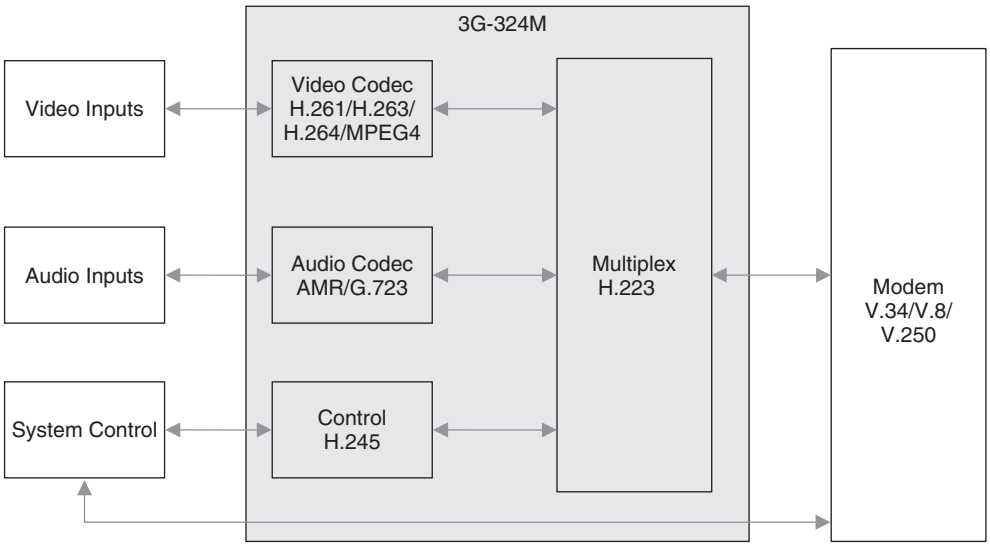


Figure 11.1 3G-324M basic structure

The basic structure of 3G-324M is shown in Figure 11.1, which consists of a multiplexer which mixes the various media types into a single bitstream (H.223), an audio compression algorithm (either a AMR or G.723 codec), a video compression algorithm (either a H.261, H.263, H.264 or MPEG4 codec) and a control protocol which performs automatic capability negotiation and logical channel control (H.245). The goal of this standard is to combine low multiplexer delay with high efficiency and the ability to handle bursty data traffic from a variable number of sources.

11.1.3.1 Multiplexing and Error Handling

3G-324M uses a multiplex standard, H.223, to mix the various streams of audio, video, data and the control channel together into a single bitstream for transmission over the modem. H.223 has a flexible mapping scheme suitable for a variety of media and for a variable frame length. In its mobile extension, it obtains greater synchronization and control of channel errors without losing its flexibility. H.223 consists of a lower multiplex layer and a set of adaptation layers. The lower multiplex layer mixes the different media streams, whereas the adaptation layers perform logical frame, sequence numbering, error detection and error correction by retransmission. Each adaptation layer is suitable for a different type of information channel. In H.223, there are 3 operation modes which are chosen according to the degree of error resiliency required in a 3G-324M system. In the first level, the multiplexing and QoS control are supported; in the second level, a 16-bit pseudorandom noise sequence is employed to improve the synchronization; in the third level, the payload length and FEC information are added in the header in order to improve error resilience capability.

11.1.3.2 Adaptation Layers

There are three adaptation layers in 3G-324M: AL1, AL2, and AL3. AL1 is intended primarily for data and control information transferring, in which no error detection and correction mechanism is provided. AL2 is intended primarily for digital audio transferring, which includes an 8 bit cyclic redundancy code (CRC). CRC is used to identify transmission errors. AL3 is intended primarily for digital video and includes provision for retransmission and a 16 bit CRC.

11.1.3.3 The Control Channel

The H.245 protocol controls the following items:

- Logical channel that opens or closes for media transmissions.
- Determines the master terminal at the beginning of a session.
- Exchanges the capabilities between both terminals, such as the mode of multiplexing, codec support, data sharing mode, etc.
- Operation mode that is sent from the receiver side to the transmitter side to convey the preference within its capability of the codec and the associated parameters.
- Call control commands and indications that check the status of the terminals and communications.

In addition, H.245 supports the numbered simple retransmission protocol (NSRP) and control channel segmentation and reassembly layer (CCSRL) sub-layer support in order to ensure reliable operation, therefore all terminals support both NSRP and SRP modes.

11.1.3.4 Audio and Video Channels

The 3G-324M specifications define the AMR codec as mandatory and G.723.1 as a recommended audio codec, it also declares the H.263 codec as mandatory and MPEG-4 a as recommended codec for video processing. The details of these video codecs have been discussed in Chapter 5.

11.1.3.5 Call Setup

There are seven phases in the call set up procedure, designated by letters A through G. In Phase A, an ordinary telephone connection is established. In Phase B, a regular analog telephone conversation can take place before the actual multimedia communication. When either user decides to start the multimedia communication, Phase C takes place. The two modems communicate with each other and digital communication is established. Then, in Phase D, the terminals communicate with each other using the H.245 control channel. Detailed terminal capabilities are exchanged and logical channels are opened. In Phase E, actual multimedia communication takes place. Phase F is entered when either user wishes to end the call. The logical channels are closed and an H.245 message is sent to the far-end terminal to specify the new mode (disconnect, back to voice mode, or another digital mode). Finally, in Phase G, the terminals actually enter the mode specified in the previous phase.

11.2 Region-of-Interest Video Communications

As mentioned in Chapter 6, ROI based video coding and communication has been very popular for wireless video telephony. As shown in Figure 11.2, the ROI based video communications system architecture provides users with greater flexibility and interactivity in specifying their desires and enables encoders to have greater efficiency in controlling the visual quality of coded video sequences. In this section, we demonstrate a few of the latest advances in ROI based bit allocation [3] and adaptive background skipping [4, 5] techniques.

11.2.1 ROI based Bit Allocation

In the literature, many ROI bit allocation algorithms [6–10] are based on a weighted version of the H.263+ TMN8 model [11], where a cost function is created and the distortion components in various regions in the function are punished differently by using a set of preset weights. As with most of the other video standards, as mentioned in Chapter 6, TMN8 uses a Q-domain rate control scheme, which models the rate and distortion with functions of quantization step size (QP). However, recent advances in rate control research and development have demonstrated that the ρ-domain rate control model [12] (ρ represents the number of non-zero AC coefficients in a macroblock in video coding) is more accurate and thus effectively reduces rate fluctuations. It is also observed that the ρ-domain rate control approach has already been used in industry trials [13–16]. To the best of our knowledge, so far there is no general optimized ρ-domain bit allocation model for ROI video coding, although [17] used the ρ-domain rate control model in their efforts to get an ad-hoc bit allocation solution. In this section, we introduce a ρ-domain optimized weighted bit allocation scheme for ROI video coding.

11.2.1.1 Quality Metric for ROI Video

Video quality measurement is still an open issue for ROI video coding. Most of the literature uses PSNRs on ROI and Non-ROI, respectively, as a measurement for evaluating regional visual quality, however, a quality measure for the whole image has not been addressed. In [18], a weighted mean squared error (MSE) metric was proposed in order to measure perceptual video quality. In this metric, the macroblocks are classified as activity macroblocks and static macroblocks, and different weights are

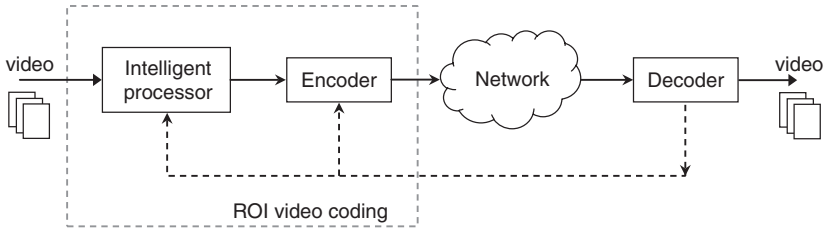


Figure 11.2 An example of ROI video coding and communications system

assigned to these macroblocks for calculating the weighted MSE for overall image. Although this measurement can be extended to use for ROI video, here we introduce a new quality measurement for ROI video coding which takes into account further aspects such as spatial/temporal visual quality.

In general, the evaluation of ROI video quality should consider at least three aspects: users' interest, video fidelity and perceptual quality of the reconstructed video data. The users' interest determines directly the classification of a video frame into ROI and Non-ROI parts and their associated perceptual importance factors. In video telephony applications, the speaker's face region is a typical ROI because a human being's facial expressions are very complicated and small variations can convey a large quantity of information. For the video fidelity factor, PSNR is a good measurement, which indicates the total amount of distortion of the reconstructed video frame compared to the original frame. In most cases, fidelity is the most important consideration for video coding, where any improvement might cause better subjective visual quality. However, it is not always the case, and that is why perceptual quality factors should also be taken into account. Perceptual quality considers both spatial errors, for example blocking and ringing artifacts, and temporal errors such as temporal flicker where the frame visual qualities change non-uniformly along the temporal axis.

Let us denote by D_R and D_{NR} the normalized per pixel distortion of the ROI and Non-ROI, and α the ROI perceptual important factor. If we assume that the relationship among the aspects mentioned above can be simplified into a linear function in video quality evaluation, then we can represent the overall distortion of the video sequence as:

$$\begin{aligned} D_{sequence} &= \alpha D_R + (1-\alpha) D_{NR} \\ &= \frac{\alpha}{M}\left[\beta \sum_{i=1}^{M} D_{RF}(f_i, \tilde{f}_i) + \gamma \sum_{i=1}^{M} D_{RS}(\tilde{f}_i) + (1-\beta-\gamma) D_{RT}(\tilde{f}_1, \ldots, \tilde{f}_M)\right] \\ &\quad + \frac{(1-\alpha)}{M}\left[\beta \sum_{i=1}^{M} D_{NF}(f_i, \tilde{f}_i)\right. \\ &\quad \left. + \gamma \sum_{i=1}^{M} D_{NS}(\tilde{f}_i) + (1-\beta-\gamma) D_{NT}(\tilde{f}_1, \ldots, \tilde{f}_M)\right] \end{aligned} \tag{11.1}$$

where f_i and $\tilde{f}_i$ are the ith original and reconstructed frames within the M frames in the video sequence, β and γ are weighting factors, D_R and D_{NR} are the total distortion for ROI and Non-ROI, D_{RF}, D_{RS} and D_{RT} are the normalized errors of ROI in fidelity, spatial perceptual quality and temporal perceptual quality, and D_{NF}, D_{NS} and D_{NT} are their counterparts for Non-ROI. It is clear that α, β and γ should be assigned real values between 0 and 1.

In low-bitrate video applications, such as wireless video telephony, blocking artifacts are the major concern of spatial perceptual quality. This kind of artifact is caused by the quantization where most of the high-frequency coefficients are removed (set to zero). The resulted effect is that the smoothed image blocks make the block boundaries quite pronounced. At the extreme low bit-rate cases, only DC coefficients will be coded which

makes the decoded image piece-wise constant blocks. In this work, we define the D_{RS} (similar for D_{NS}) as the normalized blockiness distortion, that is:

$$D_{RS}(\tilde{f}) = \frac{\text{boundaries with discontinuities}}{\text{Number of boundaries}} \tag{11.2}$$

where every boundary between blocks is checked to see if perceivable discontinuities exist. The discontinuity detection approach used in [19] is adopted, which checks the sum of the mean squared difference of the intensity slope across the block boundaries. The assumption of this approach is that the slopes on both sides of a block boundary are supposed to be identical and an abrupt change in slope is probably due to quantization.

In equation (11.1), the D_{RT} (or D_{NT}) is defined as an assigned score in the range of [0, 1] based on the variance of D_{RS} (or D_{NS}) for all the frames in the sequence. In this way, the terms on fidelity, spatial perceptual quality and temporal perceptual quality are normalized and can be bridged by weighting parameters α, β and γ to form a controllable video quality measurement. The selection of these weighting parameters is up to users based on their requirements and expectations. Again, this measurement is not a perfect metric, but it will be shown in the subsequent text that it helps the bit allocation process to favor subjective perception.

11.2.1.2 Bit Allocation Scheme for ROI Video

In video coding applications, a typical problem is to minimize $D_{sequence}$ with a given bit budget for the video sequence. The optimal solution for this complicated problem relies on an optimal frame-level rate control algorithm and an optimal macroblock-level bit allocation scheme. However, for real-time applications, such as wireless video telephony, where very limited information about future frames is available when coding the current frame, it is not practical or feasible to pursue an optimal frame-level rate control. Typically a popular greedy algorithm is resorted to which assumes that the complexity of the video content is distributed uniformly along the frames in the video sequence, and thus allocates a fraction of the available bits to each of the rest frames. For the same reason, taking care of $D_{NT}(\tilde{f}_1, \ldots, \tilde{f}_M)$ in the rate control is very difficult for these applications. Therefore, to find a practical solution and to simplify the problem we assume that good frame-level rate control is available and thus we narrow down the problem into a macroblock-level bit allocation problem. At the meantime, we propose a background skipping approach, which increases the chance of reducing the value of the term $D_{NT}(\tilde{f}_1, \ldots, \tilde{f}_M)$ because the skipped region will present the same perceptual quality as that of the previous frame and thus might reduce the fluctuation of the perceptual quality between consecutive frames. For measuring the image quality of a video frame, we use equation (11.1) by setting $\beta + \gamma = 1$.

Let us denote by R_{budget} the total bit budget for a given frame f and R the bit rate for coding the frame, then the problem can be represented by:

$$\begin{aligned} &\text{Minimize } \alpha \left[\beta D_{RF}(f, \tilde{f}) + (1-\beta) D_{RS}(\tilde{f})\right] \\ &\quad + (1-\alpha)\left[\beta D_{NF}(f, \tilde{f}) + (1-\beta) D_{NS}(\tilde{f})\right] \\ &\text{Such that } R \le R_{budget} \end{aligned} \tag{11.3}$$

Clearly, this optimization problem can be solved by Lagrangian relaxation and dynamic programming in the same fashion as in [20]. However, the computational complexity is a great deal higher than a real-time system can bear. Therefore, a low-complexity near-optimal solution is preferred. We propose a two-stage bit allocation algorithm in ρ-domain to solve this problem. In the first stage, we are solving an optimization problem:

$$\text{Minimize } \alpha D_{RF}(f, \tilde{f}) + (1-\alpha) D_{NF}(f, \tilde{f}), \text{ such that } R \leq R_{budget} \tag{11.4}$$

After the optimal coding parameters for (11.4) is obtained, in the second stage we adjust the coding parameters iteratively to reduce the term $\alpha D_{RS}(\tilde{f}) + (1-\alpha) D_{NS}(\tilde{f})$ until a local minimum is reached. Clearly, the result will be very close to the optimal solution when β is a relative large number. When $\beta = 1$, problems (11.3) and (11.4) are identical. In this section, we will focus on the first stage and solve problem (11.4).

11.2.1.3 Bit Allocation Models

In ROI video coding, let us denote by N the number of macroblocks in the frame, $\{\rho_i\}$, $\{\sigma_i\}$, $\{R_i\}$ and $\{D_i\}$ the set of ρ s, standard deviation, rates and distortion (sum of squared error) for the ith macroblocks. Thus, $R = \sum_{i=1}^{N} R_i$. We define a set of weights $\{w_i\}$ for each macroblock as:

$$w_i = \begin{cases} \dfrac{\alpha}{K} & \text{if it belongs to ROI} \\ \dfrac{1-\alpha}{(N-K)} & \text{if it belongs to Non - ROI} \end{cases} \tag{11.5}$$

where K is the number of macroblocks within the ROI. Therefore, the weighted distortion of the frame is:

$$D = \sum_{i=1}^{N} w_i D_i = [\alpha D_{RF}(f, \tilde{f}) + (1-\alpha) D_{NF}(f, \tilde{f})]^* 255^{2*} 384 \tag{11.6}$$

Hence the problem (11-4) can be rewritten as:

$$\text{Minimize D, such that } R \leq R_{budget} \tag{11.7}$$

We propose to solve (11.7) by using a modeling-based bit allocation approach. As shown in [21], the distribution of the AC coefficients of a nature image can be best approximated by a Laplacian distribution $p(x) = \frac{\eta}{2} e^{-\eta|x|}$. Therefore in [11], the rate and distortion of the ith macroblock can be modeled in (11.8) and (11.9) as functions of ρ,

$$R_i = A\rho_i + B \tag{11.8}$$

where A and B are constant modeling parameters, and A can be thought of as the average number of bits needed to encode non-zero coefficients and B can be thought of as the

bits due to non-texture information.

$$D_i = 384\sigma_i^2 e^{-\theta\rho_i/384} \tag{11.9}$$

where θ is an unknown constant.

Here we optimize ρ_i instead of quantizers because that we assume that there is an accurate enough ρ-QP table available to generate a decent quantizer from any selected ρ_i. In general, (11.7) can be solved by using Lagrangian relaxation in which the constrained problem is converted into an unconstrained problem that:

$$\underset{\rho_i}{\text{Minimize}}\ J_\lambda = \lambda R + D = \sum_{i=1}^{N}(\lambda R_i + w_i D_i) = \sum_{i=1}^{N}[\lambda(A\rho_i + B) + 384 w_i \sigma_i^2 e^{-\theta\rho_i/384}] \tag{11.10}$$

where λ^* is the solution that enables $\sum_{i=1}^{N} R_i = R_{budget}$. By setting partial derivatives to zero in (11.10), we obtain the following expression for the optimized ρ_i, that is:

$$\text{let}\ \frac{\partial J_\lambda}{\partial \rho_i} = \frac{\partial \sum_{i=1}^{N}[\lambda(A\rho_i + B) + 384 w_i \sigma_i^2 e^{-\theta\rho_i/384}]}{\partial \rho_i} = 0 \tag{11.11}$$

which is

$$\lambda A - \theta w_i \sigma_i^2 e^{-\theta\rho_i/384} = 0 \tag{11.12}$$

so

$$e^{-\theta\rho_i/384} = \frac{\lambda A}{\theta w_i \sigma_i^2} \tag{11.13}$$

and

$$\rho_i = \frac{384}{\theta}[\ln(\theta w_i \sigma_i^2) - \ln(\lambda A)] \tag{11.14}$$

On the other hand, since

$$R_{budget} = \sum_{i=1}^{N} R_i = \frac{384A}{\theta}\sum_{i=1}^{N}[\ln(\theta w_i \sigma_i^2) - \ln(\lambda A)] + NB \tag{11.15}$$

so,

$$\ln(\lambda A) = \frac{1}{N}\sum_{i=1}^{N}\ln(\theta w_i \sigma_i^2) - \frac{\theta}{384NA}(R_{budget} - NB). \tag{11.16}$$

From (11.14) and (11.16), we obtain the following model:

$$\rho_i = \frac{384}{\theta}\left[\ln(\theta w_i \sigma_i^2) - \frac{1}{N}\sum_{i=1}^{N}\ln(\theta w_i \sigma_i^2) + \frac{\theta}{384NA}(R_{budget} - NB)\right]$$
$$= \frac{R_{budget} - NB}{NA} + \frac{384}{\theta}\left[\ln(\theta w_i \sigma_i^2) - \frac{\sum_{i=1}^{N}\ln(\theta w_i \sigma_i^2)}{N}\right] \tag{11.17}$$

As mentioned in [20], another model can be obtained if assume a uniform quantizer, then the distortion is modeled differently from equation (11.9), and thus the model can be derived as:

$$\rho_i = \frac{\sqrt{w_i \sigma_i}}{\sum_{j=1}^{N}\sqrt{w_i \sigma_j}}\rho_{budget}. \tag{11.18}$$

It is also indicated that both models have a good performance which is close to the optimal solution.

11.2.2 Content Adaptive Background Skipping

The concept of content-adaptive frame/object/macroblock skipping has attracted a great deal of attentions recently. The trade off between spatial and temporal quality was first studied in [21], where a perceptual rationale is employed: that the human visual system (HVS) is more sensitive to temporal changes when the frame contains high motion activities and otherwise is more sensitive to spatial details. The same logic is also used by [22–25] in determining the skip modes. In [22], a weighted function of motion and variance of the residue was used to evaluate the target bits for objects in bit allocation, which assigned more bits to objects with a more complicated texture (with a higher variance) or more activity (with a higher motion). The skipping decision of objects are based on an optimization process of a cost function which considers both coded distortion owing to quantization error and skipped distortion owing to skipped objects. This approach will be difficult for applying in real-time video systems which have tight time constraints and are not able to obtain future frames in advance. In [23], an adaptive macroblock skipping approach was proposed for ROI transcoding, where thresholds for motion and MAAD (mean of accumulated absolute difference) of the residue are used to skip those inactive Non-ROI macroblocks. In [24], the decision for frame skipping is dependent jointly on the temporal and spatial contents of the video, and on the fullness of the buffer by using empirical rules. In [25], considering the HVS model mentioned above, the decision for frame skipping is determined adaptively by motion, quantization parameter and buffer status. The motion is evaluated based on the sorted version of the most recent motion activities, and a dynamically adjusted threshold that is coupled with available resources, spatial quality, quantization parameters and motion activity. Utilizing the HVS model, by avoiding skipping frames during high-motion scenes, superior temporal quality

is maintained. By skipping frames during low-motion scenes that are less temporally sensitive, coding bits can be saved for subsequent no-skipped frames, and spatial quality can be enhanced. Furthermore, in [25] overall temporal-spatial quality is enhanced when compared to the no-skipping and fixed-pattern solutions, given limited coding resources.

In this section, a low-complexity content adaptive background skipping scheme for ROI video coding is introduced. In this context, we use background and Non-ROI as exchangeable terms because Non-ROI in video telephony applications generally refers to background region. In this framework we consider background skipping jointly with frame-level and macroblock-level bit allocation. The skip mode is determined mainly by foreground shape deformation, foreground motion, background motion and accumulated skipped distortion owing to skipped background. A self-learning and classification approach based on the Bayesian model is proposed in order to estimate the number of skipped background (in the future frames) based on the context of motion and background texture complexity. In addition, a weighted rate control and bit allocation algorithm is proposed in order to allocate bits for the foreground and background regions.

In Figure 11.3, the system architecture of our ROI video coding system is shown, which follows a frame-by-frame processing sequence. The system adopts a ρ-domain frame-level rate control algorithm [12] and a weighted macroblock-level bit allocation algorithm. When a frame is fetched into the system, a greedy frame-level rate control module is called to assign a target ρ budget for the frame considering the remaining bits and the number of frames in the rate control window. The model is based on the assumption that the content complexity of the video frames in the rate control window is distributed uniformly and thus the bits should be allocated uniformly among the remaining frames. After that, the ROI of the frame is detected or tracked and the macroblocks in the frame are classified into ROI macroblocks and Non-ROI macroblocks. Then, motion estimation is conducted for all of the macroblocks in the current frame and the obtained motion information is used as a part of content cues in the following background skip mode decision. Once the decision, of whether or not to skip the current Non-ROI, is made, the ρ budget for current frame is adjusted, and then the macroblock-level bit allocation and the following DCT transformation, quantization and entropy coding are conducted in the same way as described in section 11.2.1.

11.2.2.1 Content-based Skip Mode Decision

Let us first define two filters $F(\{x_n\}, M, Th)$ and $G(\{x_n\}, M, Th)$, where $\{x_n\}$ is a set of real numbers in which x_n is the nth item, M an integer number and Th a threshed in the range of [0, 1], and

$$F(\{x_n\}, M, Th) = \begin{cases} 1 & x_n \textit{is greater than Th *100\% of items in } x_{n-M}, \ldots, x_{n-1} \\ 0 & \textit{otherwise} \end{cases} \tag{11.19}$$

and

$$G(\{x_n\}\, M, Th) = \begin{cases} 1 & \textit{if} \dfrac{x_n - x_{n-M}}{x_{n-M}} \geq Th \\ 0 & \textit{otherwise} \end{cases} \tag{11.20}$$

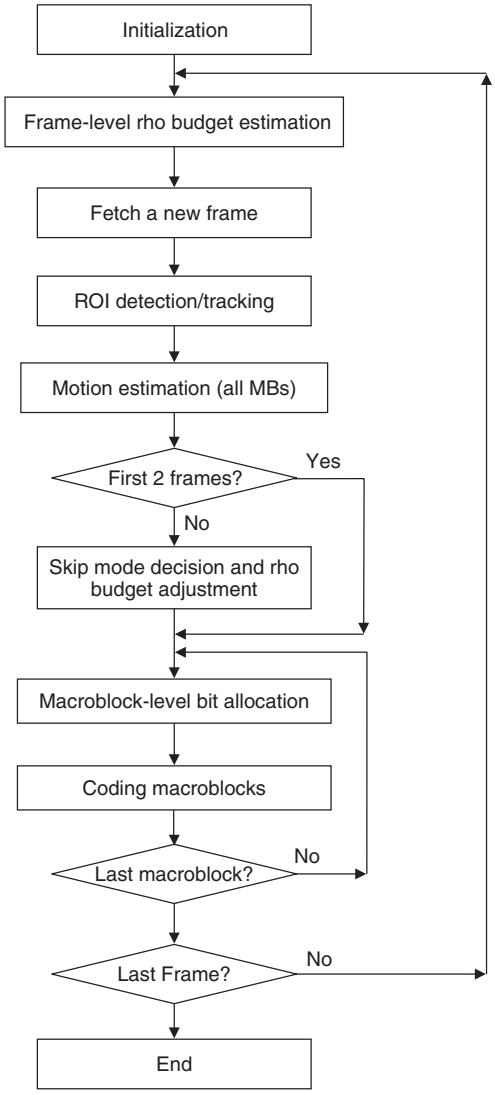


Figure 11.3 System architecture of the ROI video coding

Filter (11.19) detects within a local window (fixed length of M) if the current value x_n is in the top position (above more than $Th*100\,\%$ of items), and filter (11.20) detects if there is an increase from x_{n-M} to x_n by more than $Th*100\,\%$. These filters will be used in detecting the content status or status change, which indirectly affects the skip mode decision.

In [24, 25], the value of summed and averaged motion vectors in the frame (or recent frames) is used to represent the frame motion. The higher the motion the less skipping should be activated in order to protect possible content transition information. In ROI

video coding, both foreground and background activities are considered. When a large amount of motion occurs in background regions, the frequency of background skipping should be reduced. On the other hand, when the foreground contains a large amount of activities, the skipping of background might be helpful so as to reallocate more bits to code the foreground. Let us denote by $\{\chi_n\}$ the amount of background activity, and $\{\zeta_n\}$ the amount of foreground activity for the frame sequences, then:

$$\chi_n = \sum_{i \in Non\text{ - }ROI} (|MVx_i| + |MVy_i|) \tag{11.21}$$

where MVx_i and MVy_i are x and y component of the motion vector of ith macroblock in the nth frame, and:

$$\zeta_n = \mu_n \times \kappa_n \tag{11.22}$$

where $\{\mu_n\}$ is the ROI shape deformation factor and $\{\kappa_n\}$ is the ROI local movement factor, and

$$\mu_n = \frac{Number\ of\ pixels\ in\ nonoverlaped\ regions\ of\ ROIs\ of\ the\ (n\ -\ 1)th\ and\ nth\ frames}{Number\ of\ pixels\ in\ ROI\ of\ the\ nth\ frame} \tag{11.23}$$

and

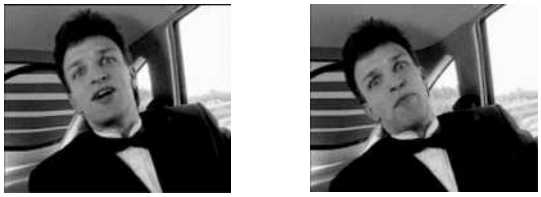
$$\kappa_n = \sum_{i \in ROI} (|MVx_i| + |MVy_i|) \tag{11.24}$$

Clearly, $\{\zeta_n\}$ can characterize the degree of the foreground activities because $\{\mu_n\}$ represents the degree of global activities such as object movement/rotation and shape deformation and $\{\kappa_n\}$ represents local activities such as change of facial expression. Two examples of these foreground activities are shown in Figure 11.4.

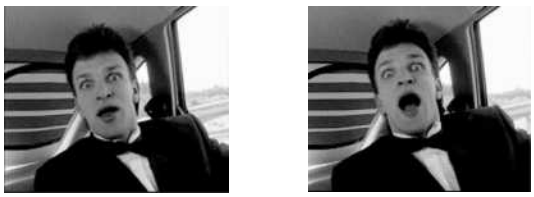
Let us denote by $\{\sigma^2_{B_n}\}$ the total energy of the background residue per frame for the frame sequence. Clearly it is also the distortion due to skipped background. So far, we can represent the skip mode decision:

$$\begin{aligned} S_n = {} & F(\{\zeta_n\}, M_2, Th_{\zeta 1})G(\{\zeta_n, 1, Th_{\zeta 2}\}) + [1 - F(\{\zeta_n\}, M_2, Th_{\zeta 1})G(\{\zeta_n\}, 1, Th_{\zeta 2})] \\ & [1 - G(\{\sigma^2_{B_n}\}, p, Th_\sigma)][1 - F(\{\chi_n\}, M_1, Th_{\chi 1})][1 - G(\{\chi_n\}, 1, Th_{\chi 2})] \end{aligned} \tag{11.25}$$

where Th_σ, M_1, $Th_{\chi 1}$, $Th_{\chi 2}$, M_2 and $Th_{\zeta 1}$ are thresholds and local window sizes defined by users, and $p-1$ the number of consecutive preceding frames of the current frame skipped background (in other words, the $(n - p)$th frame coded background but the $(n - p+1)$th, $(n - p+2)$th, ... and $(n-1)$th frames skipped background). When $S_n = 1$, the background of the current frame is skipped, otherwise, it is coded. Clearly from (11.25) it is observed that the system chooses to skip background when there is a sharp increase of the amount of foreground activity or the foreground contains large activity, otherwise, if background contains large motion or the accumulated distortion due to skipped background is rather high, then the background will be coded.



(a) Global activity (face movement)



(b) Local activity (change of facial expression)

Figure 11.4 Examples of frames with large activity in foreground

11.2.2.2 ρ Budget Adjustment

In Figure 11.3, the frame-level ρ budget estimation is based on an assumption that the whole frame is coded, however, in this system some backgrounds in the sequence will be skipped, therefore adjustment on ρ budget is necessary. Here we consider three types of strategies: 1) Greedy strategy, which simply reduces the ρ budget based on the texture complexity of ROI and Non-ROI when the skip mode is on, and does nothing if the background is coded; 2) *'Banker' strategy*, which reduces the ρ budget when the skip mode is on, but stores the savage of these ρ's for future frames. For a frame coding its background, it will obtain all the ρ's saved from the previous frames with background skipping; 3) *'Investor' strategy*, which estimates the future skipping events based on the statistics and patterns of the previous background skipping history, and then determines the ρ budget based on the estimation.

Let us denote by $\{\rho_n^{budget}\}$ the ρ budget obtained from the frame-level rate controller, $\{\rho_n^{adjusted}\}$ the adjusted ρ budget, and n the index of current frame. In the follows we describe more details of these strategies and compare them.

Greedy strategy

The $\rho_n^{adjusted}$ using this strategy can be calculated by

$$\rho_n^{adjusted} = \begin{cases} \rho_n^{budget} & if\, S_n = 0 \\ \dfrac{\sum_{i \in ROI} \sqrt{w_i \sigma_i}}{\sum_{i \in ROI} \sqrt{w_i \sigma_i} + \sum_{i \in NON\text{-}ROI} \sqrt{w_i \sigma_i}} \rho_n^{budget} & otherwise \end{cases} \tag{11.26}$$

where σ_i represents the standard deviation of the DCT coefficients of the ith macroblock in the current frame, and w_i is the associated weights for the macroblock in macroblock-level

weighted bit allocation as defined in section 11.2.1. Equation (11.26) comes as an extension of equation (11.18).

'Banker' strategy

This strategy is a conservative approach similar to the traditional banking operation, where the customer can cash out the maximum of the total deposit of his account. In this case, the saving of ρ's in frames with background skipping seems to deposit the resource for the nearest future frame which codes its background. The calculation for adjusted ρ budget is obtained by:

$$\rho_n^{adjusted} = \begin{cases} p\rho_{n-p+1}^{budget} - \sum_{i=1}^{p-1} \rho_{n-i}^{adjusted} & if\ S_n = 0 \\ \frac{\sum_{i \in ROI} \sqrt{w_i \sigma_i}}{\sum_{i \in ROI} \sqrt{w_i \sigma_i} + \sum_{i \in NON\text{-}ROI} \sqrt{w_i \sigma_i}} \rho_n^{budget} & otherwise \end{cases} \tag{11.27}$$

where $p-1$ is the number of consecutive preceding frames of the current frame with skipped background and the $(n-p)$th frame coded its background.

'Investor' strategy

A more aggressive approach is to predict future possible events and allocate resources based on the prediction. Here we assume that the future frames with skipped backgrounds have a similar complexity in foreground as the current frame, therefore, once we estimate that there will be q frames with skipped background following the current frame, we can calculate the adjusted ρ budget by:

$$\rho_n^{adjusted} = \begin{cases} p\rho_{n-p+1}^{budget} - \sum_{i=1}^{p-1} \rho_{n-i}^{adjusted} & if\ S_n = 0\ and\ n \le 50 \\ \frac{\sum_{i \in ROI} \sqrt{w_i \sigma_i} + \sum_{i \in NON\text{-}ROI} \sqrt{w_i \sigma_i}}{2(\sum_{i \in ROI} \sqrt{w_i \sigma_i} + \frac{1}{q+1} \sum_{i \in NON\text{-}ROI} \sqrt{w_i \sigma_i})} \rho_{\mathrm{n}}^{budget} + & \\ \frac{p\rho_{n-p+1}^{budget} - \sum_{i=1}^{p-1} \rho_{n-i}^{adjusted}}{2} & if\ S_n = 0\ and\ n > 50 \\ \frac{\sum_{i \in ROI} \sqrt{w_i \sigma_i}}{\sum_{i \in ROI} \sqrt{w_i \sigma_i} + \sum_{i \in NON\text{-}ROI} \sqrt{w_i \sigma_i}} \rho_n^{budget} & otherwise \end{cases} \tag{11.28}$$

In equation (11.28), the *'investor' strategy* acts exactly the same as the *'banker' strategy* for the first 50 frames. In this period the statistics are collected for future q estimation. When $n > 50$ and $S_n = 0$, ρ is assigned an average value considering the previous saving and the predicted future saving due to background skipping.

We estimate q by using a Bayesian model and convert the problem into a multi-class classification problem, where the classes are represented by all possibilities of q (for example, classes 0, 1, 2, 3, 4, 5 if we limit q to be less than 6), and the feature vector used in making classification decision is $x_n = \chi_n, \zeta_n, \sigma_{B_n}^2)$. By defining thresholds for χ_n, ζ_n and $\sigma_{B_n}^2$, we can map the space of $\{x_n\}$ into eight classes $\{y_n\}(y_n = 0, 1, \ldots, or\ 7)$.

Therefore, for current frame, the best selection for q is the one maximizing the probability:

$$P(q|y_n) = \frac{P(y_n|q)P(q)}{P(y_n)}, \tag{11.29}$$

thus it is the q that maximizes $P(y_n|q)P(q)$. The probabilities of $P(y_n|q)$ and $P(q)$ can be obtained by a histogram technique based on the statistics of the previously processed frames. Let us denote by $H_q(y)$ the counts of frames with coded background that follows q frames with skipped background with feature vector y, then:

$$P(y_n|q) = \frac{H_q(y_n)}{\sum_y H_q(y)} \tag{11.30}$$

and $P(q)$ can be obtained by the similar approach. The diagram of the skip mode decision and ρ budget adjustment module with this strategy is shown in Figure 11.5.

In Figure 11.6, three bit allocation strategies are compared in coding the Carphone sequence. As mentioned in section 11.2.1, an ROI perceptual importance factor α is defined in order to bridge the distortion of ROI and Non-ROI so as to form a weighted

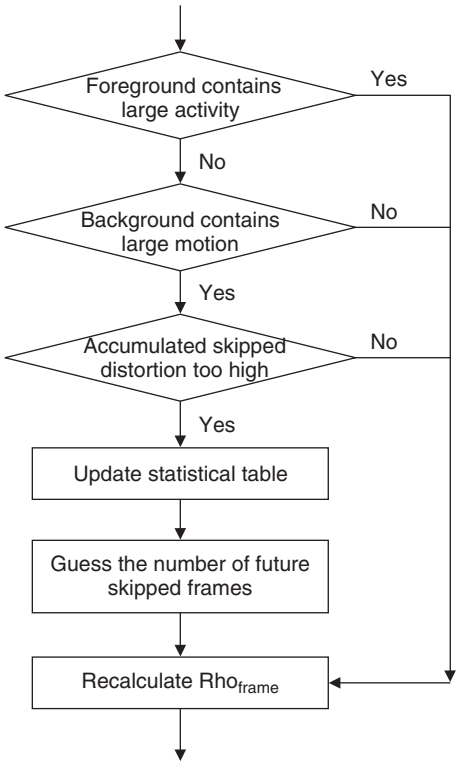


Figure 11.5 Diagram of the skip mode decision and rho budget adjustment module

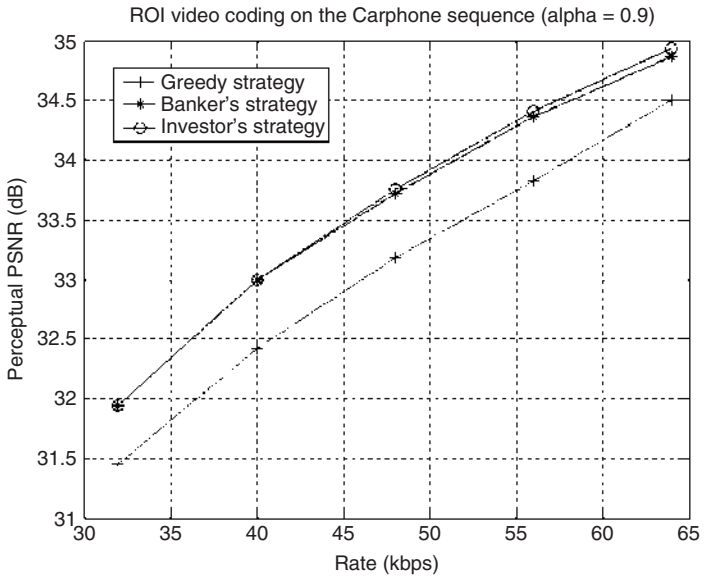


Figure 11.6 Comparison of three bit allocation strategies

distortion measurement for the frame. Therefore, the perceptual PSNR is defined as:

$$Perceptual\ PSNR = -10\log_{10}[\alpha D_R(f, \tilde{f}) + (1-\alpha)D_{NR}(f, \tilde{f})] \tag{11.31}$$

where f and $\tilde{f}$ are the original and reconstructed frames, and D_R and D_{NR} the normalized per pixel distortion of the ROI and Non-ROI. Clearly, both of the '*banker*' and '*investor' strategies* outperform the greedy strategy. The *'investor' strategy* slightly outperformed the *'banker' strategy* at higher bit rate end. Although it requires extra computational complexity for q estimation, this strategy might perform better for video sequences with repeated patterns or have self-similarity characteristics.

On the other hand, the 15 fps Carphone and other QCIF sequences at bit rates from 32 kbps to 64 kbps are tested in the H.263 Profile 3 simulations system. Four different rate control approaches are compared:

- Macroblock-level greedy algorithm [12] where the bits are allocated to the macroblocks in a uniformly distributed manner.
- Frame skipping algorithm that skips every other frame during encoding.
- Unit-based background skipping algorithm that groups every two frames into a unit and skips the background of the second frame within each unit.
- The proposed approach which content-adaptively determines the frames with skipped background, and uses the *'investor' strategy* for bit allocation.

As shown in Figure 11.7, the proposed approach outperformed all other approaches in the whole bit rate range and the gain is up to 2 dB. In Figure 11.8, the frame-level detail of

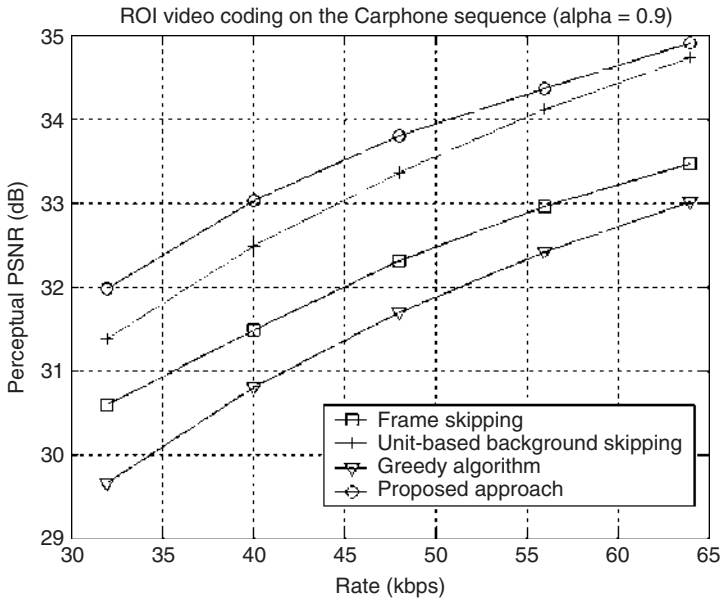


Figure 11.7 Comparison of various approaches in coding 'Carphone' sequence

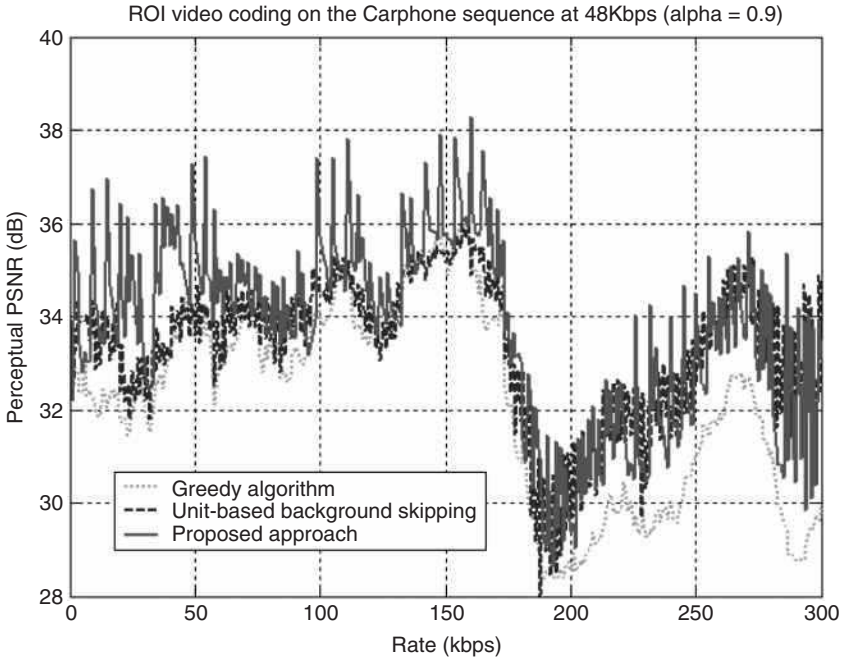


Figure 11.8 Comparison of various approaches at 48 kbps

Figure 11.9 Comparison of reconstructed frames by various approaches at 48 kbps

Figure 11.10 Visual artifacts due to background skipping

these algorithms at the 48 kbps is demonstrated. Figure 11.9 shows the reconstructed 15th frame for the compared algorithms and the advantage of the proposed approach is almost 5 dB compared to the greedy algorithm and 3 dB compared to the unit-based background skipping approach.

We have to point out that background skipping sometimes might cause visual artifacts if more than enough number of backgrounds are skipped, for example, as shown in Figure 11.10(b), the coded foreground and the background copied from the previous frame

are not aligned well thus causing artifacts at the collar. Clearly, this kind of artifact is very difficult to detect and be concealed because a certain degree of semantic information might be required in the processing. Further study on better background substitution or interpolation algorithms might be helpful in reducing such artifacts.

References

1. K. Maxwell, "Asymmetric digital subscriber line: Interim technology for the next forty years", *IEEE Commun. Mag*., October, pp. 100–106, 1996.
2. Eli Orr, "Understanding the 3G-324M Spec", can be downloaded from the weblink: http://www.commsdesign.com/designcorner/OEG20030121S0009.
3. H. Wang, K. El-Maleh, "Joint adaptive background skipping and weighted bit allocation for wireless video telephony", in *Proc. International Conference on Wireless Networks, Communications, and Mobile Computing*, Maui, Hawaii, USA, June 2005.
4. H. Wang, K. El-Maleh, and Y. J. Liang, "Real-time region-of-interest video coding using content-adaptive background skipping with dynamic bit reallocation", in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Toulouse, France, May 2006.
5. Y. J. Liang, H. Wang, and K. El-Maleh, "Design and implementation of ROI video coding using content-adaptive background skipping", in *Proc. IEEE International Symposium on Circuits and Systems*, Kos, Greece, May 2006.
6. M. Chen, M. Chi, C. Hsu and J. Chen, "ROI video coding based on H.263+ with robust skin-color detection technique", *IEEE Trans. Consumer Electronics*, Vol. 49, No. 3, Aug. 2003. pp. 724–730.
7. C. Lin, Y. Chang and Y. Chen, "A low-complexity face-assisted coding scheme for low bit-rate video telephony", *IEICE Trans. Inf. & Syst*., Vol. E86-D, No. 1, Jan. 2003. pp. 101–108.
8. S. Sengupta, S. K. Gupta, and J. M. Hannah, "Perceptually motivated bit-allocation for H.264 encoded video sequences", *ICIP'03*, Vol. III, pp. 797–800.
9. X. K. Yang, W. S. Lin, Z. K. Lu, X. Lin, S. Rahardja, E. P. Ong, and S. S. Yao, "Local visual perceptual clues and its use in videophone rate control", *ISCAS'2004*, Vol. III, pp. 805–808.
10. D. Tancharoen, H. Kortrakulkij, S. Khemachai, S. Aramvith, and S. Jitapunkul, "Automatic face color segmentation based rate control for low bit-rate video coding", in *Proc. 2003 International Symposium on Circuits and Systems (ISCAS'03)*, Vol. II, pp. 384–387.
11. J. Ribas-Corbera and S. Lei, "Rate control in DCT video coding for low-delay communications", *IEEE Trans. Circuits Systems for Video Technology*, Vol. 9, No. 1, pp. 172–185, Feb. 1999.
12. Z. He and S. K. Mitra, "A linear source model and a unified rate control algorithm for DCT video coding", *IEEE Trans. Circuits and System for Video Technology*, Vol. 12, No. 11, Nov. 2002. pp. 970–982.
13. H. Wang and N. Malayath, "Macroblock level bit allocation", US patent pending, May 2005.
14. H. Wang and K. El-Maleh, "Region-of-Interest coding in video telephony using Rho domain bit allocation", US patent pending, March 2005.
15. H. Wang and N. Malayath, "Two pass rate control techniques for video coding using a MINMAX approach", US patent pending, Sept. 2005.
16. H. Wang and N. Malayath, "Two pass rate control techniques for video coding using a rate-distortion characteristics", US patent pending, Sept. 2005.
17. T. Adiono, T. Isshiki, K. Ito, T. Ohtsuka, D. Li, C. Honsawek and H. Kunieda, "Face focus coding under H.263+ video coding standard", in *Proc. IEEE Asia-Pacific Conf. Circuits and Systems*, Dec. 2000, Tianjin, China, pp. 461–464.
18. C. Wong, O. Au, B. Meng, and H. Lam, "Perceptual rate control for low-delay video communications", *ICME'2003*. Vol. III, pp. 361–364.
19. S. Minami and A. Zakhor, "An optimization approach for removing blocking effects in transform coding", *IEEE Trans. Circuits Systems for Video Technology*, Vol. 5, No. 2, pp. 74–82, April 1995.
20. H. Wang, G. M. Schuster, A. K. Katsaggelos, "Rate-distortion optimal bit allocation scheme for object-based video coding", *IEEE Trans. Circuits and System for Video Technology*, July-September, 2005.

21. F. C. M. Martins, W. Ding, and E. Feig, "Joint control of spatial quantization and temporal sampling for very low bit rate video", in *Proc. ICASSP*, May 1996, pp. 2072–2075.
22. J. Lee, A. Vetro, Y. Wang, and Y. Ho, "Bit allocation for MPEG-4 video coding with spatio-temporal tradeoffs", *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 13, No. 6, June 2003, pp. 488–502.
23. C. Lin, Y. Chen, and M. Sun, "Dynamic region of interest transcoding for multipoint video conference", *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 13, No. 10, Oct. 2003. pp. 982–992.
24. F. Pan, Z. P. Lin, X. Lin, S. Rahardja, W. Juwono, and F. Slamet, "Content adaptive frame skipping for low bit rate video coding", in *Proc. 2003 Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing, and the Fourth Pacific Rim Conference on Multimedia*, Vol. 1, Dec. 2003, Singapore, pp. 230–234.
25. Y. J. Liang and K. El-Maleh, "Adaptive frame skipping for rate-controlled video coding", US patent pending, May 2005.

12

Wireless Video Streaming

12.1 Introduction

This chapter discusses the basic elements of wireless video streaming. Unlike download-and-play schemes, which require the entire video bitstream to be received by the client before playback can begin, video streaming allows a client to begin video playback without having to download the entire bitstream. Once video playback starts, it can continue without interruption until the end of the presentation. In order to enable playback without interruption even when the network bandwidth fluctuates, a client initially buffers the data it receives and begins playback after a delay of up to several seconds. This delay is fixed and does not depend on the length of presentation [1]. In order to achieve continuous playback, the interval between the time a video frame is transmitted by the server and the time it is displayed by the client should be the same for all frames. This means that there is a deadline for each frame when all packets that correspond to the frame must be available to the client for display. If some packets are missing at the deadline, they will be considered lost and error concealment will be employed in the decoding of the video frame. If packets that missed the deadline happen to arrive at the client later, they will simply be discarded. This concept of deadlines for the video packets is central to video streaming and will be discussed in detail later in this chapter.

With respect to the number of clients, video streaming may be classified as point-to-point, multicast and broadcast [2]. In point-to-point video streaming, there is one server and one client (unicast video streaming). Video conferencing is a special case of point-to-point video streaming, which requires low latency (low initial buffering delay). An important property in point-to-point communication is whether feedback exists between the client and server. If it exists, the server is able to adapt its processing based on the information it receives from the client regarding the quality of the channel.

Broadcast video streaming involves one server and multiple clients (one-to all communication). A classic example of this is terrestrial or satellite digital television broadcast. Owing to the large number of clients, feedback is usually not feasible, limiting the server's ability to adapt to changing channel conditions.

4G Wireless Video Communications Haohong Wang, Lisimachos P. Kondi, Ajay Luthra and Song Ci
 ISBN: 978-0-470-77307-9

Multicast video streaming also involves one server and multiple clients, but not as many as with broadcast video streaming. Thus, it can be best characterized as one-to-many communication (as opposed to one-to-all). An example of multicast video streaming is IP-Multicast video streaming over the Internet. However, IP-Multicast is not widely available on today's Internet. Thus, multicast implementations at the application layer have been proposed.

As mentioned previously, there is typically an initial buffering delay before a client starts playback. In classic, non-interactive video streaming, this delay can be significant, up to several seconds. The viewer will have to tolerate the initial delay but, once the video starts playing, it will be delivered continuously at the correct frame rate. Again, this requires that the interval between the time a video frame is transmitted and the time it is displayed be constant for all frames. For interactive applications such as video conferencing, however, the delay (latency) cannot be large; otherwise it will be hard to have a conversation between two or more people. A typical maximum latency for interactive applications is 150 ms [2].

Depending on the application, video streaming may require real-time video encoding, or may only involve the transmission of an already encoded video bitstream. Applications that require real-time encoding are video conferencing (and all interactive applications), broadcast digital television and any streaming application where the input video sequence is live. However, the majority of video streaming applications utilize pre-encoded video. This removes any computational complexity constraints regarding video compression at the server and also allows for non-causal video compression techniques such as multi-pass encoding. However, real-time video encoding can adapt efficiently to changing channel conditions, whereas pre-encoded bitstreams offer limited flexibility. Scalable video coding may be used to encode these stored bitstreams so that transmission can adapt to changing bandwidth by adding or dropping enhancement layers.

12.2 Streaming System Architecture

The building blocks of a typical video streaming system are shown in Figure 12.1.

The video sequence to be transmitted is first source-encoded (compressed). Then, 'Application Layer QoS control' takes place [3], which adapts the bitstream according to network status and QoS requirements. Data are then transmitted using appropriate transport protocols. The inverse operations are performed at the client (receiver). We next discuss these operations in more detail.

12.2.1 Video Compression

As in all types of video communications, video compression has to be performed on the video sequence to be transmitted. Depending on the application, compression may be real-time or non-real-time. Real-time video compression is required for interactive applications, such as video conferencing, as well as in applications where the video sequence to be transmitted is live. In all other cases, non-real-time video compression may be used.

Since video transmission has to adapt to changing channel conditions, it is beneficial to use scalable video coding. Scalable coding provides an elegant solution for coping

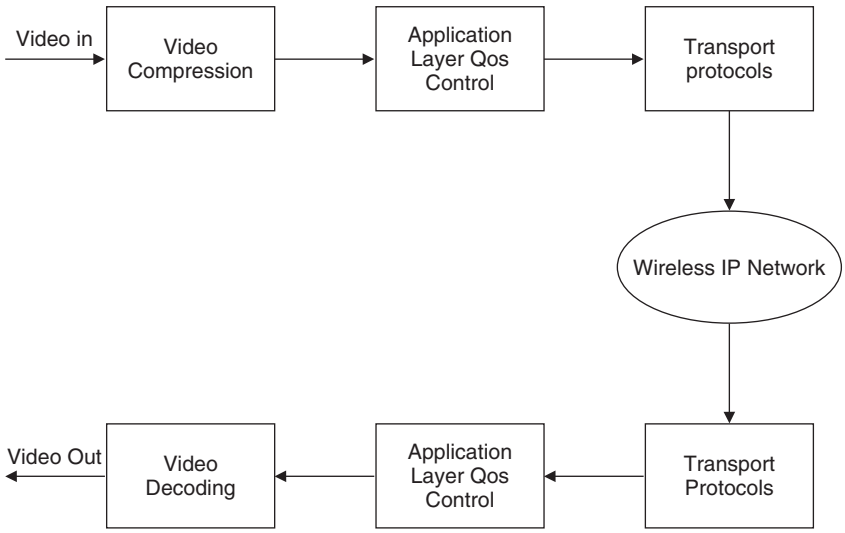


Figure 12.1 Video streaming system architecture

with bandwidth fluctuations, especially when compression is not done in real time and the bitstream is stored. In real-time compression, the target bit rate may be adjusted in non-scalable coding using rate control based on channel feedback. This is impossible to do in non-real-time compression, since the bistream has already been compressed at a specific target bit rate and stored. However, using scalable video coding, video transmission can adapt to changes in bandwidth by adding or dropping enhancement layers as appropriate, even if the bistream has already been compressed and stored.

Scalable coding is also very useful in the case of a server that serves multiple clients, which are connected to the network at different speeds (heterogeneous network). In that case, the base layer may be streamed to low-speed clients, while the base plus one or more enhancement layers may be streamed to faster clients.

Alternatives to scalable video coding for coping with changing channel bandwidths include *transcoding* and *multiple file switching* [2]. In transcoding, the video data are decompressed and recompressed again. The recompression may be done at a lower target bit rate in order to cope with bandwidth fluctuations. Other uses of transcoding are to achieve spatial downsampling, frame rate reduction or to obtain a video bitstream encoded using a different video compression standard (for example, MPEG-2 to H.264). Transcoding may address the problem of channel bandwidth fluctuations, but has two main drawbacks. The decoding and re-encoding operation typically results in some loss of video quality. Furthermore, the computational complexity of transcoding is significant, although there are ways to reduce it by reusing information selectively from the original bitstream (such as motion vectors and mode decisions) for the recompression.

In multiple file switching, more than one non-scalable bitstreams of the same video are encoded and stored. Each bitstream corresponds to a different target bit rate. In early video streaming implementations, the client had to choose from a set of bitstreams encoded at

different bit rates based on its connection speed (e.g. dialup, DSL, T1, etc.) Once a choice was made, the same bitstream was used for the whole session. Later, multi-rate switching became available, which allows for switching between different bitstreams within the same session, should channel conditions dictate it. SI and SP frames in H.264 provide an efficient way of switching between video bitstreams. Multiple file switching does not have the drawbacks of high computational complexity and reduced video quality, like transcoding. However, it requires multiple bitstreams to be stored at the server, which leads to increased storage costs. Furthermore, in practice, only a small number of bitstreams are used, thus, the granularity of available bit rates is small. This limits the ability of the system to adapt to varying transmission rates.

Multiple Description Coding (MDC) may also be used in video streaming and be combined with path diversity. Thus, if multiple network paths exist between the server and client, each description may be transmitted over a different path. Thus, the client will receive video even if only one path is operational.

12.2.2 Application Layer QoS Control

Application layer QoS control aims to maintain good video quality in the presence of changing channel conditions. Several QoS control techniques have been proposed for video streaming. Some are specific to video while others are applicable to general networking problems.

Network congestion leads to bursty packet losses that are detrimental to video quality. Congestion control aims to reduce congestion by appropriately matching the bit rates of the video stream to the available network bandwidth. There are two main mechanisms for congestion control: rate control and rate shaping [3]. We briefly discuss both of them next.

12.2.2.1 Rate Control

Rate control in the context of networking refers to determining the appropriate transmission bit rate that will minimize network congestion. It should be emphasized that the term 'rate control' has a different meaning here than in the context of video compression. In video compression, 'rate control' refers to algorithms that adjust coding parameters in order to meet a target bit rate, while, in the context of networking, it only refers to determining the transmission bit rate. There are three main types of rate control: Source-based rate control, receiver-based rate control and hybrid rate control [3]. These techniques may be used in general networking problems and are not specific to video.

In source-based rate control, the transmission rate is adapted by the transmitter based on feedback information. Depending on how the current network bandwidth is determined, source-based rate control may be probe-based or model-based. With the probe-based approach, the transmitter probes for the available network bandwidth by adjusting the transmission rate so that the packet loss rate is maintained below a certain threshold P_{th} [4]. The sending rate may be adjusted using (a) additive increase and multiplicative decrease [4], or, (b), multiplicative increase and multiplicative decrease [5]. With the model-based approach, instead of probing for the network bandwidth, the transmitter uses

the following equation (throughput model for a TCP connection) to estimate it [6].

$$\lambda = \frac{1.22 \times MTU}{RTT \times \sqrt{p}} \tag{12.1}$$

where λ is the throughput of a TCP connection, MTU (Maximum Transmission Unit) is the packet size used by the connection, RTT is the Round Trip Time for the connection, and p is the packet loss rate. The transmitter uses equation (12.1) to determine the video transmission rate. Using this model, video streaming data can compete fairly with TCP flows, thus decreasing the risk of congestion. Model-based rate control is also referred to as 'TCP-friendly' rate control.

In receiver-based rate control, the receivers (clients) adjust the receiving data rates by adding and dropping layers. Clearly, receiver-based rate control can only be applied to layered multicast. As in source-based rate control, receiver-based rate control may be probe-based or model-based. In probe-based receiver-based rate control, the receiver adds and drops multicast layers and monitors the resulting packet loss rate to probe the network bandwidth [7]. In model-based receiver-based rate control, the receiver uses equation (12.1) to estimate the network bandwidth.

In layered multicast, hybrid rate control may also be used, where the client adds or drops multicast layer and the server also adjust the transmission rate based on feedback by the clients [8].

12.2.2.2 Rate Shaping

Rate shaping refers to techniques which allow for the adjustment of the bit rate of a pre-compressed video bitstream in order to adapt to changing network conditions. Clearly, the most efficient and elegant way to do that is to use scalable video coding. Then, the server may add or drop enhancement layers according to the available network bandwidth. In the case of layered multicast and receiver-based rate control, the multicast layers are defined to coincide with scalable layers. Then, the receiver is able to add or drop layers in order to adjust the received bit rate.

There exist rate-distortion optimal techniques to determine a policy for transmitting and re-transmitting packets of a bitstream in the presence of feedback from the receiver. Such techniques will be discussed later in this chapter.

As mentioned earlier, multiple file switching may be used to adjust the transmission bit rate. Rate shaping may be performed (with limited success) even if only a single non-scalable pre-compressed bit stream is available. In [9], several 'rate filters' are proposed to adapt the bit rates of bitstreams. Most of these filters are also applicable to non-scalable bitstreams. We briefly describe these filters next.

- *Codec Filters.* Codec filters correspond to transcoding, as described earlier. The video bitstream is decompressed and then recompressed at a lower bit rate.
- *Frame-Dropping Filters.* As the name implies, frame-dropping filters drop frames in order to reduce the transmitted bit rate. B-frames are dropped first, since no frames depend on them, followed by P-frames and, finally, I-frames. Frame-dropping filters reduce the bit rate at the expense of a reduced frame rate.

- *Layer-Dropping Filters.* Layer-dropping filters drop enhancement layers from a scalable video bit stream, as described earlier.
- *Frequency Filters.* Frequency filters operate on the DCT coefficients of the compressed bitstream. Low-pass filters remove the high-pass coefficients from the bitstream. Color-reduction filters also perform low-pass filtering, but only on the chrominance components. Color-to-monochrome filters completely discard the chrominance information, thus converting the video to monochrome.
- *Requantization Filters.* Requantization filters perform 'inverse quantization' on the DCT coefficients and then requantize them using a larger quantization step size, thus reducing the bit rate at the expense of increased distortion. Frequency filters and requantization filters can be seen as special cases of transcoding.

12.2.2.3 Error Control

Error control is used at the application layer to ensure an acceptable QoS in the presence of adverse channel conditions. All the techniques that were described in Chapter 7 (error resilience, channel coding, error concealment) can be seen as forms of error control. If feedback is available from the receiver to the transmitter, retransmissions may also be employed. However, in video streaming, there is a strict deadline by which each packet needs to be received by the client in order to be useful. There is no point in attempting to retransmit a lost packet if, even it is received, it will be after the deadline. Thus, the retransmissions are *delay-constrained*. The problem of delay-constrained retransmission will be discussed in section 12.3.

12.2.3 Protocols

We next discuss the network protocols that are typically used in video streaming. We will assume that an Internet Protocol (IP) network is used. These protocols can be divided into three categories [3]:

- *Network-Layer Protocol.* IP is the network-layer protocol for video streaming and provides network addressing and other basic network services.
- *Transport Protocol.* The transport protocols provide end-to-end network transport functions. Some transport protocols used in video streaming are the User Datagram Protocol (UDP), Transmission Control Protocol (TCP), Real-time Transport Protocol (RTP) and Real-time Transport Control Protocol (RTCP). UDP and TCP are lower-layer transport protocols, while RTP and RTCP are upper-layer transport protocols, which are implemented on top of the lower-layer transport protocols.
- *Session Control Protocol.* Session control protocols are used to control data delivery during an established session. An example of a session control protocol is the Real Time Streaming Protocol (RTSP).

Figure 12.2 shows the relations between these categories of protocols. In a typical video streaming system, the compressed bitstream is packetized by the RTP layer, while the RTCP and RTSP layers provide control information. Then, the packetized stream is

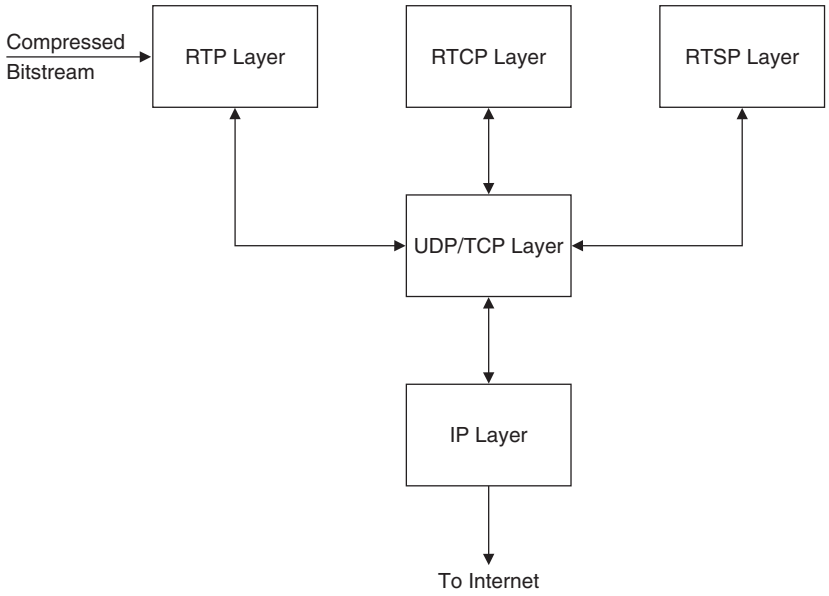


Figure 12.2 Protocols for video streaming

passed to the UDP/TCP layer and then to the IP layer. The inverse procedure is carried out at the receiver.

12.2.3.1 Transport Protocols

We next discuss the transport protocols in more detail. As mentioned earlier, these protocols include UDP, TCP, RTP and RTCP. UDP and TCP are lower-layer transport protocols while RTP and RTCP run on top of them. UDP and TCP provide multiplexing, flow control and error control. The main difference between UDP and TCP is that TCP guarantees that a packet will be delivered successfully via retransmissions, while UDP does not. However, the unlimited retransmissions that are employed by TCP make it impossible to meet the delay constraints associated with video streaming. Thus, UDP is typically used in video streaming. However, UDP does not guarantee packet delivery and an upper-level protocol such as RTP must be used to detect packet loss [3]. For more information on the delay constraints of video streaming, see section 12.3.

RTP is a protocol designed to provide end-to-end transport functions for real-time applications [10]. In addition to RTP, there is also RTCP, which provides QoS feedback to the participants of an RTP session. RTP provides the following functionalities:

- *Time-Stamping*. Time-stamping is used for the synchronization of different media types, for example, video and audio. RTP provides the time-stamping but the synchronization itself is done by the applications.

- *Sequence Numbering.* RTP assigns a number to each packet. Thus, if some packets are lost, this can be easily detected at the receiver. Also, if the packets arrive out of order, they can be put back in the correct order.
- *Payload Type Identification.* RTP puts payload type information in a packet header. Specific payload codes have been assigned for common payload types, such as MPEG-2, etc.
- *Source Identification.* RTP also puts information in the packet header regarding the source of the data so that the receiver may identify difference sources.

RTP is basically a data transfer protocol. Its companion protocol is RTCP, which is a control protocol. Participants of an RTP session typically send RTCP packets to report on the quality of the communication and other information. More specifically, RTCP provides the following functionalities [3]:

- *QoS Feedback.* QoS feedback is the primary function of RTCP. Applications receive feedback on the quality of data delivery. This feedback is useful for the senders, the receivers, as well as third-party monitors. QoS feedback is provided through reports from the senders and receivers. These reports contain information on the quality of reception, such as the fraction of lost RTP packets since the last report, the fraction of lost RTP packets since the beginning of reception, packet jitter and the delay since receiving the last sender's report.
- *Participant Identification.* RTCP transmits SDES (source description) packets that contain textual information about the session participants, such as name, address, phone number, email address, etc.
- *Control Packet Scaling.* In order to scale the RTCP control packet transmission to the number of participants, the total number of control packets is kept to 5% of the total session bandwidth. Furthermore, 25% of the control packets are allocated to sender reports and the other 75% of the control packets are allocated to receiver reports. At least one control packet is sent within five seconds at the sender or receiver in order to prevent control packet starvation.
- *Intermedia Synchronization.* RTCP sender reports contain timing information that can help in synchronization of different media types (for example, video and audio).
- *Minimal Session Control Information.* RTCP is also capable of transporting session information, such as the name of the participants.

In addition to RTP and RTCP, there is also RTSP, which is a session control protocol. The main function of RTCP is to provide VCR-like functionality, such as 'stop', 'rewind', 'fast-forward', etc.

12.2.4 Video/Audio Synchronization

In this chapter, we are concerned primarily with the streaming of video data. However, in most practical applications, video data are multiplexed and transmitted along with audio data. Thus, synchronization between video and audio is required in order to provide a pleasing user experience. If, for example, the audio is not synchronized with the lips of the speaker, this would be very annoying to the viewer. Or, in video streaming of a

football (soccer) game, if the audio of the play-by-play announcer is not synchronized with the video, it is possible that the announcer will be heard shouting 'Goal!' before the ball can be seen passing the goal line.

A widely used method for synchronization is axes-based specification, or time-stamping [11]. In axes-based specification or time-stamping, timing information is inserted periodically in the media streams by the sender. These time-stamps provide a correspondence between the media streams and dictate how they should be presented together. This time-stamp information is used at the receiver to synchronize the media streams properly. As mentioned previously, the RTP protocol offers time-stamping functionality.

It should be noted that video/audio synchronization is not the only type of synchronization needed in streaming. For example, in distance learning, where slides are presented along with a narration audio stream, it is very important for the slides to be synchronized with the audio. If the narration does not correspond to the slide currently in display, the presentation will be very hard to understand.

12.3 Delay-Constrained Retransmission

The main characteristic of video streaming, which distinguishes it from off-line downloading is that the receiver begins playback before the entire bitstream is downloaded. Furthermore, once playback begins, it continues uninterrupted until the end of the presentation. The requirement for uninterrupted playback leads to delay constraints [2].

More specifically, there is a deadline by which each video frame has to be received and decoded. Let Δ be the time interval between displayed frames. Δ is the inverse of the frame rate. Thus, for a frame rate of 30 frames/s, Δ is equal to 33 ms, while for a frame rate of 10 frames/s, it is equal to 100 ms. Let us assume that the first frame in a video sequence (let's call it frame 0) that arrives at the transmitter at time $t = 0$ is displayed at the receiver at time $T > 0$. T includes the initial buffering delay that was mentioned in section 12.1. This buffering delay is designed to combat fluctuations in the channel bandwidth and also enable retransmissions. The initial buffering delay (and, subsequently, T) may be selected by the system designer (there is, of course, a minimum practical value of T, which depends on the encoding and decoding times and the minimum time required to transmit a video frame given the channel bandwidth and video coding target bit rate). A relatively small buffering delay is required for interactive applications, while a larger delay, of the order of several seconds, may be used for other applications.

Now, once the first frame has been displayed at the receiver, playback must continue at the original frame rate. Thus, the time interval between displayed frames must remain equal to Δ until the end of the presentation. This leads to the following deadlines for each frame of the video sequence:

- Frame 0 must be received and decoded by time T.
- Frame 1 must be received and decoded by time $T + \Delta$.
- Frame 2 must be received and decoded by time $T + 2\Delta$
- Etc.

In general, frame n must be received and decoded by time $T + n\Delta$.

This means that there is a strict deadline when all packets for a video frame need to be received. If one or more frame packets are not available by the deadline, error concealment will have to be used to estimate the missing information. If any packets arrive after their deadline, they will be discarded, since the video frame they belong to will have already been displayed.

For packet N that belongs to video frame n, we define the deadline:

$$T_d(N) = T + n\Delta \tag{12.2}$$

Thus, packet N needs to be delivered by time $T_d(N)$ in order to be useful.

Error control via channel coding (Forward Error Correction) can always be used in video streaming. If a feedback channel between the receiver and transmitter is available, retransmission of lost packets may also be employed. However, the above-mentioned delay constraints need to be taken into account. In general, retransmission may be used if the one-way trip time is short with respect to T.

We next discuss three non-rate-distortion-optimized techniques for delay-constrained retransmission for unicast video streaming: receiver-based, sender-based, and hybrid control [3].

12.3.1 Receiver-Based Control

In receiver based control, the receiver requests retransmissions from the transmitter. Its goal is to minimize the requests for retransmissions of packets that will not arrive before their corresponding deadlines. When the receiver detects the loss of packet N, it checks if:

$$T_c + RTT + D_s < T_d(N) \tag{12.3}$$

where T_c is the current time, RTT is the estimated round trip time, D_s is an appropriately chosen slack term, and $T_d(N)$ is the deadline for packet N. If inequality (12.3) is true, then the receiver requests the retransmission of packet N. The transmitter retransmits the packet upon reception of the receiver's request.

12.3.2 Sender-Based Control

In sender-based control, the transmitter makes the determination on whether to retransmit a packet. The main difference between the receiver-based and sender-based control is that, in the former, the receiver knows the deadline $T_d(N)$, while in the latter, the sender only has an estimate $T'_d(N)$. In sender-based control, once the sender receives information from the receiver that packet N is lost, it checks the inequality:

$$T_c + RTT/2 + D_s < T'_d(N) \tag{12.4}$$

If the inequality is true, then the sender retransmits packet N. It becomes clear that sender-based control also attempts to avoid retransmitting packets that are likely to miss their deadlines.

12.3.3 Hybrid Control

Hybrid control is a simple combination of sender-based and receiver-based control. Thus, the receiver may use inequality (12.3) before requesting retransmission and the sender may use inequality (12.4) to decide on whether to retransmit the packet.

12.3.4 Rate-Distortion Optimal Retransmission

It is clear from the above discussion that in video streaming using a non-scalable bitstream, the objective is to maximize the number of received packets through retransmissions while avoiding retransmissions that will likely prove useless due to delay constraints. Thus, each lost packet is retransmitted, if possible, subject to the delay constraints. However, in a scalable bitstream, there is a hierarchy. Thus, if a hierarchically more important packet is lost, it is pointless to try to retransmit a packet of lesser importance, since the latter will be useless, even if it is received successfully.

It should be pointed out that there are dependencies even in non-scalable bitstreams. For example, in a typical non-scalable compressed video sequence consisting of I and P frames (IPPPP...), decoding of the current frame requires successful reception of the previous frame. However, a P frame may still be decoded, with the use of error concealment, even if parts of the previous frame are not received.

In the general case of a video bitstream with dependencies, it is not clear how to design the best packet retransmission policy. For example, if several packets are lost, which ones should be transmitted first? There have been several research efforts in optimal streaming [1, 12–26]. Many of these works are not applicable strictly for video streaming and may also be applied to streaming of other types of media.

The framework in [1] assumes that the encoded data are packetized into *data units*. The server puts a data unit into a *packet* in order to transmit it. If the packet is lost, the corresponding data unit may be put into another packet and retransmitted. A packet may contain only a single data unit, whereas the same data unit can appear in multiple packets (through retransmissions).

As mentioned previously, there are dependencies in all scalable bitstreams. Such dependencies can be modeled as a *Directed Acyclic Graph (DAG)*. Figure 12.3 shows examples of dependency DAGs. Figure 12.3(a) shows the dependencies that exist in any embedded bitstream. Data unit 1 is the base layer while data units $2, 3, \ldots, L$ are the enhancement layers. In order for data unit 2 to be decodable, data unit 1 is required, etc. Figure 12.3(b) shows the dependency in a typical video sequence encoded using temporal scalability and I, P and B frames. A P frame depends on the previous I or P frame, while a B frame depends on two I or P frames. However, no frames depend on B frames. Figure 12.3(c) shows the dependencies for a typical MPEG Fine Granularity Scalability (FGS) video bitstream. Again, P frames depend on the previous I or P frame and also enhancement layers depend on the previous layer.

Thus, the dependence DAG specifies the dependencies between the data units in a video bitstream and is computed offline. Along with the DAG, the following information is stored for each data unit l: Its size B_l in bytes, its importance Δd_l (its differential distortion) and its timestamp $t_{DTS,l}$ (its deadline). The quantity Δd_l is the amount by

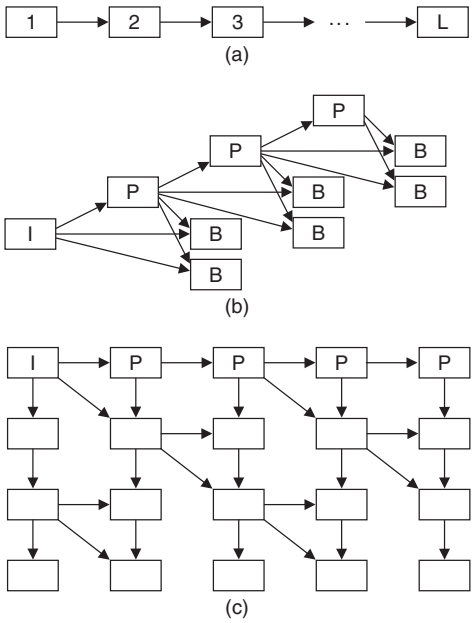


Figure 12.3 Directed acyclic dependence graphs. (a) Sequential dependencies typical of embedded codes. (b) Dependencies between IBBPBBPBBP video frames. (c) Typical dependencies for MPEG-4 progressive fine grain scalability mode. From [1], used with permission. Copyright IEEE, 2006

which the distortion of the received video will decrease if packet l is available to the decoder. t_{DTS}, l is the time by which data unit l must be received in order to be useful.

Chou [1] solves the video streaming problem in a rate-distortion framework. The rate R is defined as the expected cost of streaming the entire presentation (in our case, video sequence). The rate may refer to the total number of bytes transmitted, or, more generally, it may mean the total cost of transmitting the presentation. If the transmission of a data unit using a transmission option π has a cost per source byte $\rho(\pi)$, the cost of transmitting a data unit of size B bytes is $B\rho(\pi)$. Then, R is the expected value of the total cost, averaged over all possible realizations of the random channel for a given video sequence. The quantity D refers to the expected value of the total distortion of the video sequence, averaged over all possible realizations of the random channel for a given video sequence.

The objective of the video streaming algorithm is, given any presentation θ, to minimize the expected distortion $D = D_\theta(R)$ for an expected rate R. Each of the dots in Figure 12.4(a) denotes a possible (R, D) pair, which results from a specific retransmission policy. The dotted line shows the convex hull of all (R, D) pairs. The problem of minimizing the distortion subject to a given rate can be solved by minimizing the Lagrangian cost $D + \lambda R$ for some positive Lagrangian multiplier λ.

The problem of transmitting a single data unit is considered next. Chou [1] shows that the solution of the problem of transmitting a single data unit may be used as part of the solution to the main problem of minimizing the expected video distortion subject to a

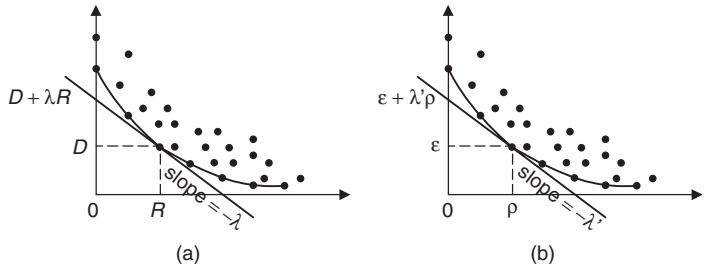


Figure 12.4 (a) Set of achievable distortion-rate pairs, its lower convex hull (dotted), and an achievable pair (R, D) minimizing the Lagrangian $D + \lambda R$. Each dot is the (R, D) performance of some algorithm. (b) Likewise, the set of achievable error-cost pairs, its lower convex hull, and an achievable pair (ρ, ε) minimizing the Lagrangian $\varepsilon + \lambda' \rho$. From [1], used with permission. Copyright IEEE, 2006

rate constraint. Let us assume that a transmission option (policy) π is used to transmit the data unit. The policy determines how the retransmissions are performed. Since we are talking about the transmission of a single data unit, it is possible to normalize its rate and distortion. Thus, the transmission of a data unit is associated with an expected cost ρ and an expected error ε. The cost ρ may be defined as the total number of transmissions of the data unit and ε may be defined as the probability of loss (unsuccessful reception) of the data unit (even after the retransmissions).

We assume that there are N transmission opportunities for data unit l before its deadline elapses. We denote these times as $t_{0,l}, t_{1,l}, \ldots, t_{N-1,l}$. Our problem is to determine which of these opportunities should actually be used to transmit the data unit. The objective is to minimize the error ε subject to a constraint on the cost ρ. This problem may again be solved using Lagrangian optimization, by minimizing the Lagrangian $\varepsilon + \lambda' \rho$ for some positive Lagrange multiplier λ'. Each of the dots in Figure 12.4(b) represents a (ρ, ε) point that can be achieved using a specific retransmission policy. The dotted line is the convex hull of these points.

Let us now concentrate on the scenario of sender-driven retransmission with feedback where the receiver sends an acknowledgement packet the instant that it receives the data packet. Obviously, once the sender receives an acknowledgment packet for a data unit, it will not attempt to transmit it again. The problem of transmitting a single data unit can be seen as a Markov decision process with finite horizon N. Such a process is represented by a trellis of length N. Any action that is taken at any state of the trellis influences the outgoing transition probabilities. A path through the trellis corresponds to a specific transmission policy for the data unit. Figure 12.5 shows an example of such a trellis. The trellis starts at time s_0, when the sender has to decide to either transmit the data unit, taking action $a_0 = 1$, or not to transmit it, taking action $a_0 = 0$. If the sender chooses to transmit the data unit, then, just before time s_1, it observes whether it has received a packet acknowledging successful reception of the data unit ($o_0 = 1$), or not ($o_0 = 1$). If the reception of the data unit has been acknowledged by time s_1, then the process enters a final state at time s_1. Otherwise, the sender decides again at time s_1 whether or not to transmit the data unit and observes before time s_2 whether it has received an

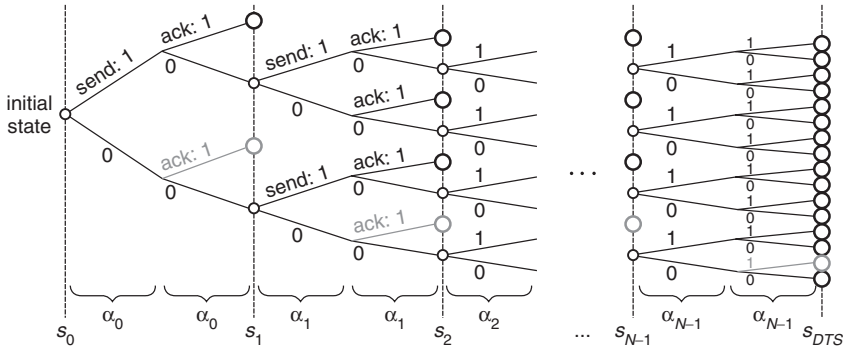


Figure 12.5 Trellis for a Markov decision process. Final states are indicated with double circles. From [1], used with permission. Copyright IEEE, 2006

acknowledgement. This process continues until an acknowledgement is received or the N transmission opportunities have been exhausted.

The trellis is used in the minimization of the Lagrangian cost $\varepsilon + \lambda'\rho$. This may be done using dynamic programming [14] or branch and bound algorithms [27].

We now concentrate on the original problem, which is the minimization of the distortion D subject to a constraint on the rate R for the presentation of the whole video sequence. This is equivalent to the minimization of the Lagrangian cost $D + \lambda R$. This problem is completely characterized by the dependence between the data units (the dependence DAG), the set of incremental distortions Δd_l, the packet sizes B_l, as well as $\varepsilon(\pi)$ and $\rho(\pi)$, the error and cost associated with a specific retransmission policy π for a packet. The dependence DAG, Δd_l and B_l depend on the source, source code and packetization, while $\varepsilon(\pi)$ and $\rho(\pi)$ depend on the transmission scenario and channel characteristics.

An iterative approach is proposed for minimizing the Lagrangian cost $D + \lambda R$, called the *Iterative Sensitivity Adjustment (ISA)* algorithm. This algorithm uses the solution of the problem of transmitting a single data unit (minimizing $\varepsilon + \lambda'\rho$) as part of the solution of minimizing $D + \lambda R$. More information can be found in [1].

In practice, a rate control algorithm is required for controlling the instantaneous rate of data packet transmissions dictated by the ISA algorithm. The rate control algorithm increases or decreases λ to adjust the number of data units selected for transmission at each transmission opportunity. Of particular interest is the case in which λ is adjusted so that exactly one data unit is selected for transmission at each transmission opportunity. Chou [1, 14] proposes an algorithm that selects λ in this case. The algorithm requires a series of approximations. The overall system proposed in [1] is called Rate-Distortion Optimized system (RaDiO).

12.4 Considerations for Wireless Video Streaming

The discussion on video streaming presented so far in this chapter is not specific to wireless transmission but may be also be applied to wired networks. However, wireless channels require additional considerations owing to the following problems [3]:

- *Bandwidth Fluctuations.* The available bandwidth of a wireless channel fluctuates with time. This may be due to a variety of reasons, including multipath fading, co-channel interference, noise disturbances and, mobile users, the changing distance between sender and receiver.
- *High Bit-Error Rate.* Wireless channels have a much higher bit error rate than wired channels due to fading and much higher noise levels.
- *Heterogeneity.* In a multicast scenario, different receivers may have different characteristics in terms of latency, visual quality, processing capabilities, power limitations and bandwidth limitations.

As mentioned previously, scalable video coding can deal effectively with the bandwidth fluctuations. Also, channel coding may be used instead of or in addition to retransmissions to cope with the higher bit error rates. Furthermore, the topics of error resilience and error concealment are more important in wireless video streaming due to the higher error rates. Scalable video coding used in conjunction with channel coding and unequal error protection is also very efficient for video transmission over wireless channels.

The ISA algorithm in [14] and [1] has been extended in [12] and [13] to wireless video streaming. In that work, forward error correction (channel coding) is used for error control in addition to retransmissions. An Incremental Redundancy (IR) scheme is used. In such a scheme, rate compatible channel codes are used. Thus, the bits for high-rate codes (less error protections) are subsets of the bits of low-rate codes (high error protection). The bits that correspond to the high-rate codes are transmitted first. If the bit errors cannot be corrected, more bits are transmitted. These bits are combined with the original received bits to create a lower-rate code, which may be able to correct the bit errors.

12.4.1 Cross-Layer Optimization and Physical Layer Consideration

The concept of *cross-layer optimization* and the related concept of *joint source-channel coding* are important in wireless video transmission. Shannon's *Principle of Separability* states that the design of source and channel coding can be separated without loss of optimality as long as the source coding produces a bit rate that can be carried by the channel (a rate that does not exceed the channel capacity). While being an important theoretical derivation, this principle relies on the crucial assumption that the source and channel codes can be of arbitrarily large lengths. In practical situations, due to limitations on the computational power and delay constraints, this assumption does not hold. Thus it is beneficial to consider the problems of source and channel coding jointly. Some representative works of joint source-channel coding for wireless video transmission include [28–32] and [33].

Cross-layer optimization can be seen as a generalization of joint source-channel coding. The *Open Systems Interconnection (OSI)* Reference Model specifies seven layers for communication systems [34]:

- Physical Layer.
- Data Link Layer.
- Network Layer.

- Transport Layer.
- Session Layer.
- Presentation Layer.
- Application Layer.

Traditionally, each of these layers has been considered separately. This makes system design easier. However, it has recently been shown that *cross-layer design and optimization* can be beneficial. Some recent work on cross-layer optimization for wireless video transmission includes [35–42] and [43]. Cross-layer design and optimization has been discussed in Chapter 9.

12.5 P2P Video Streaming

So far in this chapter, we have discussed video streaming from a server to one or more receivers. The topic of peer-to-peer (P2P) video streaming has gained interest in recent years, after the wide use and popularity of P2P file transfer. In P2P systems, there are no servers and all users are peers. Thus, the structure of a P2P system is decentralized. We will next discuss P2P video streaming systems briefly.

The advantages of P2P systems include their capability for self organization, bandwidth scalability and network path redundancy [44]. However, in P2P systems, peers typically join or leave the system rather frequently, making the system unstable. Furthermore, different peers may have different uplink and downlink bandwidths as well as different processing power. Thus, P2P systems lack any QoS guarantees and the problem of using them for streaming video data, which have strict delay requirements, is a challenging one.

P2P streaming systems rely on self-organization of the peers. There are two main network architectures used in P2P video streaming: *Tree-based overlays* and *mesh overlays*. In tree-based overlays, the peers are organized in a tree structure. The root of the tree is the source peer, while the leaves are the client peers. The intermediate peers in the tree push the video content from the source to the clients. Such architectures are easy to implement and maintain by the source. However, they suffer from high instability caused from peers joining or leaving the system. Also, each client is connected to the source over a single path. Thus, the available bandwidth is limited by the minimum upload bandwidth among the peers in the path.

In mesh overlays, the peers self-organize in a directed mesh. Thus, the data from the source peer are distributed among multiple paths. Each peer is connected to one or more parent peers and one or more child peers. Mesh overlays are more robust to peers entering and leaving the system than tree-based overlays. Furthermore, the existence of more than one path between the source and the client is very important.

The use of scalable video coding is appropriate in P2P video streaming in order to meet the constraints imposed by the bandwidth available at any given point in the network. Scalable video coding is also useful in cases where there is a large heterogeneity between the peers in terms of their access bandwidth and processing power [44].

Multiple Description Coding (MDC) may also be used in P2P video streaming. P2P systems typically offer multiple paths between a peer transmitting the video and a peer receiving it. MDC is a natural choice for this type of situation, since different descriptions may be transmitted via different paths. Thus, if a path becomes unavailable during the

course of the presentation (perhaps due to a peer leaving the system), video reception will still be possible via the other paths.

A fundamental problem in P2P streaming systems is how to select the best subset of paths to use between the source and client, and also how to determine the optimal rate allocation between the selected paths. There are two main ways of dealing with this problem: *Receiver-driven streaming* and *distributed path computation*. In receiver-driven streaming, the client coordinates the streaming process. Content location information can be accessed by the receiver at super nodes/servers as in BitTorrent or PPLive. Alternatively, such information may be obtained from other peers using search algorithms adapted to decentralized systems, or the receiver peer may just probe the network connections toward candidate source nodes [44]. Then, the client makes an informed decision of source peers and network transmission paths based on the network connectivity information and streaming session characteristics it received [45].

In practice, it is impossible for a client to receive accurate information about the topology of the whole P2P streaming system, especially if it is very large. In distributed path computation, each intermediate node makes an individual routing decision for each upcoming packet, based only on local topology information [46]. However, distributed path computation may lead to suboptimal streaming strategies, since no peer has complete knowledge of the network status.

References

1. P. A. Chou and Z. Miao, "Rate-Distortion Optimized Streaming of Packetized Media," *IEEE Transactions on Multimedia*, Vol. 8, No. 2, April 2006, pp. 390–404.
2. J. G. Apostolopoulos, W.-T Tan and S. Wee, "Video Streaming: Concepts, Algorithms and Systems," Technical Report HPL-2002-260, HP Laboratories, 2002.
3. Y. Wang, J. Ostermann and Y.-Q Zhang, *Video Processing and Communications*, Prentice-Hall, 2002.
4. D. Wu, et al, "On End-to-End Architecture for Transporting MPEG-4 Video over the Internet," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No. 6, September 2000, pp. 923–941.
5. T. Turletti and C. Huitema, "Videoconferencing on the Internet," *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, June 1996, pp. 340–351.
6. S. Floyd and K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet," *IEEE/ACM Transactions on Networking*, Vol. 7, No. 4, August 1999, pp. 458–472.
7. S. McCanne, V. Jacobson and M. Vetterli, "Receiver-Driven Layered Multicast," in *Proc. ACM SIGCOMM'96*, August 1996, pp. 117–130.
8. S. Y. Cheung, M. Ammar and X. Li, "One the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution," in Proc. *IEEE INFOCOM'96*, Vol. 2, March 1996, pp. 553–560.
9. N. Yeadon, F. Garcia, H. Hutchison and D. Shepherd, "Filters: QoS Support Mechanisms for Multipeer Communications," *IEEE Transactions on Selected Areas in Communications*, Vol. 14, No. 7, September 1996, pp. 1245–1262.
10. H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," IETF, RFC 1889, January 1996.
11. G. Blakowski and R. Steinmetz, "A Media Synchronization Survey: Reference Model, Specification, and Case Studies," *IEEE Journal on Selected Areas in Communications*, Vol. 14, No. 1, January 1996, pp. 5–35.
12. J. Chakareski and P. A. Chou, "Application Layer Error Correction Coding for Rate-Distortion Optimized Streaming to Wireless Clients," in *Proc. International Conference on Acoustics, Speech and Signal Processing*, Vol. 3, Orlando, FL, May 2002, pp. 2513–2516.
13. J. Chakareski and P. A. Chou, "Application Layer Error Correction Coding for Rate-Distortion Optimized Streaming to Wireless Clients," *IEEE Transactions on Communications*, Vol. 52, No. 10, October 2004, pp. 1675–1687.