Model-Assisted and Activity-Assisted Video Coding

Jae-Beom Lee

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

in the Graduate School of Arts and Sciences

Columbia University

2000

ABSTRACT

Model-Assisted and Activity-Assisted Video Coding

Jae-Beom Lee

Recent experiments have shown that human beings tend to be more sensitive to errors around semantic areas in a foreground object because human eyes lock on to and track such objects (e.g., a person's face). This is an important consideration in object-based video coding where scene components are captured and generated as objects that perhaps have different importance to human structural perception.

In this thesis, we take advantage of the aforementioned property to achieve better perceptual video coding in two ways. First, we develop a model-assisted very low bit rate video coding scheme that draws off the assumption that the incoming sequence is a video telephony type source. Second, we develop another method to replace the role of models – namely, activity-assisted video coding that does not rely on modeling the source. This method is applied not only for texture data but also for shape data.

In model-assisted coding, we propose a technique that selectively encodes areas of different importance to the human eye in terms of space and time for moving images. As a means of implementation, an object-based rate control algorithm is proposed using a rate distortion model.

We then develop a rate control algorithm for shape data based on buffer occupancy, thus making it a part of an activity-assisted video coding framework. We formulate the buffer-constrained adaptive quantization problem for shape coding, and offer an algorithm for the optimal solution under buffer constraints.

Activity-assisted coding is a natural extension of the model-assisted coding; if the model in a scene is not obvious, activity can function as a replacement for models. In this extension, a technique is introduced that assigns more bits to important unit blocks while preserving the same average bit rate of existing rate control algorithms. We first derive temporal and spatial balance equations. We then propose a technique that selectively allocates bits to areas of different activity in terms of space and time for video sequences. As a means to implement the new technique, an object-based rate control algorithm based on the form of optimal buffered compression is presented.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my sincere gratitude to Prof. Alexandros Eleftheriadis who supervised the research work contained in this thesis. I am particularly indebted to his guidance, his acute understanding of the rate control issues, and his encouragement to pursue my interests in very low bit rate video coding and content-based video coding. I was lucky to meet him as my advisor, and it was my great pleasure to work with him.

During the development of this thesis I also had the benefit of interacting with a number of colleagues and friends. I thank in particular Professors Shih-Fu Chang and Dimitris Anastassiou. Their deep understanding about media processing and compression always impressed me a lot. I was indebted to their helpful comments through several stages of my research. I thank as well my colleagues Di Zhong and Seung-Yup Paek for their nice help and hospitality. Special thanks go to my colleagues Daby Sow and Javier Gomez-Castellanos with whom I enjoyed life at Columbia, and also Sunil Patel, Arkady Kopansky, Chih-Kwan Wu, Jin-Soo Cho, Jin-Wha Yang, Ju-Youb Song, Danny Hong and Goun-Young Kim who assisted me in the development of theories in this thesis. I would also like to thank Drs. Dinei Florencios and Michael Isnardi for providing me with the opportunity of spending a very stimulating summer at Sarnoff Corporation.

The quality of the thesis has been significantly improved by the diligent efforts of the members of my defense committee: Profs. A. Eleftheriadis, S.-F. Chang, D. Anastassiou, S. Nayar and Dr. P. Tiwari whom I all thank. I want to express my warmful thanks to Dr. Prasoon Tiwari for his kind help at C-Cube Microsystems.

Finally, I would like to express my gratitude to my father, Doo-Yong Lee, my mother, Bong-Gum Lee, my wife, Yun-Hee Jang, and my daughter, Da-En Lee, for their support throughout the lengthy years during which this thesis was prepared.

Chapter 1

Introduction

1.1. Introduction

Current video coding standards, which enable video storage and broadcasting from medium to high bandwidths, provide excellent texture coding schemes and are classified as a first generation technology. The ISO MPEG-1 compression scheme [43, 56, 53] achieves 100 to 1 compression ratio while MPEG-2 achieves 50 to 1 compression ratio; only one or two percent of the original data size is generated from encoders, transmitted, and decompressed at decoders into data substantially the same as the original. The MPEG-1 and MPEG-2 standards were primarily targeted to provide high coding efficiency for the storage and transmission of pixel/sample-based video and audio, such as the compression of conventional video and audio for CD-ROM and digital television. The MPEG-1 video standard was primarily optimized for the coding of noninterlaced video at bit-rates of 1.2-1.5 Mbit/s, while the MPEG-2 video standard was primarily optimized for coding of interlaced video at bit rates of 4-9 Mbit/s [43, 45, 3].

What the MPEG-1 and MPEG-2 cannot do, however, is to achieve very low bit rates with acceptable visual quality. Images deteriorate rapidly at high compression ratios due to severe artifacts that appear in the decoded images [19].

One possible way to improve quality is to take into account the semantics of objects and to incorporate them into the coding systems. Techniques that are shifting towards this direction are referred to as second generation technology. In order to describe images/video more efficiently, second generation technology works on the understanding that the images are composed of different entities called *objects*, and it tries to take advantage of their perceptual properties.

After the successful completion of the MPEG-1 and MPEG-2 coding standards - which created wide interest in the area of digital audio-visual representation and communication - MPEG decided to start a new standardization effort in 1993, named the project MPEG-4, and intended completion by late 1998.

When MPEG-4 video was started, it was anticipated that with continuing advances in non-block based coding schemes – for example, in region based and model based coding – a scheme capable of achieving very high compression, mature for standardization would emerge [36, 7, 52, 34, 48, 76, 10, 11, 16, 57, 63, 88, 87]. This did not happen and its original focus was modified in July 1994, from coding with high efficiency of videophone scenes at very low bit rates to flexible coding of generic scenes that will facilitate a number of important functionalities for multimedia applications. Among the functionalities that were considered important for MPEG-4 was "object-based coding" [68, 15, 82, 64].

To provide a true, generic, object-based, and flexible multimedia environment, MPEG-4 integrates and supports coding for many audio-visual data types. Examples include natural and synthetic audio and video, as well as graphics, face and body animation, etc. Most importantly, a data representation is supported where access to the "object" in both natural and synthetic audio or video scenes is made available [68, 15].

In this thesis, we choose to focus on an issue of object-based video coding that

is sometimes called "content-based video coding." We will assume that readers are familiar with standards-based video coding techniques such as MPEG-1, MPEG-2, MPEG-4, H.261 and H.263 [43, 45, 3, 46, 81, 13] that are mentioned throughout this thesis.

1.2. Human Visual and Human Structural Perception

The core part of media compression (i.e., video or audio compression) is not "signal-only-compression," which is usually achieved by statistical techniques. The most important part of media compression is an understanding of the "human perception model," by which a certain portion of signals that are not significant for recognizing semantic objects can be eliminated.

For example, let's say that we are using a vector quantization technique (VQ) and that we have a codebook derived from certain training sequences. Also, let's assume that the codebook vectors are composed of certain high frequency patterns due to the properties of training data. What if human visual perception is not sensitive to image data resulting from patterns in the codebook? In this case, people probably cannot recognize the patterns or cannot tell the differences among them well, which is clearly not the purpose of compression. For another example we consider rate control mechanisms based on "objective-only-measurement," such as optimal buffered compression [60, 59, 72, 71]. What if the rate control algorithm wastes bits excessively on blocks to which human visual perception is not sensitive? In this case, people cannot see an improvement despite the fact that a larger bit budget was assigned. The result, simply stated, is a wasteful bit budget; therefore, an understanding of some basic aspects of human perception is essential to reduce imperceptible data in moving picture sequences.

One property of the human visual perception (HVP) that can be used in com-

pression systems is "activity masking." In portions of a picture where strong edges and rapid intensity changes occur, the eye has difficulty discerning small variations in intensity. Conversely, in very smooth or placid areas, the same small changes may be quite noticeable. Since image quality measurements note that the eye is drawn to areas that show artifacts, finer quantization values for DCT coefficients are appropriate for the areas where quantization artifacts are most visible. To take advantage of HVP – which usually recognizes errors more easily in smooth areas rather than in active ones (e.g., spatial masking effect) – some researchers have tried to allocate more bits to smooth areas [56, 67, 66, 65, 26].

On the other hand, some recent research has shown that human perception actively locks on to or tracks important semantic elements in the foreground of scenes [50, 51, 25, 23, 24, 22, 21, 18, 17, 20, 73, 74]. Although rapid motion is known to mask coding artifacts, the human visual system has the ability to lock on to and track specific moving objects, such as a person's face. Utilizing this human element is especially effective for very low bit rate video coding because in "noisy" situations, human perception locates key semantic elements first, whether the scene is active or smooth. This property was reported in the first competition stage of MPEG-4 video development by a group at AT&T [38, 33, 31, 32]. Throughout this thesis, we will call this property "Human Structural Perception" (HSP). For example, in very low bit rate videotelephony situations, state-of-the-art coding algorithms produce artifacts that are systematically present throughout the coded images. These artifacts usually affect all areas of the image without discrimination. Viewers, however, will generally find coding artifacts to be more noticeable in areas of particular interest to them. For example, a user of a videotelephony or video teleconferencing system will typically focus his or her attention on the face(s) of the person(s) on the screen, rather than on areas that show clothing or background.

Generally speaking, for foreground human objects, HSP works well; while for background objects, HVP generally dominates, because people don't lock on to and track background objects. Likewise, for very low bit rate video, HSP usually works well, and for high quality video, HVP tends to be the superior choice. It is because people try to locate important semantic elements first in noisy situations. Therefore, it is better to apply a bit distribution policy that fits each unique application.

1.3. HVP/HSP and Object-Based Rate Control

The rate control unit is the most important module in standards-based video compression systems. Rate control is the art and science of utilizing the limited buffer resources of encoders and limited network bandwidth resources to maximize quality. The main point is not to use the buffer resource for trivial visual information, but to maximize the use of it for important visual information. As we noted previously, the importance of visual information depends on characteristics of human vision or human structural perception.

In practice, the optimization of rate control schemes for HVP/HSP is a very complex task, and commercial implementors regard their optimization techniques as guarded property. The key of rate control is the intelligent incorporation of characteristics of HVP and HSP. In particular, in order to sustain a reasonable target bit rate and to prevent the encoding buffer from over- or under-flowing, resources should be accurately managed through quantization parameters (QPs). A QP usually determines the quantization step size of DCT coefficients in a macroblock (MB), thus controlling the quality of each MB in standards-based video compression systems. We would like to assert that if a visually important area is coded finely, other areas should be allowed to suffer in order to maintain a constant output target bit rate.

To incorporate the characteristics of HSP other than HVP into rate control, it is necessary to develop an intelligent rate control scheme with which one can assign different bit budgets to *different areas* of importance to human structural perception. To do so means that object-based rate control should be introduced to take advantage of the characteristics of HSP.

1.4. Thesis Contributions

In this thesis, we explore novel model-assisted and object-based video coding techniques that take advantage of HSP. Our approach with these techniques is to use semantic areas, such as objects in scenes, to take advantage of human perception. The unique properties of our proposal about object-based rate control algorithms point in two directions: rate distortion model based rate control and buffer occupancy based rate control, which are in the form of model-assisted coding and activity-assisted coding, respectively.

In Chapter 3, we delve into the concept of Spatio-Temporal Model-Assisted Compatible (STMAC) coding, a technique for very low bit rate video coding that selectively encodes areas of greater importance to the human eye. Our motivation is the fact that eye contact and lip synchronization are of primary importance in person-to-person communication. Fully automatic algorithms are described for performing model area detection and motion tracking. Following model detection algorithms, an Area Selective (AS) rate control scheme is presented; one that effects area-selective bit allocation. The decoder does not need to be changed in any way, though the encoder's rate control unit is modified. As a result, the scheme can be used in compliance with current standards-based codecs.

In Chapter 4 on MPEG-4 shape coding, we include an optimal buffered compression algorithm as defined in the forthcoming MPEG-4 international standard. The

MPEG-4 shape coding scheme consists of two steps: initially, distortion is introduced by down and up scaling; then, context-based arithmetic encoding is applied. Since arithmetic encoding is "lossless," the down-up scaling step is considered as a virtual quantizer. We formulate the buffer-constrained adaptive quantization problem for shape coding, and then propose an algorithm for the optimal solution under buffer constraints.

The optimal buffered compression framework [60, 59, 72, 71] only monitors the occupancy of the encoder's buffer; therefore, encoders can assign quite bad quantizer levels even for visually important macroblocks (MBs) when the buffer is nearly full. In Chapter 5, we continue with MPEG-4 shape and texture coding, and we introduce a technique that assigns more bits to visually important unit blocks while preserving the same average bit rate of existing rate control algorithms. We begin by deriving temporal and spatial balance equations. Then, Spatio-Temporal Buffer Rate Modulation (STBRM) – a technique that selectively allocates bits to areas of different activity in terms of space and time for video sequences – is explained. We also present a rate control mechanism that is proper to STBRM in the form of optimal buffered compression, providing a way to classify MBs (or Binary Alpha Blocks-BABs) into several categories that measure "activity levels." The most important implication of this work is to show a potential way that the performance of the optimal buffered compression technique proposed by Ortega and Vetterli can be achieved with subjectively enhanced visibility under certain conditions. We demonstrate this property by experimental results through MPEG-4 shape and texture codecs.

We cannot make a blanket statement about where to spend more bits in scenes; bit allocation truly depends on how important the objects are according to human eyes and which perception dominates in a particular case. Our semantics-based

activity-assisted rate control could be a way to take into account these concerns, considering that we are dealing with object-based coding such as MPEG-4. Thus, we can change the policy object-by-object. Essentially, our approach can use any policy based on activity, object-by-object. This strongly implies that the activity-assisted video coding is not limited to very low bit rate video coding applications only.

This thesis follows the historical trace of MPEG-4. Understandably, then, we will talk about a very low bit rate video coding technique in the first half of this thesis. We then will move on to object-based coding issues such as MPEG-4 shape coding and MPEG-4 texture coding in the second half. In the next chapter, we will begin our exploration with a brief overview of existing rate control schemes and a practical low bit rate video system, H.263, that will be directly or indirectly used for later chapters in this thesis.

Chapter 2

Video Rate Control

In the development process of international standards, there were several test, simulation, verification models where proposed algorithms were verified. In this section, we investigate the background ideas of several rate control schemes including some internal models of international standards. There are two major branches in the rate control schemes: buffer occupancy based rate control and rate distortion model based rate control. The advantage in using buffer occupancy is that the control of buffer occupancy is the objective of rate control [56]. As pointed out in [14], buffer occupancy depends on the position of the picture in the group of pictures (GOP). The other approach is developed based on rate distortion model. The advantage in controlling rate based on rate distortion model is that a picture-specific target rate can be used for each picture type, and even for each individual picture in a GOP. In this approach, however, any excess or deficiency in bits must be carried over from one control unit to the next. Otherwise, residual errors will accumulate and cause overflow or underflow [56]. Many practical rate control algorithms are a combined form of these two major approaches. Overall 6 different rate control schemes are examined in the following section. Among all, the first two rate control schemes (i.e., Reference Model 8 (RM8) and optimal buffered compression) are buffer occupancy based rate control, while others are mainly rate distortion model based rate control.

2.1. Rate Control Schemes

2.1.1 Reference Model 8 (H.261)

H.261 is a block-based, motion-compensated transform coding (DCT) design. The $p \times 64$ codec, H.261, uses only I-pictures and P-pictures. A FIFO buffer is assumed between the coder and the channel, and several times per picture is monitored for fullness. The quantization parameter, Q_p (proportional to quantization step size, or MQUANT as it is referred to in the standard) is increased or decreased according to whether the buffer is relatively full or empty. It prescribes the quantization of DCT coefficients using identical uniform quantizers with dead zones for all AC coefficients, and there is no perceptual frequency weighting.

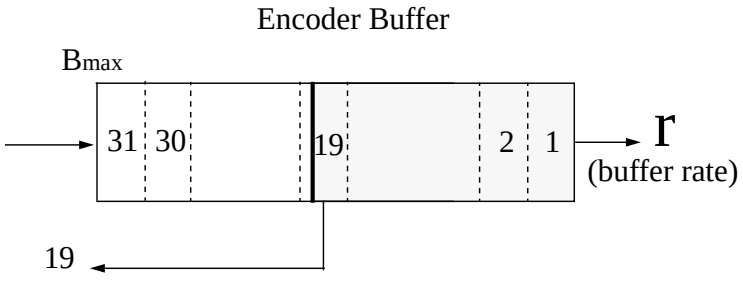


Figure 2-1: RM8 rate control.

RM8 is a "reference implementation" of an H.261 encoder that was developed and used internally by the H.261 working group, with the purpose of providing a common environment in which experiments could be conducted. The Q_p value is only dependent of current buffer occupancy as shown in Figure 2-1. To be specific,

the very first picture, which is an I-picture, is coded with a constant Q_p of 16. The output buffer is then initialized at 50% occupancy. For the remaining pictures, Q_p is adapted at the start of each line of MBs, where a rectangular array of 11×3 MBs defines a group of blocks (GOB). In other words, Q_p will be adapted three times within each GOB. The buffer occupancy is examined after the transmission of each MB and, if overflow occurs, the next MB is skipped. Note that this will result in a small temporary buffer overflow; in other words, the MB that caused the overflow will be transmitted (this can easily be done by using a "guard zone" in the output buffer). Q_p is updated "linearly" with the buffer occupancy according to the relation

$$Q_{p_i} = \min\{31, \lfloor \frac{B_i}{B_{max}/32} \rfloor + 1\}, \tag{2.1}$$

where Q_{p_i} is the value of Q_p selected for MB i, B_i is the output buffer occupancy just prior to coding MB i, and B_{max} is the output buffer size. A buffer size of $6400 \times q$ bits is used, given a bit rate of $q \times 64$ kbps of the video signal only.

2.1.2 Optimal Buffered Compression

Optimal trellis-based buffered compression is to provide optimal buffer control strategies for video sequences in a finite buffer environment [60]. Figure 2-2 depicts the optimal buffered compression approach, which is meant to select the optimal quantizer.

Optimal buffered compression and "optimal quantizer" are defined as follows.

Problem Definition: Given a set of quantizers, a sequence of blocks to be quantized, and a finite buffer, select the optimal quantizer for each block so that the total cost measure is minimized and the finite buffer is never in overflow.

Note that underflow is not included in this definition. An underflow constraint

would be redundant, given that the objective of minimizing distortion would tend to increase the coding bit rate and thus automatically penalize underflow.

Consider the allocation for N blocks, and suppose there are M quantizers available to code each block. Let d_{ij} and r_{ij} be, respectively, the distortion and the number of bits produced by the coding of block i with quantizer j, and let r be the channel rate per block. Define an admissible solution x as a selection of one quantizer for each block, i.e., a mapping from $1, 2, ..., N$ to $1, 2, ..., M$, $x = \{x(1), x(2), ..., x(N)\}$, where each $x(i)$ is the index of one of the M quantizers for block i. Therefore, $(r_{1x(1)}, ..., r_{Nx(N)})$ and $(d_{1x(1)}, ..., d_{Nx(N)})$ are, respectively, the rate and distortion for each block and a given choice of quantizers x.

Now, define the buffer occupancy at stage i, $B(i)$ for a given admissible solution x. Let $B(1) = r_{1x(1)} + B(0), B(2) = max(B(1) + r_{2x(2)} - r, 0)$ and, in general

$$B(i) = max(B(i-1) + r_{ix(i)} - r, 0) \tag{2.2}$$

where the buffer occupancy at each block instant is increased by the coding rate of the current block and decreased by the channel rate. $B(0)$ is the initial buffer state.

General Formulation : (Integer Programming)

The problem is to find the mapping x that solves

$$min(\sum_{i=1}^{N} d_{ix(i)}), \tag{2.3}$$

subject to

$$B(i) \leq B_{max}, \forall i = 1, ..., N$$

where B_{max} is the buffer size.

This problem is similar to the one known in the integer programming literature

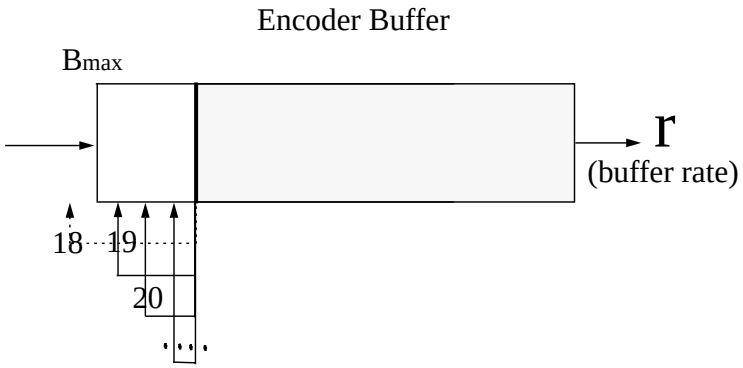


Figure 2-2: Optimal buffered compression is dealing with a marginal buffer to assign the best quantizer.

as the generalized assignment problem.

There could be many methods to solve that optimization problem under certain conditions. Dynamic programming to perform a sequential search through the set of possible solutions can provide a possible solution to the problem, while eliminating the ones that turn out to be suboptimal along the way. Generally dynamic programming [8, 9] can be used in finding the minimum cost path through a tree or trellis where each branch has an attached cost which is additive over the path. The main result of Bellman's optimality principle states in Figure 2-3 that if the minimum cost path from A_1 to A_4 passes through A_3 then its subpath going from A_1 to A_3 is also the optimal path from A_1 to A_3. Thus to find the optimal path we can set out to find sequentially the optimal path from the initial stage to successive stages, while knowing that any paths that have been pruned along the way would not have been global optimal.

In our context, we can use dynamic programming to sequentially eliminate sub-

optimal solutions. Let's grow a tree where each stage represents one block and where the different states represent a possible cumulative bit use up to that stage. Then we can rule out solutions for which there is an alternative providing less total distortion for the same rate. If two paths converge to the same node in the tree (i.e., the two solutions use the same number of bits for blocks considered), then we need only keep the minimum cost path. It can be interpreted to be a method to deal with a marginal buffer to assign the optimal quantizers for blocks under the assumption of independent and additive error measure. The "optimal" quantizers are a quantizer set which gives the smallest overall distortion in all possible quantizer combinations without overflow and underflow. To assign the optimal quantizer at each block is a dynamic programming problem to solve. A modified version of this idea was used used to solve the independent quantization problem of optimal buffered compression in [60, 59]. And the same theory was extended to a dependent quantization case in [72, 71].

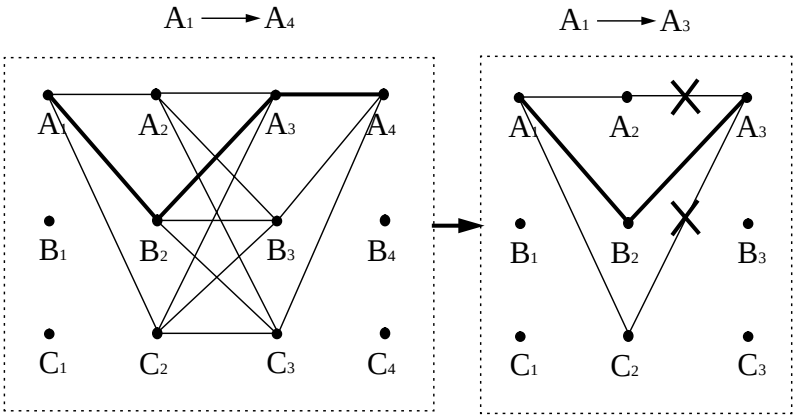


Figure 2-3: An example of dynamic programming.

2.1.3 Test Model Number 5 (H.263)

The idea of Test Model Number 5 (TMN5) rate control is to use an approximation of a general rate distortion function. A general rate distortion function is a convex function [28]. The first step of the derivation of a rate control formula is to approximate the rate distortion function by an inverse proportional curve as shown in Figure 2-4. If the rate distortion property is stationary, X will be a constant through blocks and frames. If X is a constant, we can see that

$$B_{previous} \times Q_{previous} = B_{current} \times Q_{current} \tag{2.4}$$

$$= X.$$

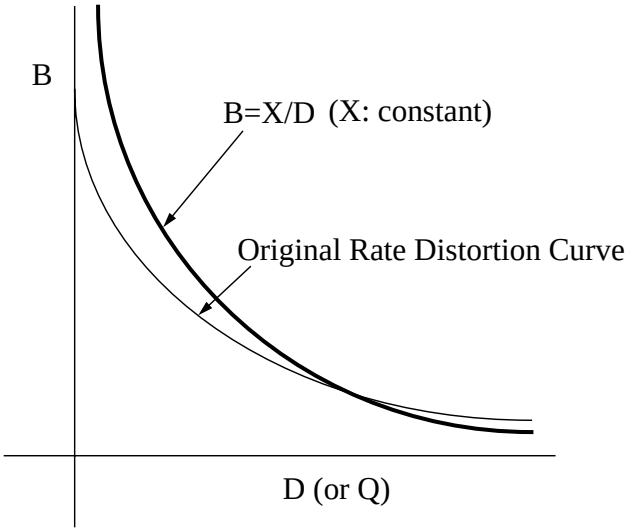


Figure 2-4: A rate distortion curve and its inverse proportional approximation.

Equation 2.4 can be represented by:

$$Q_{current} = Q_{previous}(1 + \frac{B_{previous} - B_{current}}{B_{current}}). \tag{2.5}$$

If we define $\triangle B$ as $B_{previous} - B_{current}$, then Equation 2.5 can be written as:.

$$Q_{current} = Q_{previous}(1 + \frac{\triangle B}{B_{current}}). \tag{2.6}$$

In H.263, rate distortion characteristic is quite stationary since generally there is only one picture type (i.e., P) except the first frame (i.e., I). So, this can be used as a quantizer parameter prediction from one frame to another frame. However, doing frame-based rate control is not good idea since a limited buffer can be easily overflowed or underflowed due to incoming sequences' randomness. Therefore, we break the equation into two parts once more to handle macroblock level rate control as follows:

$$Q_{current} = Q_{previous}(1 + \frac{\triangle B}{2B_{current}} + \frac{\triangle B}{2B_{current}}). \tag{2.7}$$

A good rate control algorithm usually adapts itself for a global and local utilization of resource simultaneously. To consider a global and local utilization of resource, we intentionally give the second term to adjustment of difference between target and actual bits for frames (i.e., inter), and do the third term to adjustment of the difference between target and actual bits for macroblocks (i.e., intra). A potential candidate for the rate control algorithm which assigns resources globally and locally can be the following:

$$Q_{current,MB} = \overline{Q_{previous,frame}}(1 + \frac{\triangle B_{frame}}{2B_{current,frame}} + \frac{\triangle B_{MB}}{2B_{current,MB}}). \tag{2.8}$$

The last step is to match the constant X compared with the inverse proportional graph and the actual data. This should be obtained by experiments. To consider the experimental factor into Equation 2.9, the above equation is modified with K as follows:

$$Q_{current,MB} = \overline{Q_{previous,frame}}(1 + \frac{\triangle B_{frame}}{2B_{current,frame}} + K\frac{\triangle B_{MB}}{2B_{current,MB}}). \quad (2.9)$$

There may be two different approaches for the experimental factor K. For each frame, first of all, K can be calculated based on previous frames' data to localize the formula for the consideration of randomness in incoming sequences. Secondly, we fix the K factor through frames as a constant, but we control the frame rate based on buffer occupancy. The second approach was taken in TMN5 rate control. The K factor was approximately defined to be 12/5 through experiments. We refer readers to the section 2.2.1 for detailed explanation about H.263.

2.1.4 Test Model 5 (MPEG-2)

The basic idea in Test Model 5 (TM5) is to maintain a fixed ratio between the average quantization parameters, Q_I, Q_P, and Q_B, where these are defined to be the arithmetic mean of the quantization parameters over all macroblocks in I-, P-, and B-pictures, respectively. TM5 is a rate control algorithm based on a rate distortion model. We have seen previously that a rate distortion curve can be approximated by an inverse proportional curve. This curve can be refined into actual data with an appropriate constant X, which is usually computed based on experiments. In particular, X values are different with different picture types. For example, the X value of I frames is generally higher than that of P frames since more bits are generated from MBs in I frames. Similarly the X value of P frames is higher than that of B frames. If we take the inverse proportional approximation, we have the following relationship.

$$B_I \times Q_I = X_I,\ B_P \times Q_P = X_P,\ and\ B_B \times Q_B = X_B. \quad (2.10)$$

If we divide the second expression by the first one, a relationship can be represented by:

$$\frac{B_P \times Q_P}{B_I \times Q_I} = \frac{X_P}{X_I}. \tag{2.11}$$

If we define K_P as Q_P/Q_I, the above expression is re-written by:

$$\frac{B_P}{B_I} = \frac{X_P}{X_I} \times \frac{1}{K_P}. \tag{2.12}$$

Note that K_P is defined by the ratio of "frame-averaged" quantization parameters since B_P, B_I, X_P and X_I are frame based expressions. The above ratio is expected to be less than 1, considering I frames generate more bits. A similar relation holds between B and I frames as follows:

$$\frac{B_B}{B_I} = \frac{X_B}{X_I} \times \frac{1}{K_B}, \tag{2.13}$$

where the definition of K_B is Q_B/Q_I. With the same reason, the above ratio is expected to be less than 1.

Since we already defined K_P and K_B, we can define the ratio of average quantization parameters between P and B frames with K_P and K_B as follows.

$$\frac{B_P \times Q_P}{B_B \times Q_B} = \frac{X_P}{X_B}. \tag{2.14}$$

Then, it can be re-written by:

$$\frac{B_P}{B_B} \times \frac{Q_P}{Q_I} \times \frac{Q_I}{Q_B} = \frac{B_P}{B_B} \times K_P \times \frac{1}{K_B} = \frac{X_P}{X_B}. \tag{2.15}$$

Therefore, we conclude that

$$\frac{B_P}{B_B} = \frac{X_P}{X_B} \times \frac{K_B}{K_P}. \tag{2.16}$$

To understand that K_P and K_B are maintained as constants is very important. That is, the purpose of rate control in TM5 is to maintain certain quality ratio through different picture types (i.e., I, P, and B frames). The quality of movies is acceptable to human perception based on experiments when the constants are taken as $K_P = 1$ and $K_B = 1.4$. To maintain the quality ratios, we intentionally fluctuate each frame's bit budget. Note that the value X, the constant value which fits the its approximation (i.e., an inverse proportional curve) to the actual data, is considered as a constant in the current picture, and is calculated from the closest picture of the same type.

TM5 consists of three steps: target picture bit budget allocation, MB quantization parameter assignment, and activity modulation at MB quantization parameters.

To allocate the target bit budget for a picture is easy if the same coding technique is applied to each picture and input pictures are stationary. For example, if the target bit rate is 1.5 Mbps and input is frame-based video, then each frame bit budget is supposed to be 50Kbps (i.e., 1.5/30 Mbps since the input is a video of 30 frames per second). This is, however, not the case in MPEG-video since each picture type such as I, P, and B uses different coding techniques, thus generating different amount of output bits. Therefore we should expect a different target bit budget for a different picture type. Generally we expect more picture bit budget for I pictures, while we do a smallest bit budget for B pictures. We define R as a left over bits expected in a GOP through current picture, so R is given with $R := R - B_{I,P,B}$ after encoding a picture, as shown in Figure 2-5.

Since R is a left over bits, the target picture bit budget, T, should be the following

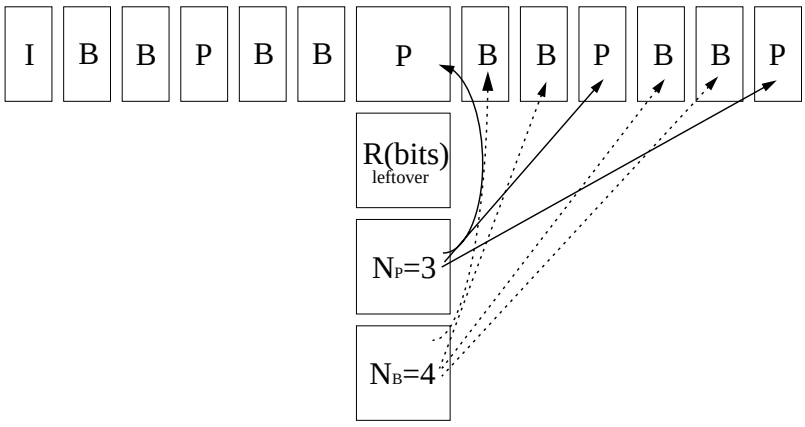


Figure 2-5: Parameters' implication in a GOP structure.

form:

$$T = \frac{R}{N_{left}}, \tag{2.17}$$

where N_{left} is the number of frames left over through current picture in a GOP.

For I frames, the target bit budget for a picture is expected to be higher than other two types. Thus, N_{left} is expected to be lower than others proportionally. Considering an I frame is the first in a GOP, $N_{left} = 1 + \alpha_I N_P + \beta_I N_B$. The values of N_P and N_B should be interpreted to be lower than the actual number (i.e. α_I and β_I factors) from the I frames' point of view, because P and B frames generate less bits as such. Note that the factors, α_I and β_I, can be taken aforementioned bit budget ratios, B_P/B_I and B_B/B_I. Therefore, we can set the picture target bit rate by:

$$T_I = \max\{\frac{R}{1 + \frac{B_P}{B_I} N_P + \frac{B_B}{B_I} N_B}, \frac{\overline{B}}{8}\} \tag{2.18}$$

for I frames,

$$T_P = \max\{\frac{R}{N_P + \frac{B_B}{B_P}N_B}, \frac{\overline{B}}{8}\} \tag{2.19}$$

for P frames, and

$$T_B = \max\{\frac{R}{\frac{B_P}{B_B}N_P + N_B}, \frac{\overline{B}}{8}\} \tag{2.20}$$

for B frames, where $\overline{B}$ is the average picture bit budget (i.e., target bit rate/picture per second). Note that to avoid buffer underflow, one eighth of the average picture bit budget is selected when the target bit budget allocation is too small.

The next step is to assign a quantization parameter to the current MB. In TM5 the quantization parameter, Q_j, is taken to be proportional to virtual buffer fullness, d_j,

$$Q_j = Ad_j, \tag{2.21}$$

where A is constant and j is the current macroblock number.

In the virtual buffer, a target MB bit budget is constantly going out by the value of T/MB_count, where MB_count is the total number of MBs in a picture. The purpose of MB level rate control is to spend the target picture bit budget entirely through MBs in the current picture. Therefore, the virtual buffer occupancy is updated after encoding each MB as follows:

$$d_j^I = d_0^I + b_{j-1} - [\frac{T_I \times (j-1)}{MB\_count}] \tag{2.22}$$

for I frames,

$$d_j^P = d_0^P + b_{j-1} - [\frac{T_P \times (j-1)}{MB\_count}] \tag{2.23}$$

for P frames, and

$$d_j^B = d_0^B + b_{j-1} - [\frac{T_B \times (j-1)}{MB\_count}] \tag{2.24}$$

for B frames, where d_0^I, d_0^P, and d_0^B are initial virtual buffer fullness at the start of each picture and b_j is the number of bits generated by encoding all macroblocks in the picture up to and including j.

In experiments of TM5 development process, the maximum size of the virtual buffer, r_{virmax}, was taken as "two times" the the average picture bit budget (i.e., $2 \times \overline{B}$). Therefore, the relation in Equation 2.21 can be re-written as follows:

$$Q_j = (\frac{31}{virtual\_buffer\_max})d_j = (\frac{31}{r_{virmax}})d_j. \tag{2.25}$$

For example, 31 is given to a quantization parameter when the current buffer occupancy, d_j, is the same as the maximum size of the virtual buffer, r_{virmax}. It is reasonable to give the highest value for a quantization parameter since the virtual buffer is about to overflow. Note that Q_j is updated "linearly" with the buffer occupancy according to Equation 2.25 when d_j is lower than r_{virmax}.

The last step is to modulate Q_j with a measure of the local spatial activity based on the minimum variance of the four luma blocks in the macroblock in TM5. As a rule of thumb, a given stimulus will be harder to see in a very active block - one with very complex and strong spatially-varying intensities. This characteristic is the so called "spatial masking effect." For example, in a portion of picture where strong spatially-varying intensities or rapid intensity changes occur, the eye has difficulty discerning small variations in intensity. Conversely, in very smooth areas, the same small changes may be quite visible [56]. This is taken into account in TM5. There are a lot of ways to define an activity measure. Activity measure in TM5 is obtained by taking a minimum value out of several activity candidates as follows:

$$ACT_j = 1 + \min\{vblk1, vblk2, ..., vblk8\}, \tag{2.26}$$

where the first half (i.e., $vblk1$, ..., $vblk4$) of total eight candidates are variances of luminance frame-organized sub-blocks and the last half (i.e., $vblk5$, ..., $vblk8$) of luminance field-organized sub-blocks [56].

The spatial activity, ACT, is normalized by an average value, AVG_ACT, obtained for the previous picture and mapped to the range 0.5 to 2.0 by a nonlinear function:

$$N\_ACT_j = \frac{2 \times ACT_j + AVG\_ACT}{ACT_j + 2 \times AVG\_ACT} \tag{2.27}$$

Note that N_ACT_j is the weighting factor for predetermined quantization parameters.

If we take $ACT_j = k \times AVG\_ACT$ and put it in Equation 2.27, it can be represented by:

$$N\_ACT_j = \frac{2 \times k + 1}{k + 2}. \tag{2.28}$$

Note that $k = 1$ (i.e., the current MB's activity is the same to the average activity) makes the weighting factor 1. Further, $k \to 0$ makes the value 0.5, while $k \to \infty$ makes it 2.0. That is, if the current MB's activity is lower than the average activity, the quality of the macroblock is enhanced on purpose. On the other hand, if the current MB's activity is higher than the average activity, the quality of the macroblock is degraded to take advantage of spatial masking effect.

If we define a new quantization parameter Q_{pj}, which is actually used for rate control, it can be found by the following relationship in TM5 for activity measure consideration:

$$Q_{pj} = N\_ACT_j \times Q_j. \tag{2.29}$$

2.1.5 Verification Model 5 (MPEG-4)

Recently a new rate control scheme is proposed in Verification Model 5 (VM5) for MPEG-4 video. The idea of VM5 is to use a quadratic approximation about rate distortion function. The rate control scheme proposes a way to allocate the target bit budget for each frame based on that approximation. The distortion measure is assumed to be the average quantization scale of a frame. The rate distortion function is modeled as a second order inverse proportional function of the distortion measure as shown in Figure 2-6. This scheme has a closed form solution for the target bit allocation which includes MPEG-2 TM5 rate control scheme as a special case.

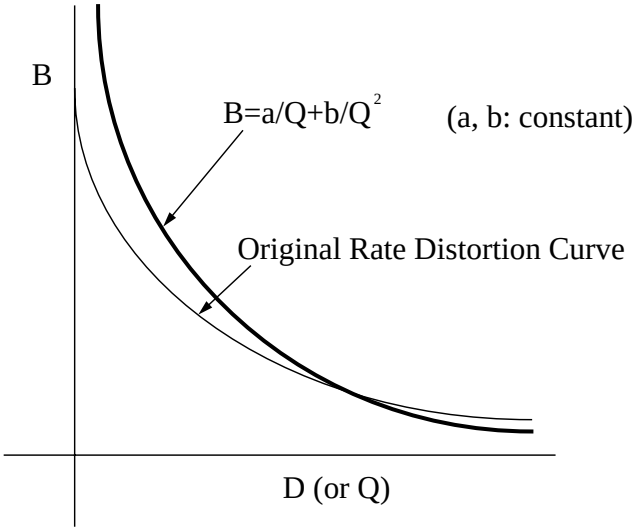


Figure 2-6: A second order function of the inverse of distortion measure.

As we see in Figure 2-6, the rate distortion function is approximated by a second order inverse proportional function as follows.

$$B = \frac{a}{Q} + \frac{b}{Q^2}. \tag{2.30}$$

In this formulation, a and b are constants in a picture, and the distortion measure is represented as the average quantization scale of a frame with a specific picture prediction type.

The encoder collects the bit rate and average quantization parameter for each type of picture at the end of encoding each frame. Then, the model parameters, a_I, a_P, a_B, b_I, b_P, b_B can be found. Since statistics of previous frames are known to the encoder, linear regression analysis and the formula below can be used to find parameters a and b.

$$b_k = \frac{n\sum_{i=1}^{n} B_i - (\sum_{i=1}^{n} Q_i^{-1})(\sum_{i=1}^{n} Q_i B_i)}{n\sum_{i=1}^{n} Q_i^{-2} - (\sum_{i=1}^{n} Q_i^{-1})^2} \tag{2.31}$$

$$a_k = \frac{\sum_{i=1}^{n} Q_i B_i - b_k Q_i^{-1}}{n} \tag{2.32}$$

where n is the number of frames in the past, Q_i and B_i actual encoding average quantization parameter and bit count in the past, and $k = I, P, B$. The scheme is not limited by the method of finding the parameters. They can be solved for target bit rates $T_I, T_P, and\ T_B$ based on the following six equations:

$$K_P \times Q_I = Q_P, \quad K_B \times Q_P = K_P \times Q_B, \tag{2.33}$$

$$T_I + N_P \times T_P + N_B \times T_B = R, \tag{2.34}$$

$$T_I = a_I \times Q_I^{-1} + b_I \times Q_I^{-2}, \tag{2.35}$$

$$T_P = a_P \times Q_P^{-1} + b_P \times Q_P^{-2}, \tag{2.36}$$

$$T_B = a_B \times Q_B^{-1} + b_B \times Q_B^{-2}, \tag{2.37}$$

where K_P, K_B are constant ratios for I, P, and B pictures, T_I, T_P, T_B target bit rates for I, P, and B pictures, Q_I, Q_P, Q_B average quantization parameters for I, P, and B

pictures, N_I, N_P, N_B frames to be encoded for I, P, and B pictures, and R remaining number of bits in the current GOP.

The first two equations make the assumption that we want to make a constant ratio between the average quantization parameters between individual picture types as was explained in the previous section. The constant K_P, and K_B are 1 and 1.4, respectively. The variable R is updated for every group of pictures according to the assigned bit rate. The last three equations state that a second order rate distortion function is assumed with the distortion measure as the average quantization parameters of the encoded picture. In the formulation, the distortion measure Q and the actual picture rate are found after the encoding of each picture. Using the information, the model parameters for the last three equations can be found. Based on the model found in parameter estimation, we can allocate the target bit budget before encoding. A closed form solution of the above six equations for the unknown $T_I, T_P, and\ T_B$ can be found.

Equations 2.36- 2.37 are substituted into Equation 2.34, thus making it:

$$T_I + N_P \times (Q_P^{-1} + b_P \times Q_P^{-2}) + N_B \times (a_B \times Q_B^{-1} + b_B \times Q_B^{-2}) = R. \tag{2.38}$$

Furthermore, from Equations in 2.36 Equation 2.38 can be written as:

$$T_I + N_P \times (Q_P^{-1} + b_P \times (K_P \times Q_I)^{-2}) + N_B \times (a_B \times (K_B \times Q_I)^{-1} + b_B \times (K_B \times Q_I)^{-2}) = R \tag{2.39}$$

and from Equation 2.35

$$(b_I + N_P \times b_P \times \frac{1}{K_P^2} + N_B \times b_B \times \frac{1}{K_B^2}) Q_I^{-2} + (a_I + N_P \times a_P \times \frac{1}{K_P} + N_B \times a_B \times \frac{1}{K_B}) Q_I^{-1} - R = 0. \tag{2.40}$$

The rest of the problem is to solve this quadratic equation for Q_I^{-1}.

The formulas are different for I, P, and B frames. To find T_I (Target bit rate for I frame), we define the following terms, which were used in the previous derivation.

$$\alpha = b_I + N_P \times b_P \times \frac{1}{K_P^2} + N_B \times b_B \times \frac{1}{K_B^2} \tag{2.41}$$

$$\beta = a_I + N_P \times a_P \times \frac{1}{K_P} + N_B \times a_B \times \frac{1}{K_B}$$

$$\gamma = -R$$

$$\delta = \beta^2 - 4\alpha\gamma$$

If $\delta < 0$, there is no real value solution for Q_I^{-1} for the quadratic equation. In this case, we take the first order inverse proportional approximation as follows.

$$T_I = a_I \times Q_I^{-1} \tag{2.42}$$

where $Q_I^{-1} = -\gamma/\beta$. If δ is not negative, there exists a real value solution for the quadratic equation. Therefore, T_I can be found by:

$$T_I = a_I \times Q_I^{-1} + a_P \times Q_I^{-2} \tag{2.43}$$

where $Q_I^{-1} = (\sqrt{\delta} - \beta)/2\alpha$. The same derivation is applied to T_P and T_B cases.

To find T_P (Target bit rate for P frame),

$$\alpha = N_P \times b_P + N_B \times b_B \times \frac{K_P^2}{K_B^2} \tag{2.44}$$

$$\beta = N_P \times a_P + N_B \times a_B \times \frac{K_P}{K_B}$$

$$\gamma = -R$$

$$\delta = \beta^2 - 4 \times \alpha \times \gamma$$

If $\delta < 0$, then

$$T_B = a_I \times Q_P^{-1} \tag{2.45}$$

where $Q_P^{-1} = -\gamma/\beta$. Otherwise,

$$T_P = a_I \times Q_P^{-1} + a_P \times Q_P^{-2} \tag{2.46}$$

where $Q_P^{-1} = (\sqrt{\delta} - \beta)/2\alpha$.

To find T_B (Target bit rate for B frame),

$$\alpha = N_P \times b_P \times \frac{K_B^2}{K_P^2} + N_B \times b_B \tag{2.47}$$

$$\beta = N_P \times a_P \times \frac{K_B}{K_P} + N_B \times a_B$$

$$\gamma = -R$$

$$\delta = \beta^2 - 4 \times \alpha \times \gamma$$

If $\delta < 0$, then

$$T_B = a_I \times Q_B^{-1} \tag{2.48}$$

where $Q_B^{-1} = -\gamma/\beta$. Otherwise,

$$T_B = a_I \times Q_B^{-1} + a_P \times Q_B^{-2} \tag{2.49}$$

where $Q_B^{-1} = (\sqrt{\delta} - \beta)/2\alpha$.

The above solution can be reduced to TM5 target bit allocation when the formula for $\delta < 0$ is always used.

2.1.6 Rate Control based on Feedback-Recoding

The rate control system in [29] is based on recoding. The basic unit for that control system is the slice, where in that work the slice is defined to be a macroblock row. The problem is to choose the quantization parameters to meet "exactly" the target bit budget. The target bits for a picture is postulated to be available from an algorithm such as in [27, 44, 66]. Global rates as a function of quantization parameter and picture type are shown, and are modeled with an equation of the form:

$$B = \alpha + \beta \times Q^{-\gamma}, \tag{2.50}$$

where $0 \leq \gamma \leq 2$ and B is the generated bits for a picture, α and β constants.

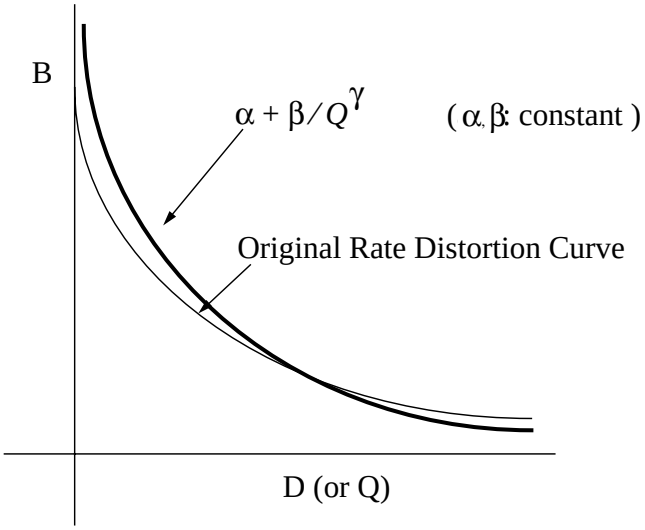


Figure 2-7: An exponential approximation of the rate distortion function.

For different frames and sequences, the values of the three parameters (α, β, γ) are different, and the good global fitting described above is not guaranteed for all frames and sequences. Therefore, the α, β, γ values should be "localized" for local fitting. Typically, I-pictures are bounded by $0.5 \leq \gamma \leq 1$ and P-pictures

by $0.5 \leq \gamma \leq 1.5$, which is reasonably consistent with the approximately inverse proportionality used in MPEG-2 TM5 and other systems. It was found to be more accurate and effective to fit the rate distortion curves by the model over a limited range with two parameters (α, β) and with fixed $\gamma = 1$. In other words, the curve can be localized with two points of (B, Q) through the "try and error" method.

The rate control system is used to adjust a reference quantization parameter at each slice. A trial encoding is performed using the reference quantization parameter and if the bits generated matches the target rate, that result is used; otherwise, the quantization parameter is adjusted up or down by an integer amount, depending on whether the rate is too high or too low. The amount of the adjustment is not computed from the Equation 2.50; rather, it is an integer of value 1, 2, or 4, approximately proportional to the reference quantization parameter. If necessary, the picture is recoded with the revised quantization parameter (i.e., hence the term, feedback recoding).

The quantization parameter can only take on integer values between 1 and 31. However, the target bit count may not correspond to an integral Q. If the same quantization parameter is applied to the entire picture, the actual bit count can differ significantly from the target bit count, as illustrated in Figure 2-8. To solve the problem, we will use the extra freedom of slice-based Q selection. For simplicity of discussion, we use each row of MBs' as a slice; other slice structures can be treated similarly. It is assumed that each slice of a picture contributes equally to the total bit count. Suppose that the rate distortion curve is estimated to be:

$$\hat{B} = B(Q). \tag{2.51}$$

Let the target bit count for a picture be B_t. Its corresponding quantization parameter is thus $Q_t = B^{-1}(B_t)$, which, in general, is not an integer. Let $Q_1 = \lfloor Q_t \rfloor$,

which is the largest integer not greater than Q_t. The number of bits $B(Q_1)$ or $B(Q_1+1)$ with all slices quantized by either Q_1 or (Q_1+1) may not be close to B_t. The difference $B(Q_1) - B(Q_1+1)$ is divided by the total number of slices, N_{slice}. Then the target bit B_t can be approximated by a closer amount,

$$N_1 = \lfloor \frac{B_t - B(Q_1+1)}{B(Q_1) - B(Q_1+1)} \times N_{slice} + 0.5 \rfloor \tag{2.52}$$

if N_1 number of slices are quantized with Q_1 an the rest $(N_{slice} - N_1)$ number of slices with (Q_1+1).

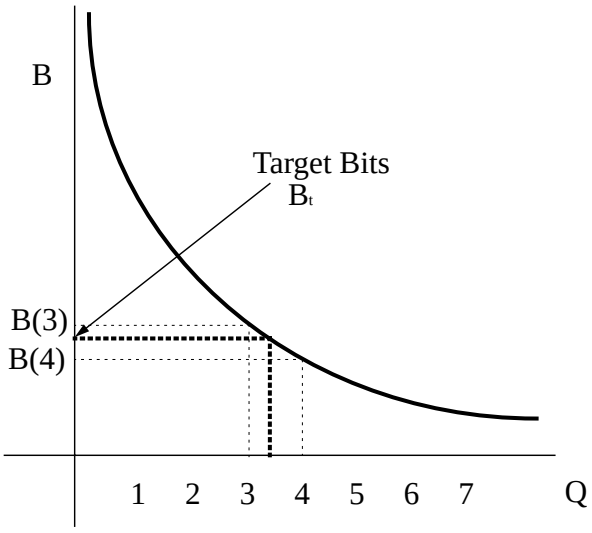


Figure 2-8: Illustration of slice Q selection with three slices. Equal division at B axis results in unequal division at Q axis. For the target number of bits, once slice will be quantized with $Q = 4$ and the other two with $Q = 3$.

For example, assume there are three slices in a picture (see Figure 2-8). If the three slices are quantized with $Q = 3$ or $Q = 4$, the total number of bits for the picture would be $B_{slice}(3)$ or $B_{slice}(4)$, respectively. Both generated bits are quite different from R_t. However, if two out of three slices are quantized with $Q = 3$ and

the other one with $Q = 4$, the total number of bits is $B(4) + \frac{2}{3}(B(3) - B(4))$, which is much closer to B_t. With more slices, finer division of the B axis will produce more accuracy. Another way to look at the problem is that a noninteger picture quantization parameter can be used. It is noted that even though the rate distortion model is estimated from integral frame Qs', it is used to determine noninteger picture quantization parameters.

Finally, the assumption of equal contribution from each slice is not always valid. To tackle this, we use a predetermined pattern to mix slices of different Q's to make it appear that quantization parameters are evenly distributed within the entire picture. For example, if five slices out of total 15 slices are to be coded with $Q = 3$ and the other 10 with $Q = 4$, we can mix them in the following pattern: (4, 3, 4, 4, 3, 4, 4, 3, 4, 4, 3, 4, 4, 3, 4).

To apply this into the rate control algorithm, first, we set the initial quantization parameter Q_0. Second, the target bit rate B_t for the current picture is allocated, such as in [44] and [67]. Third, we perform the normal motion compensation and DCT transform, and set $Q_1 = Q_0$, and encode the picture with Q_1 to obtain its bit count B_1. If $|B_1 - B_t| < \epsilon$, then we transmit the coding result with setting $Q_0 = \lfloor Q_1 \rfloor$, and go to the second step for the next picture. If $B_1 < B_t - \epsilon$, we let $Q_2 = Q_1 - \delta$, or if $B_1 > B_t + \epsilon$, we let $Q_2 = Q_1 + \delta$, and encode with Q_2 to get B_2. And, finally, we localize the rate distortion curve $\hat{B} = B(Q)$ from (Q_1, B_1) and (Q_2, B_2) by the try and error method. If $Q_t = B^{-1}(B_t)$ is between Q_1 and Q_2, the correct parameter is found. Then, we use the technique in the Figure 2-8 to quantize and code slices, set $Q_0 = \lfloor Q_t \rfloor$, and go to the second step for the next picture. Otherwise, we let $Q_1 = Q_2$ and $B_1 = B_2$, and go to apply the criterion of the third step.

2.2. Rate Control for H.263

In this section, we provide an overview about rate control algorithm and bitstream structure of a practical low bit rate video coding system, H.263, that will be directly or indirectly used for later chapters in this thesis.

2.2.1 Detailed Explanation about H.263 Rate Control

Besides the ISO standards, the ITU-T has also developed video and audio coding as well as multiplex standards. The ITU-T H.263 standards is aimed at coding of video at the very low bit rates of 10-24 kbit/s and is based on the earlier ITU-T H.261 video standard which was optimized at 64 kbit/s. In a general sense, the H.263 standard uses the motion compensated DCT coding framework which is also common to H.261, the MPEG-1 and the MPEG-2 standards. This consists of partitioning each picture into macroblocks, where a macroblock consists of 16×16 luminance (Y) block (composed of 4, 8×8 blocks) and corresponding 8×8 chrominance blocks of Cb, and Cr. Each macroblock can be coded as intra (original signal) or as inter (prediction error signal). Spatial redundancy is exploited by DCT coding. Temporal redundancy is exploited by motion compensation which is used to determine the prediction error signal. Block DCT coefficients are quantized and entropy coded. Details of H.263 include motion compensation with accuracy of half-pixel as well as optional modes such as PB-frames, unrestricted motion vector, advanced 8×8 block prediction, and syntax-based arithmetic coding. These modes are options that are negotiated between a decoder and an encoder. The ITU-T has continued work on further embellishing H.263 by adding more features and optional modes [41, 81].

The rate control scheme of H.263 is based on "variable frame rate" algorithm. Although rate control and buffer regulation are not subject to standardization, it is extremely important for the overall quality of video, at least at these very low

bit rates. In the standardization process for H.263, new coding techniques were evaluated with constant quantizer parameters and constant frame rates, in order to exclude the influence of the buffer regulation. However, this means that variable bit rate coding is used, giving near constant quality, what is in contrast with what can be achieved with real life fixed bit rate coding. Therefore, a buffer regulation strategy was developed so that realistic results could be obtained.

In very low bit rate coding, frame skipping is normally used as a means to reduce the bit rate while keeping acceptable picture quality. As the number of skipped frames is normally variable and depends on the buffer fullness, the question is how this should be related to buffer regulation by varying the quantizer step size. In fact, there are three basic types of buffer regulation:

1. Fixed frame rate, variable quantizer parameter
2. Fixed quantizer parameter, variable frame rate
3. Variable quantizer parameter, variable frame rate

With type 1, large fluctuations in picture quality will occur. With type 2, picture quality will be almost constant, but large fluctuations in frame rate will occur. With type 3, fluctuations will occur in both picture quality and frame rate, but the fluctuations can be moderate compared to type 1 and type 2. The buffer regulation in TMN5 is of type 3. For each picture, the following is calculated:

1. The target number of bits for the actual picture $\overline{B}$, based on mean quantizer parameter for the previous picture $\overline{Q_{pi-1}}$.
2. The initial quantizer parameter Q_{pinit} for the actual picture, based on $\overline{Q_{pi-1}}$, the number of bits spent for the already coded part of the actual picture.

3. The actual quantizer parameter for each macroblock, based on $\overline{Q_{pinit}}$, $\overline{B}$ and the number of bits spent for the already coded part of the actual picture.

For this buffer regulation, it is assumed that the process of encoding is temporarily stopped when the physical transmission buffer is nearly full. This means that buffer overflow and forced to fixed blocks will not occur. However, this also means that no minimum frame rate can be guaranteed. Which frame to be coded next depends on the number of bits in the buffer.

The following equations describe the TMN5 rate control algorithm:

$$Q_{pnew} = \overline{Q_{pi-1}}(1 + \frac{\triangle_1 B}{2\overline{B}} + \frac{12\triangle_2 B}{R}) \tag{2.53}$$

with

$$\triangle_1 B = B_{i-1} - \overline{B} \tag{2.54}$$

and

$$\triangle_2 B = (B_{a_*}) - \frac{a_*}{A_*}\overline{B} \tag{2.55}$$

where:

$\overline{Q_{pi-1}}$ is the mean quantizer parameter for the previous picture;

$\overline{B}$ is the target number of bits for picture;

R is the target bit rate;

B_{i-1} is the number of bits spent for the previous $(i-1)$ picture;

a_* is the index of the current macroblock;

A_* is the number of macroblocks in a picture;

B_{a_*} is the number of bits spent for the picture prior to coding the a_*-th macroblock;

The buffer content (for variable frame rate operation) is updated after each complete picture in the following way:

`buffer_content = buffer_content +` $B_{i,99}$ `;`

```
while (buffer_content > 3 · R/F ) {
        buffer_content = buffer_content- R/F;
        frame_incr++;
}
```

where F is the frame rate of source material. In other words, when the buffer occupancy exceeds an average of three frames, subsequent frames are skipped until the occupancy drops below that threshold.

The initial values and updated parameters are selected under the following way (as in TMN4):

The first picture is coded with $Q_p = 16$. After the first picture, as initial values buffer content is set to:

$$\frac{R}{f_{target}} + 3 \times \frac{R}{F}, \tag{2.56}$$

and the picture bit budget is set to:

$$B_{i-1} = \overline{B}. \tag{2.57}$$

At the start of each frame, the target frame rate is updated to:

$$f_{target} = 10 - \frac{\overline{Q_{pi-1}}}{4}, \tag{2.58}$$

and the picture bit budget is set to:

$$\overline{B} = \frac{R}{f_{target}}. \tag{2.59}$$

Note that the target frame rate f_{target} is allowed only in the following range:

$$4 < f_{target} < 10. \tag{2.60}$$

Finally, the calculated Q_{pnew} must be adjusted so that the difference fits in the definition of DQUANT where the value ranges in a set {-2, -1, 1, 2}.

2.2.2 Bitstream structure of H.263

The video multiplex is arranged in a hierarchical structure with four layers. From top to bottom the layers are:

- Picture
- Group of Blocks
- Macroblock
- Block

Unless specified otherwise the most significant bit is transmitted first. This is bit 1 (the left most bit). Unless specified otherwise all unused or spare bits are set to "1."

Picture Start Code (PSC) is a word of 22 bits. Its value is 0000 0000 0000 0000 1000 00. All picture start codes shall be byte aligned. Temporal Reference (TR) is an 8-bit number which can have 256 possible values. It is formed by incrementing its value in the previously transmitted picture header by one plus the number of non-transmitted pictures (at 29.97 Hz) since the previously transmitted one.

Type Information (PTYPE) is about picture type such as I, P and PB-frames. If bit 6-8 indicates a different source format than in the previous picture header, the current frame shall be an INTRA coded frame.

Quantizer information (PQUANT) is a fixed length codeword of 5 bits which indicates the quantizer QUANT to be used for the picture until updated by any subsequent GQUANT or DQUANT. The codewords are the natural binary repre-

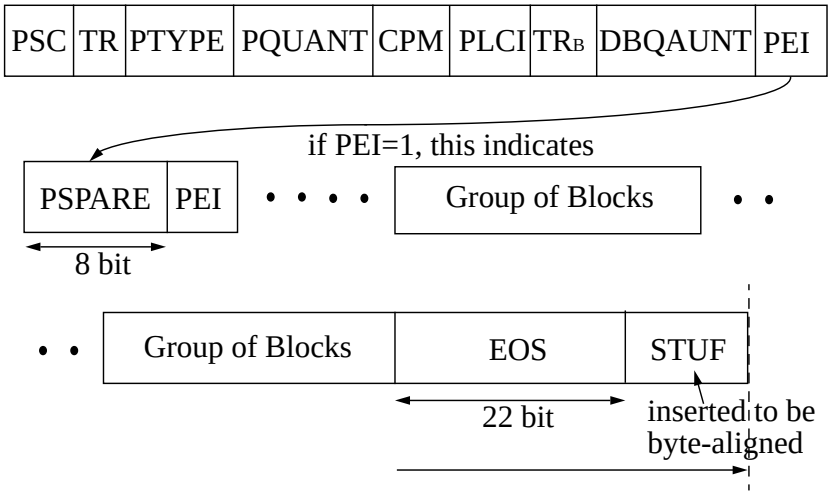


Figure 2-9: Structure of picture layer.

sentations of the values of QUANT which, being half the step sizes, range from 1 to 31.

Continuous Presence Multipoint (CPM) is a codeword of 1 bit that signals the use of Continuous Presence Multipoint mode (CPM); "0" is off, "1" is on.

Picture Logical Channel Indicator (PLCI) is a fixed length codeword of 2 bits that is only present if Continuous Presence Multipoint mode is indicated by CPM. The codewords are the natural binary representation of the logical channel number for the picture header and all following information until the next Picture or GOB start code.

Temporal Reference for B-frames (TR_B) is present if PTYPE indicates "PB-frame" and indicates the number of non-transmitted pictures (at 29.97 Hz) since the last P- or I-picture and before the B-picture. The codeword is the natural binary representation of the number of non-transmitted pictures plus one.

Quantization information for B-pictures (DBQUANT) is present if PTYPE indicates "PB-frame." In the decoding process a quantization parameter QUANT is obtained for each macroblock. With PB-frames QUANT and BQUANT as defined to be scaled as follows. If DBQUANT is 00, 01, 10, and 11, BQUANT is scaled by 5/4, 6/4, 7/4, and 8/4 of QUANT respectively. If the value for BQUANT resulting is less than 1 or greater than 31, it is clipped to 1 and 31.

Extra Insertion Information (PEI) is a bit which when set to "1" signals the presence of the following optional data field.

Spare information (PSPARE) is not used. If PEI is set to "1," then 9 bits follow consisting of 8 bits of data (PSPARE) and then another PEI bit to indicate if a further 9 bits follow and so on. Encoders shall not insert PSPARE until specified by the ITU. Decoders shall be designed to discard PSPARE if PEI is set to 1. This will allow the ITU to specify future backward compatible additions in PSPARE. If PSPARE is followed by PEI=0, PSPARE=xx000000 is prohibited in order to avoid start code emulation.

End Of Sequence (EOS) is a codeword of 22 bits. Its value is 0000 0000 0000 0000 1111 11. It is up to the encoder to insert this codeword or not. EOS may be byte aligned. This can be achieved by inserting less than 8 zero-bits before the start code such that the first bit of the start code is the first (most significant) bit of a byte.

Stuffing (STUF) is a codeword of variable length consisting of zero bits. Encoders may insert this codeword directly after an EOS codeword. The last bit of STUF must be the last (least significant) bit of a byte, so that the video bitstream including EOS and STUF is a multiple of 8 bits from the first bit in the video bitstream. Decoders shall be designed to discard STUF.

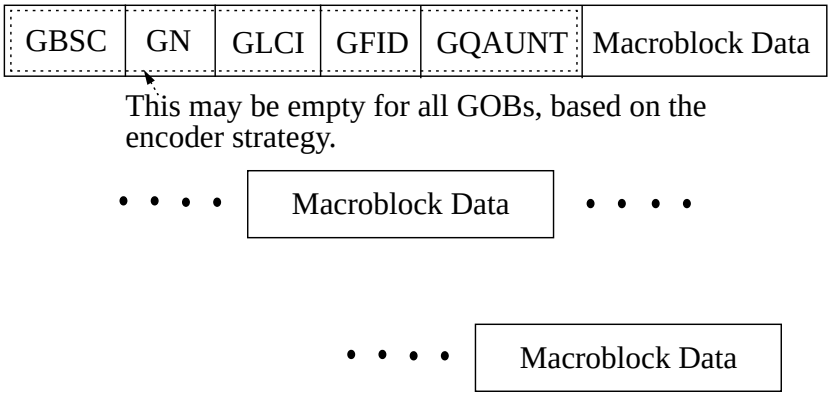


Figure 2-10: Structure of GOB layer.

Group of Block Start Code (GBSC) is a word of 17 bits. Its value is 0000 0000 0000 0000 1. GOB start codes may be byte aligned. This can be achieved by inserting less than 8 zero bits before the start code such that the first bit of the start code is the first (most significant) bit of a byte.

Group Number (GN) is a fixed length codeword of 5 bits. The bits are the binary representation of the number of the Group of Blocks.

GOB Logical Channel Indicator (GLCI) is a fixed length codeword of 2 bits that is only present if Continuous Presence Multipoint is indicated by CPM. The codewords are the natural binary representation of the logical channel number for the GOB header and all following information until the next Picture of GOB start code.

GOB Frame ID (GFID) is a fixed length codeword of 2 bits. GFID shall have the same value in every GOB header of a given picture.

Quantizer information (GQUANT) is a fixed length codeword of 5 bits which indicates the quantizer QUANT to be used for that GOB until updated by any subsequent DQUANT. The codewords are the natural binary representations of the values of QUANT which, being half the step sizes, range from 1 to 31.

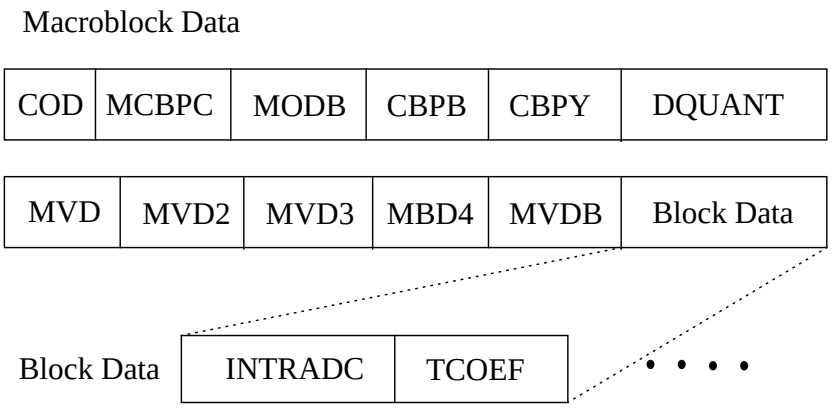


Figure 2-11: Structure of macroblock and block layers.

Coded macroblock indication (COD) is a bit which set to "0" signals that the block is coded. If set to "1," the block is not coded and the remaining part of the macroblock layer is empty; in that case the decoder shall treat the macroblock as an INTER block with motion vector for the whole block equal to zero and with no coefficient data. COD is only present in pictures for which PTYPE indicates "INTER," for each macroblock in these pictures.

Macroblock type and Coded block pattern for chrominance (MCBPC) is a variable length code giving information about the macroblock type and the coded block pattern for chrominance. MCBPC is always included in coded macroblocks. An extra codeword is available in the tables for bit stuffing. This codeword should

be discarded by decoders. The macroblock type gives information about the macroblock and which data elements are present. The coded block pattern for chrominance signifies C_B and/or C_R blocks when at least one non-INTRADC transform coefficient is transmitted. $CBPC_N = 1$ if any non-INTRADC coefficient is present for block N, else 0, for $CBPC_5$ and $CBPC_6$ in the coded block pattern. When MCBPC=stuffing, the remaining part of the macroblock layer is skipped. In this case, the preceding COD=0 is not related to any coded or not-coded macroblock and therefore the macroblock number is not incremented. For pictures coded in TNTER mode, multiple stuffings are accomplished by multiple sets of COD=0 and MCBPC=Stuffing.

Macroblock mode for B-blocks (MODB) is present for MB-type 0-4 if PTYPE indicates "PB-frame" and is a variable length codeword indicating whether B-coefficients and/or B-vectors are transmitted for this macroblock.

Coded block pattern for B-blocks (CBPB) is only present if indicated by MODB. $CBPB_N = 1$ if any coefficient is present for B-block N, else 0, for each bit $CBPB_N$ in the coded block pattern. The utmost left bit of CBPB corresponds with block number 1.

Coded block pattern for luminance (CBPY) is variable length codeword giving a pattern number signifying those Y blocks in the macroblock for which at least one non-INTRADC transform coefficient is transmitted. $CBPY_N = 1$ if any non-INTRADC coefficient is present for block N, else 0, for each bit $CBPY_N$ in the coded block pattern. The utmost left bit of CBPB corresponds with block number 1. For a certain pattern $CBPY_N$, different codewords are used for INTER and INTRA.

Quantizer information (DQUANT) is a two bit code to define change in QUANT. The differential values for the different codewords are given. QUANT ranges from 1 to 31; if the value for QUANT after adding the differential value is less than 1 or

greater than 31, it is clipped to 1 and 31 respectively.

MVD is included for all INTER macroblocks (in PB-frames mode also for INTRA macroblocks) and consists of a variable length codeword for the horizontal component followed by a variable length codeword for the vertical component.

Motion vector data (MVD_{2-4}) are three codewords included if indicated by PTYPE and by MCBPC, and consist each of a variable length codeword for the horizontal component followed by a variable length codeword for the vertical component of each vector.

Motion vector data for B-macroblock (MVDB) is only present if indicated by MODB and consists of a variable length codeword for the horizontal component followed by a variable length codeword for the vertical component of each vector.

A macroblock comprises four luminance blocks and one of each of the two color difference blocks. The structure of the block layer is shown Figure 2-11. INTRADC is present for every block of the macroblock if MCBPC indicates MB type 3 or 4. TCOEF is present if indicated by MCBPC or CBPY. For coding of the symbols in the Syntax-based Arithmetic Coding mode refer to Annex E of [41].

DC coefficient for INTRA blocks (INTRADC) is a codeword of 8 bits. The code 0000 0000 is not used. The code 1000 0000 is not used, the reconstruction level of 1024 being coded as 1111 1111.

Transform coefficient (TCOEF) is variable length codeword for texture data. The most commonly occurring EVENTs are coded with the variable length codes. The last bit "s" denotes the sign of the level, "0" for positive and "1" for negative. An EVENT is a combination of a last non-zero coefficient indication (LAST; "0": there are more nonzero coefficients in this block, "1": this is the last nonzero coefficient in this block), the number of successive zeros preceding the coded coefficient (RUN), and the non-zero value of the coded coefficient (LEVEL). The remaining

combinations of (LAST, RUN, LEVEL) are coded with a 22 bit word consisting of 7 bits ESCAPE, 1 bit LAST, 6 bits RUN and 8 bits LEVEL. Use of this 22-bit word for encoding the combinations. For the 8-bit word for LEVEL, the codes 0000 0000 and 1000 0000 are not used.

2.3. Concluding Remarks

We have reviewed a couple of concurrent video rate control schemes. The rate control schemes are mainly composed of buffer-occupancy based rate control and rate distortion model based rate control. Most of current video standards-based techniques are in a form of these two combination. And then we, as an example, have examined H.263 rate control more thoroughly. This was important because H.263 is one of famous representatives in practical very low bit rate video codecs. The focus of explanation was given on the aspect of "variable frame rate" rate control. Finally, the bit stream structure of H.263 was studied as an example in this chapter. The study was intended since bit stream structures of many standards-based codecs look very similar to each other. Some of these ideas will be used in the later chapters.

Chapter 3

Spatio-Temporal Model-Assisted Compatible Coding for Very Low Bit Rate Video Telephony

3.1. Introduction

In practical very low bit rate image transmission systems like H.263, a common characteristic is the use of reduced image sizes (around QCIF) and the decrease of the transmitted frame rate from its original 30 (or 25) to an average of 10 or lower. Such spatial and temporal downsampling is an obvious way to increase compression, enabling compression ratios of 50-100:1 or more. Of course, this unconditional discarding of source information comes at a significant price in terms of the overall quality of the coded pictures. Since there are many applications of very low bit rate image transmission (e.g., in wireless or the Internet), finding improved techniques is an important issue.

In very low bit rate videotelephony situations, state-of-the-art coding algorithms produce artifacts that are systematically present throughout the coded images; all the more as the image content in terms of motion and texture is high. These artifacts usually affect all areas of the image without discrimination. Viewers, however, will generally find coding artifacts to be more noticeable in areas of particular interest

to them. For example, a user of a videotelephony or video teleconferencing system will typically focus his or her attention on the face(s) of the person(s) on the screen, rather than on areas that show clothing or background. Although rapid motion is known to mask coding artifacts, the human visual system has the ability to lock on to and track specific moving objects, such as a person's face. Communication between users of very low bit rate videotelephony or video teleconferencing will be intelligible and pleasing only when facial features are not plagued with an excessive amount of coding artifacts [33].

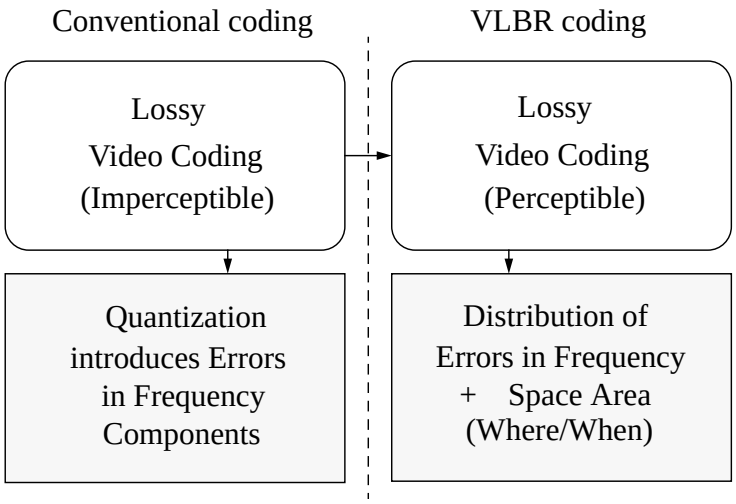


Figure 3-1: Recent trends in video coding technology.

Recent model-based approaches have been receiving a great deal of attention [4, 5, 12, 35] to overcome such artifacts at a very low bit rate. Contrary to conventional coding methods which efficiently represent 2-D waveforms of image signals, model-based coding represents image signals using structural image models which in some sense take into account the 3-D properties of the scene [5]. In a generic model-based coding system, each input video frame to the encoder is analyzed, and a geometric

model of the data is constructed. The model can be either two- or three-dimensional, and can be obtained either through fitting or segmentation. The parameters of the model are transmitted on the channel along with an appropriately coded error signal. The latter is necessary in order to mitigate quality loss in regions of the image where the model does not give a sufficiently good fit [33].

On the other hand, other recent work presented a way to integrate techniques from computer vision to low bit rate coding systems for video telephony applications in the form of *model-assisted coding* (MAC) [33, 31, 32]. The focus was to locate and track the faces and selected facial features of persons in typical head-and-shoulders video sequences, and to exploit the location information in a "classical" video coding system. The motivation was to enable the system to selectively encode various image areas and to produce perceptually pleasing coded images where faces are sharper. Since the approach only affects the bit allocation performed at the encoder, no change is needed in the decoder. Consequently, the technique is applicable to wide range of coding techniques (including H.261 and H.263, as well as MPEG-1 and MPEG-2), with full compatibility with existing decoders [40, 41]. It also has excellent fall-back modes, when the model assumptions do not hold.

In recent years, and given the desire for ever decreasing coding rates, video coding paradigms have shifted towards visibly lossy coding as shown in Figure 3-1. This is a direct result of the inability of current techniques to operate with perceptually undetectable distortion at the desired bit rates. In the conventional paradigm, a coding system attempts to eliminate information to which the human visual systems is not sensitive. The MAC approach [33] pushes the paradigm one step further: if we must introduce visible distortion for high compression, it is more pleasing to introduce the distortion in the background area for minimizing visible artifacts to human perception. That is, the important question in MAC-style coding is where,

when, and how we can introduce "smart distortion" to save more bits.

In this chapter, we propose a technique which achieves spatially selective encoding *as well as* temporally selective one. Previous model-assisted coding focused only in the spatial dimensions, and was successful because high resolution eye contact is very important in person-to-person communication. In this chapter we also use the fact that the temporal aspects of the content have a significant effect on quality. For example, lip synchronization is very important in person-to-person communication. Lip synchronization here refers to the perceived rate of motion of the lips with respect to the speech signal; a low rate will not match consonants with a closed mouth configuration. In practical very low bit rate video transmission systems, there are two possibilities for lip synchronization loss. One possibility is from the fact that the coded frame rate is dropped from 30 to less than 10 on the average. The other possibility is from the fact that, as shown in Figure 3-2, some frames are discarded based on the occupancy of the transmission buffer at the encoder. For either case in which frames are skipped, the decoder duplicates the non-transmitted images from the previous frame. Lip synchronization is thus degraded in the duplicated frames. This degradation is much more serious when a high error rate is introduced by the communication channels, as in wireless networks.

In order not to distort lip synchronization from these possibilities and to achieve improved eye rendering, we assign different budgets of bits or frame rates to areas in different spatial and/or temporal locations. This corresponds to different areas of a sequence having different spatial resolutions and frame rates. We refer to this approach as *spatio-temporal model-assisted* coding (STMAC), to differentiate it from traditional MAC coding described in [33]. The core of the chapter is a spatio-temporally selective rate control scheme, that allows the practical implementation of selective bit allocation in both the spatial and temporal dimensions. The scheme

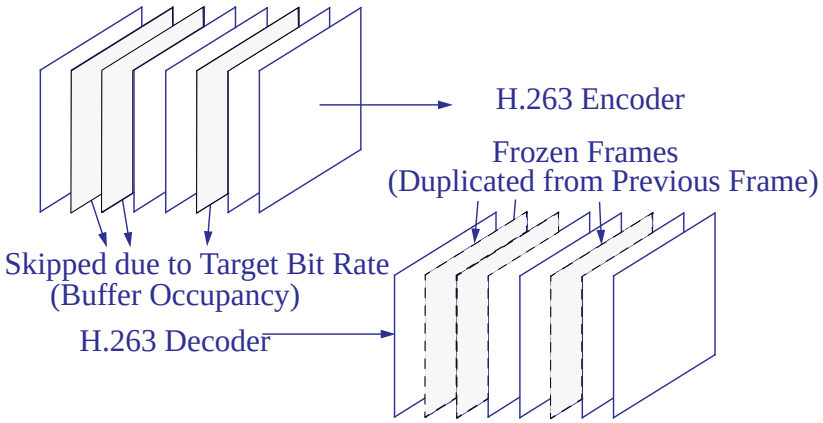


Figure 3-2: Image transmission on a very low bit rate channel.

is based in the use of spatial and spatio-temporal "balance equations" that allow us to effectively redistribute the bit budget in space and time. For the cases when the model assumptions do not hold, we also present an effective solution that uses traditional rate control as a fall-back mode. Some of our initial results with STMAC have been reported in [33, 50, 51].

As with MAC, the proposed coding technique does not require any modification on the decoder, and hence can be used in a fully compatible way. We present experimental results of the application of the STMAC concept to a motion-compensated block-based transform codec, specifically H.263, and also present comparative results with the baseline H.263 using TMN5 rate control. The technique is also applicable to other codecs of the same genre (e.g., MPEG-1 and MPEG-2), but is most appropriate for video telephony and videoconferencing applications at very low bit rates.

The structure of the chapter is as follows. Section 3.2. provides an overview

of STMAC, including its application in a motion-adaptive environment. STMAC consists of three tasks: 1) model area automatic detection and tracking, 2) motion adaptation, and 3) assignment of a proper quantizer and frame rate. We point out that the first and third tasks are highly connected with each other, since motion area detection is a pre-requisite for motion adaptation. Quantizer and frame rate assignment are directly related to the rate control scheme as well, since a target bit rate enforces the rate controller to assign a proper set of quantizer and frame rate. Therefore, we discuss the automatic detection and tracking, and motion detection methods in Section 3.2.1. The area-selective rate control scheme that implements STMAC in a real codec is described in Section 3.3.. Experimental results are presented in Section 3.4. with a summary and conclusions.

3.2. STMAC Technique Overview

The basic motivation in STMAC coding is to use both spatial and temporal *scalabilities* in a single frame simultaneously. We use the term "scalability" in this chapter to mean that different portions of the content are encoded in different resolution/quality and/or frame rate. In other words, scalability is used in the chapter to refer to a content property instead of a bitstream property. We use four different areas, namely the eye, lip, face, and background areas as shown in Figure 3-3. The eye area requires a high resolution but does not need a high temporal frequency. Conversely, medium resolution is sufficient in the lip area, but a high temporal frequency is desired for good lip synchronization. The facial area needs to change at least gradually, to avoid "shearing," and a medium temporal resolution is enough since the face does not move very rapidly. Since the background area does not generally contain essential information and changes infrequently (if at all), it can be degraded as much as possible to save bits for more important regions.

Figure 3-3 depicts the original MAC operation proposed in [33], which only considers spatial scalability. In the STMAC approach, we extend the scalable domain to include the temporal dimension, as shown in Figure 3-9. Note that the shoulder area is included in the background area. STMAC thus combines the features of Figures 3-3 and 3-9, and considers various spatio-temporal scalabilities. The same techniques discussed here can also be used for different applications, where other types of content areas may be important (as long as the areas can be robustly detected).

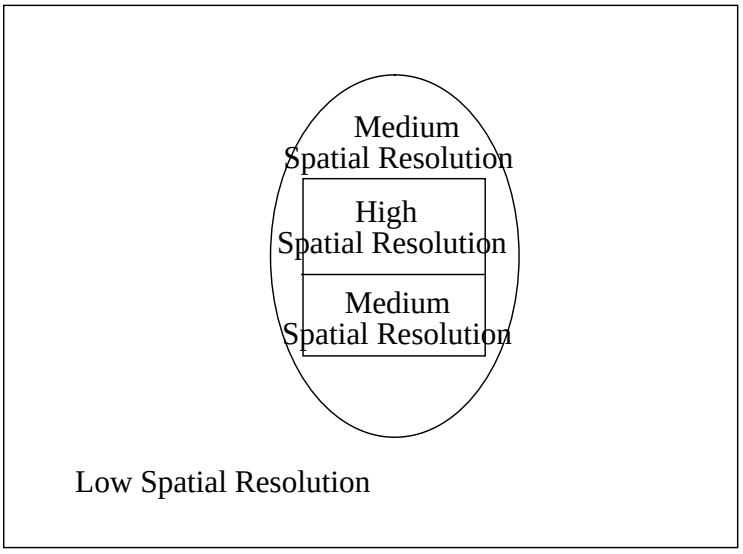


Figure 3-3: MAC coding (spatial scalability only).

3.2.1 Automatic Model Area Detection and Tracking

The first task in order to apply STMAC coding is to discriminate these four areas on a given frame. The detection of head outlines as well as outlines of persons in still or moving images has been the object of active and recent research in image processing and computer vision (see [33, 69] and references therein). In this chapter, for the

purposes of model area detection we follow essentially the same automatic procedure of conventional MAC coding as described in [33, 31, 32]. The task of detecting face locations in a sequence of images is facilitated by both the fact that people's head outlines are consistently roughly elliptical, even when the persons appear in a profile, and by the temporal correlation from frame to frame. Similarly, the task of detecting the eye-nose-mouth region is facilitated by the axial symmetry inherently present in a human face for a person facing the camera, or looking slightly to the side, and appearing in projections in successive frames of a video signal. In this section, we use the same automatic detection and tracking procedure of conventional MAC as described in [33, 32] with slight modifications.

3.2.1.1 Face and Facial Feature Modeling

The model used to represent the location of a face is simply that of an ellipse ε as shown in Figure 3-4, characterized by a center (x_0, y_0), the lengths of its minor and major axes A, B, and a 'tilt' angle θ_0. Although the upper (hair) and lower (chin) areas in actual face outlines can have quite different curvatures, ellipses provide a good trade off between model accuracy and parametric simplicity. Moreover, due to the fact that this information is not actually used to regenerate the face outline (as in a model-based scheme), a small lack of model-fitting accuracy does not have any significant impact in the overall performance of the coding process.

Since an elliptical head outline can in some cases provide only a rough estimate of the face location, the elliptical model is refined by identifying a rectangular region W inside the ellipse, which tightly captures the eyes, nose, and mouth of the person in the scene. This is depicted in Figure 3-5 with an extra degree of freedom which we introduce by allowing a slant of the rectangle's vertical axis. This additional degree of freedom ensures that the detection will be robust in the (very frequent)

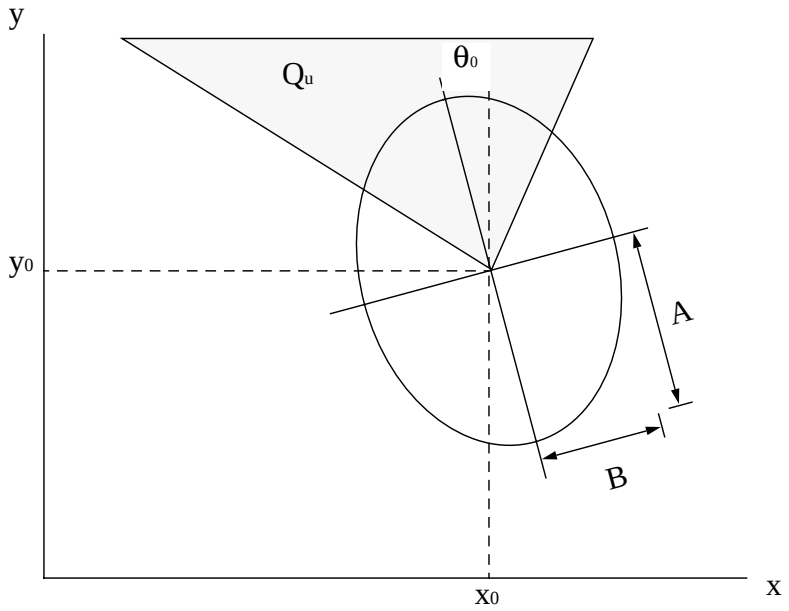


Figure 3-4: Elliptical face location model.

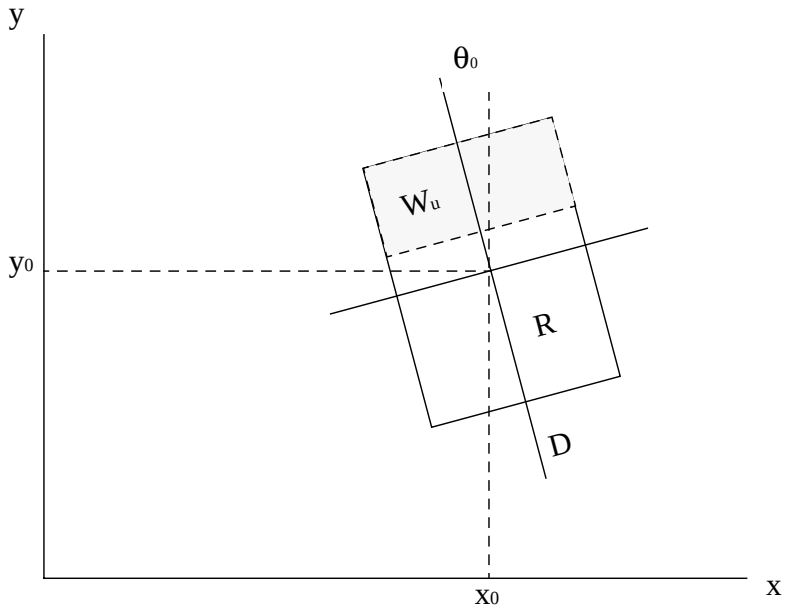


Figure 3-5: Rectangular window as eyes-nose-mouth location model.

case of slight head rotation. The upper third of window, denoted by W_u, is further identified to contain the eyes and eyebrows-two most reliably symmetric features in a human face. The window W is entirely characterized by a center (x_1, y_1), width w, height h, and slant angle θ_1.

3.2.1.2 Model Area Location Detection

The first step in performing area detection is the computation of a downsampled binary edge mask (binary thresholded gradient magnitude) of the input image. The second step is the actual head outline detection on this mask. The algorithm was designed to locate both oval shapes (i.e., 'filled') as well as oval contours partially occluded by data. The final third step consists of selecting the most likely among the potential multiple candidates. This process is described in detail in [33].

Figure 3-6 depicts a thresholded image resulting from the previous algorithms for face outline. In order to improve the face symmetry for the eyes-node-mouth area, a bilevel morphological erosion operation [79] is added to the mask generation process and performed after the ellipse is first detected. Figure 3-7 shows the mask of 3-6 after the operator is applied. The area inside the upper one third W_u of the detected rectangle is considered as eye area, while the remaining part is considered the lip area. The remaining area inside the head outline is considered face area. Figure 3-8 shows the result of automatic detection on an image.

3.2.2 Scalability Templates

The second task is to assign a proper set of quantization parameters (QP) and frame rate (frames-per-second – fps). Note that the quantization step size (QS) is proportional to twice the QP. The various areas need different types of spatio-temporal resolution. Under this constraint we need to apply a rate control mechanism to

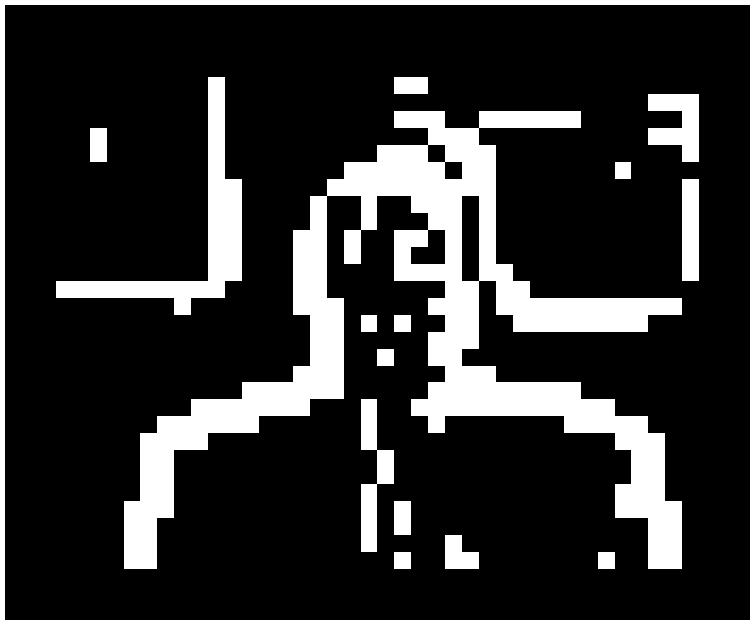


Figure 3-6: Thresholded edge image for face ellipse detection.

Figure 3-7: Bilevel morphological eroded image for face features detection.

Figure 3-8: Automatically detected model areas on a sample image.

meet a target bit rate which affects the QS and fps parameters of each model area. For convenience, we define a "template" as a set of QP and fps corresponding to the each model area. If we take the order of eye, lip, face, and background to be 1, 2, 3, and 4, we can represent the spatial QPs as $S = (S_1, S_2, S_3, S_4)$, and the temporal frequencies as $T = (T_1, T_2, T_3, T_4)$. The detailed assignment algorithm of QP and fps is described in Section 3.3.. As we will see, we do not actually assign absolute QP values, but rather relative importance factors that control how many more bits should be spent in particular area compared with the global average. This decouples the template from the target bit rate, and also helps to integrate templates with the rate control equations employed at an encoder. It is important to note that a template is not used as a "fixed" pattern, but is used as a guide by the rate controller. Given a total number of bits, part of template can be dynamically skipped at certain frames, based on encoder buffer occupancy. As a result, an overall variable frame rate (as in TMN5) rate control algorithm is used in this chapter.

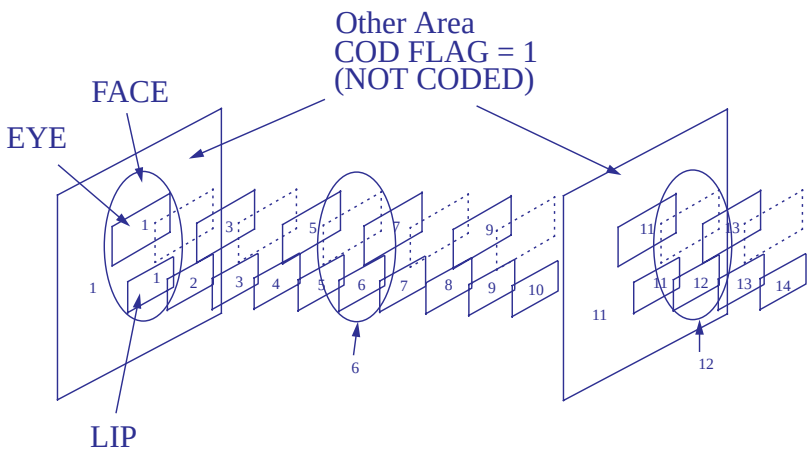


Figure 3-9: Proposed temporally scalable coding for STMAC.

3.2.3 Model Breakdown Prevention

The third task is to accommodate the motion of the model in STMAC coding. The reason for this is that the model's motion cannot be described until the next refresh occurs for each model area. As different areas are refreshed at different times, it is possible in the presence of rapid motion that the update of an area lags behind that of another, distorting the picture. Let's consider the situation where the person we are communicating with moves abruptly. As the shoulder area is coded together with the background, it is possible that the more frequent refresh of the eyes or mouth positions them unnaturally away from the shoulders. The distortion will be eliminated only at the next instance when the background is refreshed. Figure 3-19 (a) shows the effect of extremely rapid model motion (the motion has been artificially generated, and is extremely rapid at 16 pixels per frame). Therefore, avoiding model breakdown is an important issue for practical applications.

To overcome this, we refresh and encode the entire frame when there is signifi-

cant global motion, thus emulating traditional H.263 (frame-based codec). If rapid motion is not present, we use the regular STMAC coding as was described above. The decision of movement is made based on a "facial motion vector," which is calculated by the relative motion of ellipse centers in two consecutive frames. Note that this is similar to conventional conditional replenishment [58].

3.2.4 Codec Syntax Support

A basic functionality required from the codec on which STMAC is applied is to suspend a macroblock's transmission for the selected regions. All standard motion-compensated block-based transform codecs (H.261, H.263, MPEG-1, and MPEG-2) have a "not coded" mode where the decoder just copies the macroblock from the previous reference frame at the exact (with or without motion compensation). In the particular case of H.263, which we use for our experiments, the not-coded mode can be used very easily by simply setting the COD flag to 1 [41]. This is the basic function for selecting different temporal scalabilities in the different image areas. Figure 3-9 also shows the STMAC coding concept and the basic functionality in the H.263 video telephony example.

For the proposed scheme, H.263 poses a particular restrictive syntactic component. In order to increase compression, a differential quantization parameter (DQUANT) is used for signaling the macroblock quantizer selection. As a result, one is forced to use one of the possible DQUANT values of 2, 1, 0, -1, -2, and thus the difference of quantizer step size difference as we cross from one area to the next cannot exceed 2. This can make the application of STMAC on H.263 difficult because the quality difference between the background and foreground areas cannot be significantly different. To avoid this problem, at background refresh time the foreground object is forced to be copied from a previous frame, while the background is

coded with a very low quality QP such as 31. By avoiding coding the background at the same time as foreground areas, we also avoid having to compromise either the number of bits spend on the background, or the quality achieved in foreground areas.

It is important to note that in the first few frames the quality of the coded images is gradually upgraded from a bottom quality such as QP of 31, because there is no previous frame from which to copy the other model areas. In other popular encoders such as H.261, MPEG-1, and MPEG-2, this syntactic restriction does not occur, and one can take full advantage of all quantization step sizes from 1 to 31 for each macroblock.

3.3. Area Selective Rate Control

In order to implement the STMAC approach, one has to modify the rate and frame coding control of an encoder in order to effect area and temporally selective bit allocation. In all our experiments we use the H.263 coding scheme. The baseline rate control for H.263 is represented by TMN5, a test model used for the development and optimization of the standard. TMN5 rate control uses a modified average QP value based on inter and intra frame bit consumption. In addition, it exploits "variable frame rate" rate control, resulting in stable buffer control. As long as there is sufficient buffer space at the encoder, coding proceeds as usual. If the buffer capacity is exceeded, the current and potentially future frames are skipped until the buffer occupancy drops sufficiently [77]. Of course, the encoder should ensure that underflow does not occur (empty buffer) as this would mean that portion of the available bandwidth is not utilized. In very low bit rate video coding systems such as H.263, frame skipping plays an important role, since otherwise such low target bit rates (2 order down from that of MPEG-1) could not be achieved.

The number of quantizers in such a scheme is always one, and hence there is no conceptual difficulty in its application. In STMAC coding, the difficulties for the rate control module arise from the fact that we have a face model with various scalabilities in terms of space and time. For example, we have four different step sizes for the spatial component: eye area, lip area, face area, and background area. In addition, we also have four different frame rates for the temporal component, one for each of the different spatial areas. In this section, we detail a method called *Area Selective* (AS) rate control, which uses a limited number of QS ranges that are associated to the various areas of a given face model.

3.3.1 Balance Equations

We first note that we can use several different spatial and temporal model structures. In particular, we denote such models as: kS-lT, where k is the number of spatially differentiated areas and l is the number of areas with differing frame rates. Quaternary models (in either the spatial or temporal dimension) involve all of the eye, lip, face, and background regions. Ternary models involve the eye, face, and background areas. Binary models include the facial feature area (eyes and lips) and the background only. In this chapter we investigate primarily the models 4S-4T and 4S-2T.

To make rate control appropriate for STMAC, we need to assign a different bit budget to each frame. Let's assume that the human face model can move, but cannot be scaled. We use a fixed frame pattern that spans a 30 frame period (1 sec). We also assign a total bit rate of R bits/sec to this period.

Let's define p_{face}, p_{eye}, p_{lip}, and p_{back} to be the number of frames out of a 30 frame period in which each area is coded. These numbers are fixed when the model template is fixed. In our algorithm, for example, temporal template is assumed to

be $T = (15, 30, 5, 1)$ in "4T" model, whereas T is assumed to be $T = (30, 30, 30, 1)$ in "2T" model.

In addition, let's define A_{face}, A_{eye}, A_{lip}, and A_{back} to be the number of macroblocks for each of the components of the face model. Finally, we denote by β_{face}, β_{eye}, β_{lip}, and β_{back} the expected number of bits/macroblock for a given area of our model.

Based on these definitions, we observe that the following equation must hold:

$$R = \sum_{i=0}^{k-1} p_i A_i \beta_i \tag{3.1}$$

where k is the number of spatially differentiated areas in the kS-lT model. We refer to this equation as the "spatio-temporal balance equation," because it relates bit distribution in spatial and temporal dimensions with the overall bit budget. If we introduce the notation $\gamma_i = \beta_i/\beta$, where β is the average number of bits/macroblock over the 30 frame period, we can rewrite the above equation as:

$$\beta = \frac{R}{\sum_{i=0}^{k-1} p_i A_i \gamma_i} \tag{3.2}$$

This equation uses the γ_i values (supplied by the designer of the codec) to obtain the average number of bits/macroblock for the entire sequence. By using γ_i values higher than 1, we indicate that we wish to have improved spatial quality for the area at hand; similarly, a value smaller than 1 indicates that the area will have lower spatial quality.

After obtaining β, we can now examine how the bit budget should be allocated on a per-frame basis for the 30 frame period. The bit allocation should take into account which parts of the model are present (coded) in every frame. We use the indicator function $1_i(j)$ to determine if frame j is to contain information from area

i. The indicator function depends on the particular frame rate template used (the values of the p_i parameters). With these definitions, we can then express the bit budget $\overline{B_j}$ for each frame j as:

$$\overline{B_j} = \beta \sum_{i=0}^{k-1} A_i \gamma_i 1_i(j) \tag{3.3}$$

Applying spatially-selective rate control involves the modulation of the target bit budget for each area i by the factor γ_i [33]. For example, we will modulate the bit budget for the eye area by $\gamma_{eye} > 1$, therefore generating more bits inside it.

The spatio-temporal balance equation was derived based on the assumption that the model areas are only subject to translational motion, not scaling. In practice, however, the model area sizes do change thus breaking this assumption. We therefore need a way to allocate the given per-frame bit budget $\overline{B_i}$ in a way that takes into account proportionate area sizes.

First, we define A_* as the total number of encoded macroblocks in a given frame. For example, if the eye and lip areas are encoded in a certain frame, $A_* = A_{eye} + A_{lip}$. We also define a_* as the index of the currently encoded macroblock ($0 \leq a_* < A_*$). We can then write a "spatial balance equation," which constrains the bits spent in each area so that the total is equal to the per-frame bit budget $\overline{B_i}$:

$$\begin{aligned} \overline{B_i} &= (\frac{A_0}{A_*}\overline{B_i}\gamma_0 1_0(i) + \frac{A_1}{A_*}\overline{B_i}\gamma_1 1_1(i) + \cdots + \frac{A_{k-1}}{A_*}\overline{B_i}\gamma_{k-1} 1_{k-1}(i)) \\ &= \frac{\overline{B_i}}{A_*}(A_0\gamma_0 1_0(i) + A_1\gamma_1 1_1(i) + \cdots + A_{k-1}\gamma_{k-1} 1_{k-1}(i)) \end{aligned} \tag{3.4}$$

from which we have

$$A_* = A_0\gamma_0 1_0(i) + A_1\gamma_1 1_1(i) + \cdots + A_{k-1}\gamma_{k-1} 1_{k-1}(i) \tag{3.5}$$

The above equation shows the proportionate distribution of the budgeted bits in the various areas, and implies that at least one of the γ_i values is dependent on the others. Thus, for example, γ_0 should be given by:

$$\gamma_0 = \frac{1}{A_0}\left(A_* - \sum_{j=1}^{k-1} A_j \gamma_j 1_j(i)\right) \tag{3.6}$$

$$= \frac{A_* - \sum_{j=1}^{k-1} A_j \gamma_j 1_j(i)}{A_* - \sum_{j=1}^{k-1} A_j 1_j(i)}$$

If the sizes A_i do not change in the current frame with respect to the original frame where the areas were calculated (typically the first frame of the 30 frame period), then the γ_i values will be identical to the ones originally selected in the spatio-temporal balance equation. If there is a change, however, we will have to select an area which will absorb the redistribution of bits so that the bit budget for the current frame is maintained.

An alternative approach would be to proportionately perturb all the γ_i values to meet the spatial balance equation. Although plausible, it would defeat the purpose of design-time selection of relative qualities for the different areas. In addition, when the model has areas where the perceptual importance is mostly in the temporal rather than the spatial dimension, these areas are excellent candidates for absorbing spatial redistribution changes. In our model, the lip area is such a candidate. We should also point out that area size variations are usually quite small, and hence the γ_i "distortion" is small as well.

In the above discussion, the background was participating in the balance equations with its own γ_i and A_i. In our templates the background is coded only once in every 30 frame period. In addition, we usually code the background with the coarsest quantizer ($QP = 31$) in order to save as many bits as possible for the perceptually significant areas. We can then exclude the background from the balance

equation simply by subtracting the number of bits spent to code it (in the first frame of the period) from the overall bit budget for the entire period. As a result, the spatio-temporal and spatial balance equations will only involve areas that compose the model (eyes, face, lips). This modification also implies that all the p_i values for the various areas of the model must be reduced by 1 (since these areas are not coded when the background is coded).

In summary, and assuming a 4-area model, the following operations are performed in order to obtain the required coding parameters in our proposed STMAC approach.

1. Match the face model to the first frame of a 30 frame period and obtain the area sizes A_i.

2. Code the background with the coarsest quantizer ($QP = 31$).

3. Subtract the number of bits B_{back} used for the background from the bit budget of the 30 frame period R to obtain the remaining bit budget:

$$R_{left} = R - \overline{B_{back}}$$

   Use the remaining bit budget R_{left} and the template parameters in the spatial (γ_i) and temporal (p_i) dimensions to compute β (the average number of bits per macroblock for the entire period) using the spatio-temporal balance equation:

$$\beta = \frac{R_{left}}{p_{face}A_{face}\gamma_{face} + p_{eye}A_{eye}\gamma_{eye} + p_{lip}A_{lip}\gamma_{lip}}$$

   We then assign a bit budget $\overline{B_i}$ to each frame except the first using:

$$\overline{B_i} = \beta(A_{face}\gamma_{face}1_{face}(i) + A_{eye}\gamma_{eye}1_{eye}(i) + A_{lip}\gamma_{lip}1_{lip}(i)) \qquad (3.7)$$

where 1_i is the indicator function for area i.

4. For every frame after the first, we match the face model and obtain the current area sizes A_i for the areas that w ill be coded in this frame. We then use the spatial balance equation to ensure compliance with the frame bit budget, using the lip area as the dependent one:

$$\gamma_{lip} = \frac{A_* - (\gamma_{face} A_{face} 1_{face}(i) + \gamma_{eye} A_{eye} 1_{eye}(i))}{A_* - (A_{face} 1_{face}(i) + A_{eye} 1_{eye}(i))}$$

We usually take $\gamma_{eye} > 1$, $\gamma_{face} = 1$, and $\gamma_{lip} < 1$; i.e., the eye area will have finer quality than others, face areas will have average quality, whereas the lip area will not maintain average quality. After experimentation, the values $\gamma_{eye} = 1.5$, $\gamma_{face} = 1$, and $\gamma_{lip} = 0.8$ seemed to provide the best results and are thus used for all experiments reported here.

3.3.2 Rate Control

The next step is to actually enforce the desired γ_i values in the rate controller of our encoder. As mentioned earlier, we use the rate control mechanism employed in H.263 TMN5 as the basic mechanism since it is well-known and represents a good baseline with which to perform comparisons. We should point out, however, that — in principle — any rate control mechanism can be used (see [33] for a discussion of the general applicability of simple model-assisted coding). TMN5 rate control is additionally attractive due to its simplicity of implementation. Especially when it comes to practical video telephony applications, small delay and computational simplicity should be taken into account.

The idea of rate control is an extension of our previous method of buffer rate modulation in [33]. In this chapter, the bit budget is virtually modulated instead

of the buffer rate. When an encoder scans MBs to assign a quantization parameter, the value of bit budget in the rate control formula is modulated/changed as if the value of the budget were larger or smaller than the actual one. Note that the bit budget is assigned based on the spatio-temporal balance equations discussed in the previous section.

The following equations describe our modified TMN5 rate control algorithm:

$$QP_{new} = \overline{QP_{i-1}}(1 + \frac{\triangle_1 B}{2\overline{B_i}} + S\frac{12\triangle_2 B}{R}) \tag{3.8}$$

with

$$\triangle_1 B = B_{i-1} - \overline{B_{i-1}} \tag{3.9}$$

and

$$\triangle_2 B = (B_{i,a_*} - B_o) - \frac{a_*}{A_*}\overline{B_i}\ \gamma_{\zeta(a_*)} \tag{3.10}$$

where:

$\overline{QP_{i-1}}$ is the mean quantizer parameter for the previous picture;

$\overline{B_i}$ is the target number of bits for picture i;

S is a scaling constant which modifies the sensitivity of the QP_{new} adaptation equation;

R is the target bit rate;

B_{i-1} is the number of bits spent for the previous $(i-1)$ picture;

a_* is the index of the current macroblock;

A_* is the number of macroblocks in a picture;

B_{i,a_*} is the number of bits spent for the i-th picture prior to coding the a_*-th macroblock;

B_o overhead bits spent to code areas outside the model (background); and

$\zeta(a_*)$ is a mapping function that returns the area index given the index of the current

macroblock.

Equation 3.8 is identical to TMN5 when $S = 1$, as is equation 3.9. Equation 3.10 deviates from TMN5 in the exclusion of the overhead bits for the background (B_o) and the presence of the $\gamma_{\zeta(a_*)}$ factor. Note that Equation 3.10 is identical to TMN5 when $B_o = 0$ and $\gamma_{\zeta(a_*)} = 1$. The former is used because we fix the QP for the background area to the largest (coarser) value possible (31), while the latter is used to effect area-selective bit allocation. The factor S is used to investigate the effect of modulating $\Delta_2 B$ by $\gamma_{\zeta(a_*)}$. After experimentation, we observed that the value 5 gives the best results.

The buffer content (for variable frame rate operation) is updated after each complete picture in the following way (as defined in TMN5):

```
buffer_content = buffer_content + B_{i,99};
while (buffer_content > 3 · R/F ) {
            buffer_content = buffer_content- R/F;
            frame_incr++;
}
```

where F is the frame rate of source material. In other words, when the buffer occupancy exceeds an average of three frames, subsequent frames are skipped until the occupancy drops below that threshold. In our case, the target frame rate is always set at 30 (the highest frame rate among all the model areas).

3.3.3 Rate Control with Model Breakdown

It is important to note that Area Selective rate control is identical to TMN5 when $S = 1$, $B_o = 0$, and $\gamma_{\zeta(a_*)} = 1$. A key assumption in the application of Area Selective rate control is that the model is subject to moderate motion; as a result, the presence of unusually rapid motion can break the model assumptions. However, we can deal

with such a situation using a traditional frame-based rate control scheme; this can be accomplished by just setting appropriate values to our parameters.

- $\overline{QP_{i-1}}$ is set to 16 (i.e., not to the average quantizer parameter of the previous picture);
- $\overline{B_i}$ is set to $\overline{B}$, the target number of bits for a picture as for TMN5 (i.e., $R/30$);
- A_* is set to 99.

Frame-based encoding usually produces more bits than STMAC for a given frame, since potentially the entire frame has to be coded. However, the variable frame rate rate control scheme (where frames are dropped in cases of high buffer occupancy) can accommodate a potential over-generation of bits.

Consequently, without rapid motion, we use AS rate control (based on balance equations). With rapid motion, we fall back to regular TMN5 rate control. Therefore, we can regulate the buffer output rate in a stable manner in both cases. Switching between the two is trivial, since it only involves resetting of the AS rate controller's parameter values.

3.4. Experimental Results

The compression efficiency of STMAC depends on the base codec efficiency. For example, if we have an efficient baseline H.263 encoder in which all functionalities are switched on (such as Advanced Prediction mode, PB frame mode, etc.), the STMAC encoder will be correspondingly more efficient. If the specific H.263 encoder at hand is not efficient, then the quality for the specified average bit rate will not be as good. Consequently, of importance is the relative performance compared to the baseline codec. Most interesting are experiments where the relative bit rates

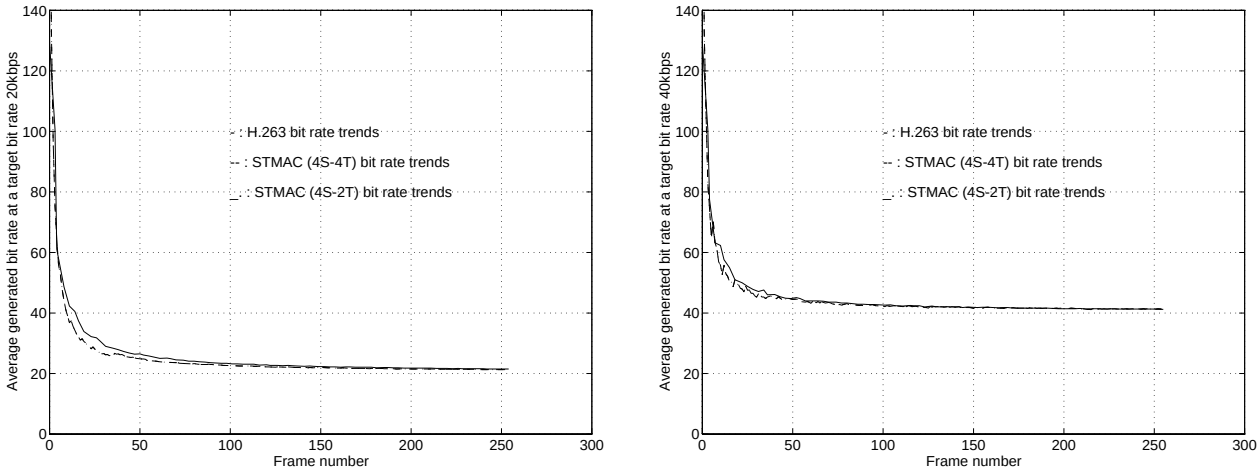
are compared for the same perceptual quality; we especially mention the quality around the face area as a criterion since the fact that human visual system locates and tracks the face elements such as eyes and lips is one of the key motivations. In the following we discuss experimental results both from the point of view of overall rate control effectiveness as well as compression efficiency and visual quality improvement.

3.4.1 Rate Control Performance

Figures 3-12(a) and Figure 3-13(a) show the generated bit rate for the Akiyo sequence for a variety of target rates using the 4S-4T and 4S-2T models, as well the generated bit rate for unmodified H.263. Part (b) of the same figures depicts the buffer evolution for various target bit rates. Note that all contents are measured at 1.66 sec after the start of encoding, giving a relative measurement with the baseline H.263. STMAC rate control converges a little more accurately than that of the baseline H.263, and its average buffer occupancy is lower than that of the baseline H.263. The 4S-2T model has a smoother occupancy level than 4S-4T since the 2T model doesn't change the area. However, in 4T case, the area to be re-encoded changes a lot based on the temporal scalability patterns.

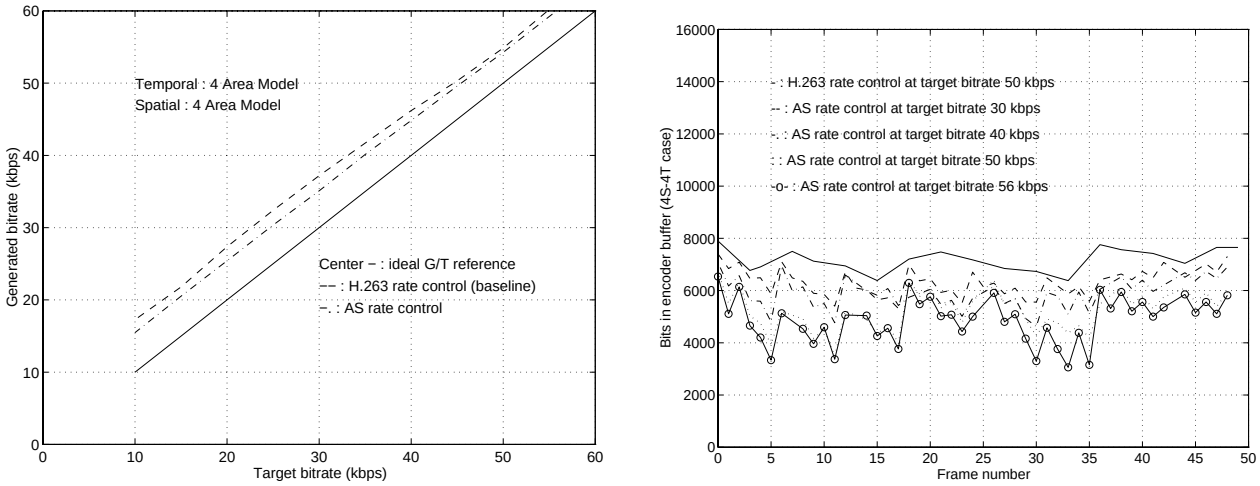
Figure 3-14 (a), (b), (c), and (d) shows the buffer occupancy in the Hypothetical Decoder Buffer (HDB) which is defined by the H.263 specification. From these figures we see clearly that, for videoconferencing scenes, the proposed rate control algorithm performs similarly (in terms of buffer dynamics and target bit rate convergence) to simple H.263.

Figure 3-10: Moving area discontinuity according to various temporal scalabilities (The face appears elongated).



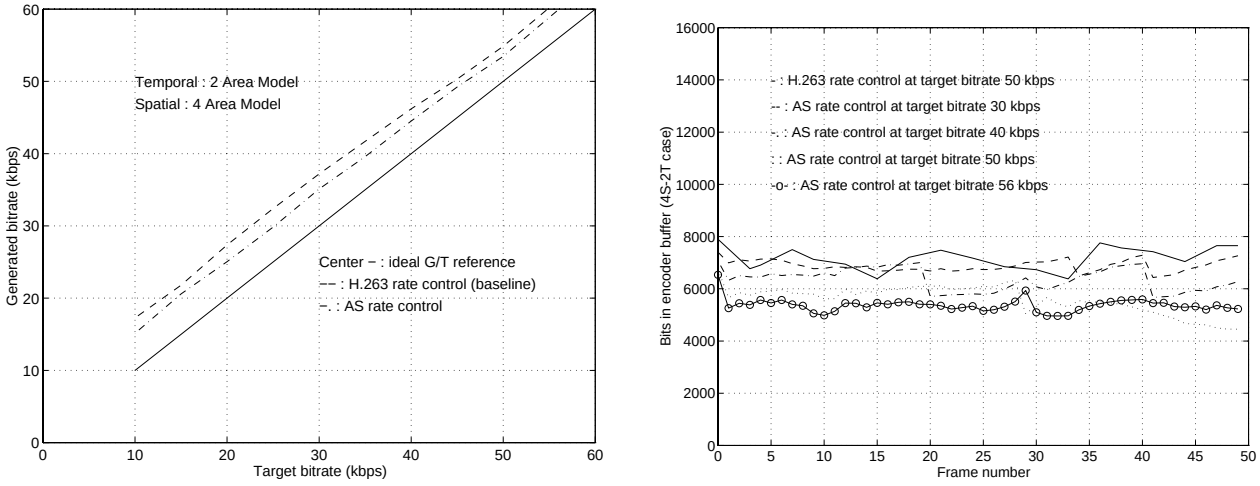
(a) (b)

Figure 3-11: Convergence comparisons between H.263 rate control and AS rate control: (a) At target bit rate 20 kbps, (b) At target bit rate 40 kbps



(a) (b)

Figure 3-12: Rate control comparisons between 4S-4T STMAC and simple H.263 in Akiyo at 1.66 sec.: (a) Real vs. target bit rate, (b) Buffer occupancy



(a) (b)

Figure 3-13: Rate control comparisons between 4S-2T STMAC and simple H.263 in Akiyo at 1.66 sec.: (a) Real vs. target bit rate, (b) Buffer occupancy

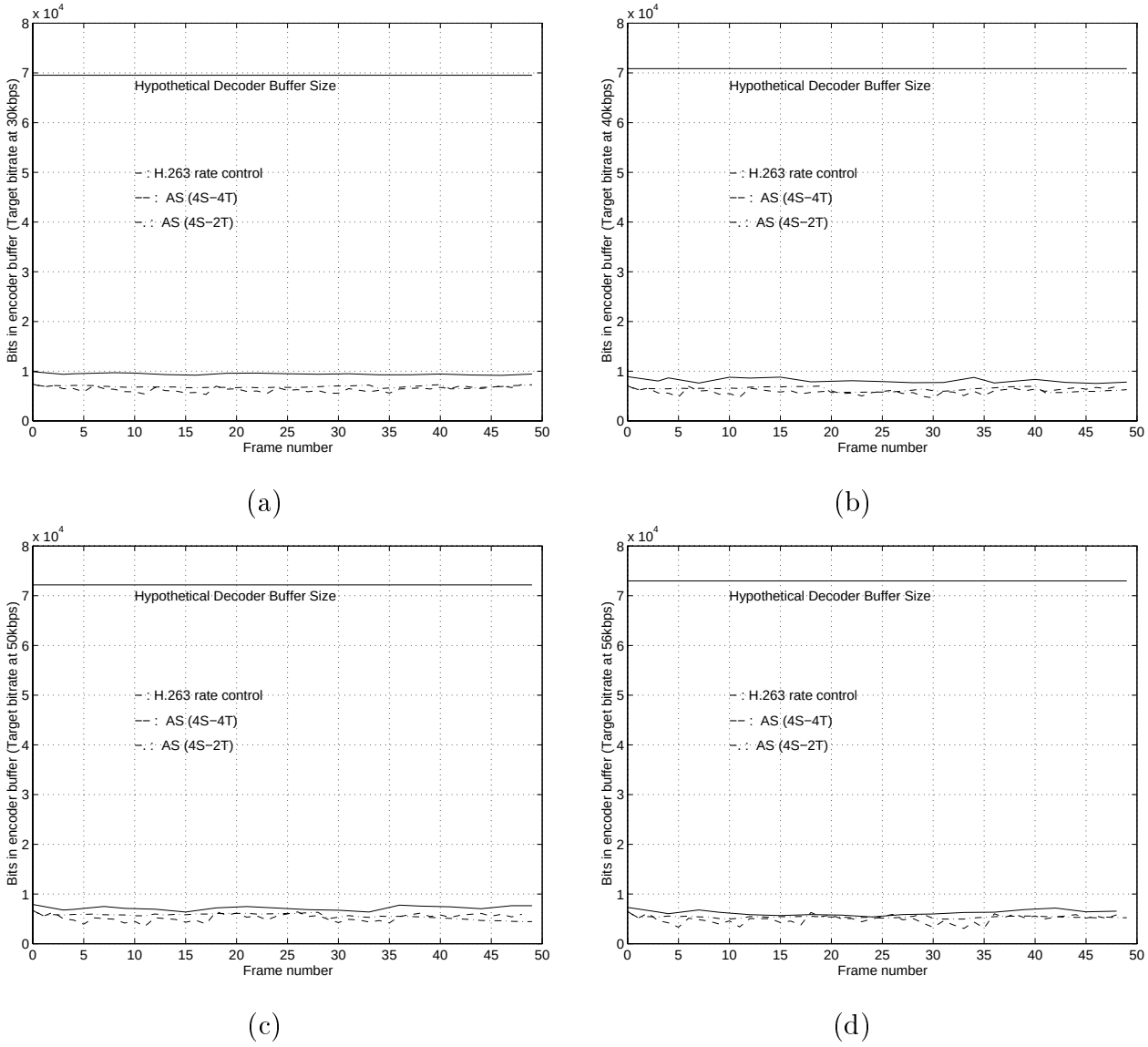


Figure 3-14: Buffer occupancy : (a) Buffer status with comparison of HDB at target bit rate 30 kbps, (b) Buffer status with comparison of HDB at target bit rate 40 kbps, (c) Buffer status with comparison of HDB at target bit rate 50 kbps, (d) Buffer status with comparison of HDB at target bit rate 56 kbps.

Target bit rate (Kbps)	H.263	STMAC 4S-4T (Ratio)	STMAC 4S-2T (Ratio)
30	18	45 (2.5)	48 frames (2.66)
35	18	44 (2.44)	48 frames (2.66)
40	18	44 (2.44)	48 frames (2.66)
45	18	45 (2.5)	48 frames (2.66)
50	18	44 (2.44)	49 frames (2.72)
56	18	45 (2.5)	50 frames (2.77)

Table 3.1: Total number of (variably) encoded frames: comparison for 1.66 sec duration at 50 kbps, Akiyo.

3.4.2 Compression Efficiency and Visual Quality

More interesting are the results in terms of compression efficiency (reduced bit rate for same visual quality) or improved visual quality (for the same bit rate).

Our experiments show that we can have more than 2 times higher temporal resolution video sequences with similar/better spatial quality around the face area using the STMAC encoder rather than a conventional H.263 encoder. Table 3.1 shows the absolute and relative frame counts between STMAC and H.263 at the same target bit rate.

Figures 3-15 and 3-16 show a decoded output frame for Akiyo and Miss America, using various spatio-temporal models including conventional H.263, and at two target bit rates (46.2 kbps and 54.26 kbps). Regular H.263 is shown in (a) and (d); (b) and (e) are 4T area models, whereas (c) and (f) are 2T area models. These figures indicate that the visual quality using STMAC is slightly better or similar around the face area. However, the temporal resolution in STMAC is much higher than H.263. Table 3.2 indicates the number of times each area is encoded in the two models that we use and also H.263.

The temporal scalability plays an important role for achieving high compression ratio in STMAC coding. A macroblock has 3,072 bits in YUV format, if there is no

Model Area	H.263	STMAC (4S-4T)	STMAC (4S-2T)
Eye	18	21	47
Lips	18	42	47
Face	18	7	47
Background	18	2	2

Table 3.2: Number of times an area is encoded (comparison for 1.66 sec duration at 50 kbps, Akiyo).

(a) producing 10 fps (b) producing 26 fps (c) producing 28 fps

(d) producing 10 fps (e) producing 26 fps (f) producing 28 fps

Figure 3-15: Decoded outputs (all from H.263 decoder: 12th frame of Akiyo) : (a) Conventional H.263 at a target bit rate 40 kbps (70.46:1 compression), (b) STMAC (4S-4T) at a target bit rate 40 kbps (179.09:1 compression), (c) STMAC (4S-2T) at a target bit rate 40 kbps (196.76:1 compression), (d) Conventional H.263 at a target bit rate 50 kbps (64.84:1 compression), (e) STMAC (4S-4T) at a target bit rate 50 kbps (147.96:1 compression), (f) STMAC (4S-2T) at a target bit rate 50 kbps (167.27:1 compression).

(a) producing 10 fps

(b) producing 26 fps

(c) producing 28 fps

(d) producing 10 fps

(e) producing 26 fps

(f) producing 28 fps

Figure 3-16: Decoded outputs (all from H.263 decoder: 22th frame of Miss America): (a) Conventional H.263 at a target bit rate 40 kbps (73.92:1 compression), (b) STMAC (4S-4T) at a target bit rate 40 kbps (134.32:1 compression), (c) STMAC (4S-2T) at a target bit rate 40 kbps (180.27:1 compression), (d) Conventional H.263 at a target bit rate 50 kbps (58.45:1 compression), (e) STMAC (4S-4T) at a target bit rate 50 kbps (115.43:1 compression), (f) STMAC (4S-2T) at a target bit rate 50 kbps (147.27:1 compression).

compression involved. Because in practical block-based transform coding systems the average compression ratio is about 50-100 to 1, a macroblock has around 30-60 bits on the average. A big part is the overhead due to representation in a bitstream. However, in STMAC the background area is just copied from previous frame so that just 1 bit is needed for each macroblock (the H.263 COD flag). In addition, the portion of the head area is less than 50 percent of a frame in typical sequences. This allows us to compress the whole background into just a small number of bytes, since each background macroblock inside of it needs just 1 bit. For example, if we consider a QCIF size image which has 99 macroblocks in a frame, we will assign less than 50 bits (6-7 bytes) for all background macroblocks of a frame in STMAC, thus making the compression ratio increase dramatically.

For example, in Figure 3-15 (b) and (e) we achieved 179.09:1 and 147.96:1 compression ratios for STMAC coding, while in (a) and (d) simple H.263 gives 70.46:1 and 64.84:1. Similarly, in Figure 3-15 (d) and (f) we achieved 196.76:1 and 167.27:1 compression ratios. Hence compression ratios in both 4S-4T and 4S-2T are very high as expected. The overall quality around the face is similar to that of baseline H.263. The eye area quality looks improved, while the lip area appears slightly degraded (spatially). The background area shows substantial blocking effects. As mentioned in the introduction, in order to achieve such very low bit rates artifacts are unavoidable. Our objective is to preserve as much as possible of the perceptually important part of the content, even if other areas have to be degraded further. Note that (a) is actually a different frame from the others (frame 10 which is repeated as frame 12); this is because H.263 skipped the frame due to a high buffer occupancy. Figure 3-16 shows a comparison of H.263 and STMAC using the Miss America sequence. This sequence has the characteristic that the face area occupies a much larger portion of the frame than the Akiyo sequence. As a result, the re-encoded

area is larger and the compression ratios are slightly lower, although they are still quite high in both 4S-2T and 4S-4T.

Figures 3-17 (4S-4T model) and 3-18 (4S-2T model) present the core idea of the proposed STMAC coding system. Figure 3-17 (a) shows PSNR results for the eye/lips/face areas. Figure 3-17 (b) shows PSNR results for entire frames. The eye/lips/face PSNRs in (a) are higher than that of the frame PSNR, as expected. Another interesting fact lies in Figure 3-17 (c) where we can see that the four different areas use different frame rates, but human perception will consider as the virtual effective frame rate the rapidest frame rate used (lips). Figure 3-18 shows the same results for the 4S-2T model. Note that 4S-4T has achieved a slightly higher PSNR than that of 4S-2T since total re-encoded area is smaller. It may, however, suffer more from potential moving area discontinuity problems (as discussed in Section 3.4.3).

We also provide simulation results comparing the STMAC scheme with conventional H.263 compression under extreme situations of ultra-rapid motion. We created an artificially moving Akiyo sequence using foreground and background Video Object Planes (VOPs) obtained from the Akiyo MPEG-4 test sequence. The foreground object is shifted 16 pixels each of frame with respect to the static background. This obviously violates completely the assumption of slow model motion which is inherent in STMAC coding, but it is a useful exercise for examining the limitations of the base STMAC technique as well as the benefits of using model breakdown prevention. Coding results are shown in Figure 3-19. As we can see, the selective encoding of the various areas results in parts of the image that contain remnants of a previous coding pass. In model breakdown prevention case (shown in (b)), motion tracking preempts the spatio-temporal and spatial balance equations and forces replenishment of the entire frame such as H.263. The drawback, of course,

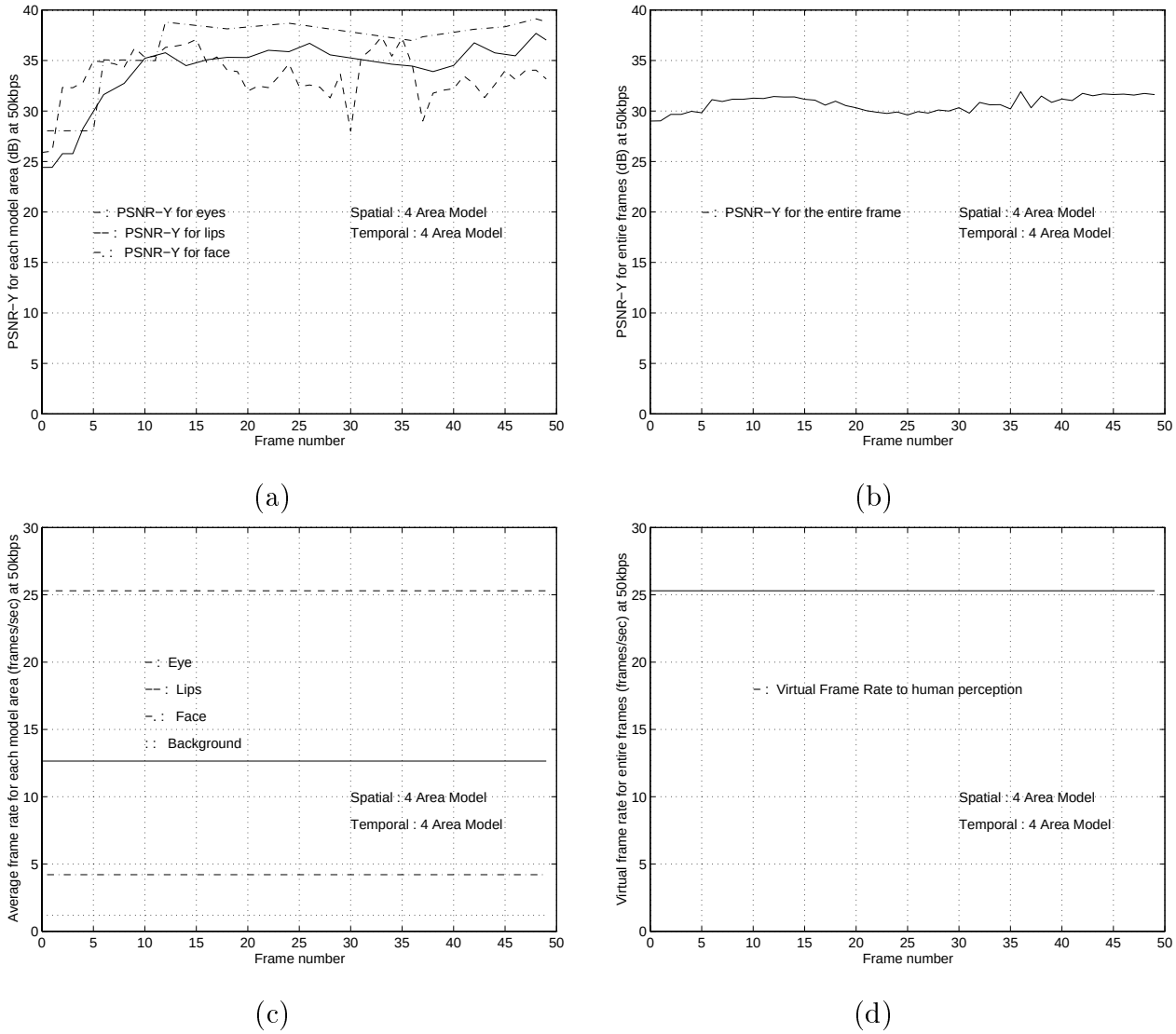
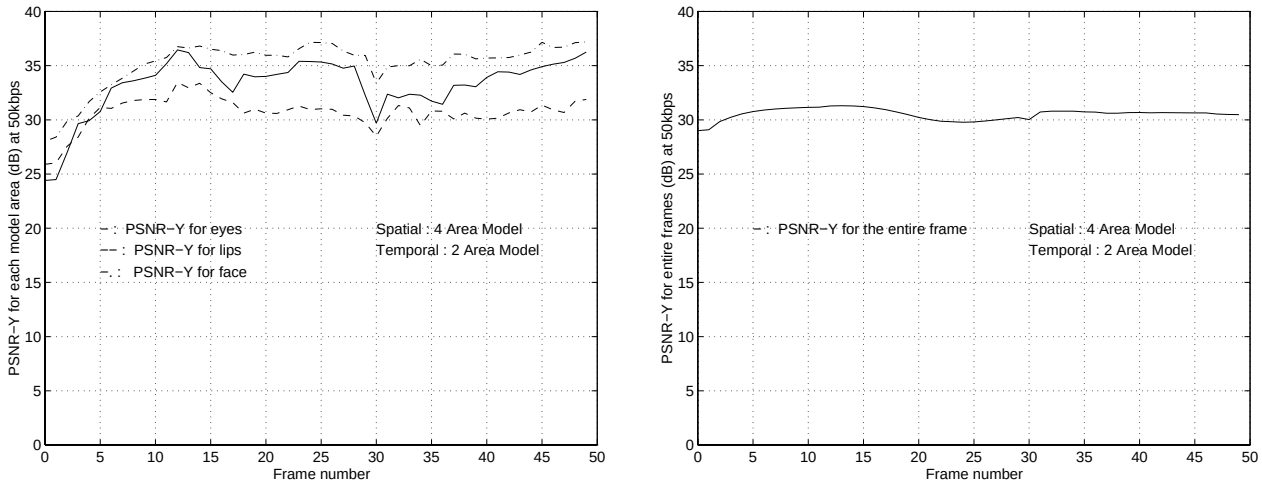


Figure 3-17: PSNR and frame rate (4S-4T, Akiyo) : (a) Eye/lip/face area PSNR (dB) with STMAC at 50 kbps, (b) Entire frame PSNR (dB) with STMAC at 50 kbps, (c) Average frame rate (fps) for model areas with STMAC at 50 kbps, (d) Average frame rate (fps) for an entire frame with STMAC at 50 kbps.



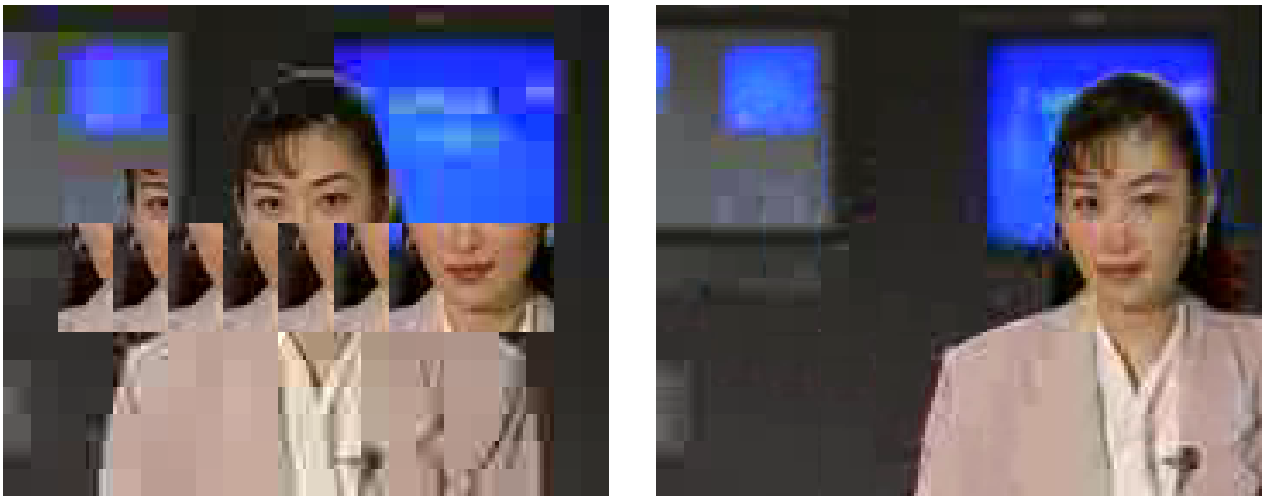
(a) (b)

Figure 3-18: PSNR (4S-2T, Akiyo) : (a) Eye/lip/face area PSNR (dB) with STMAC at 50 kbps, (b) Entire frame PSNR (dB) with STMAC at 50 kbps.

is that the areas of interest are no longer rendered with high quality (notice, for example, the eye region in these two frames) because a smaller bit budget is given to that specific frame (i.e., $R/30$). This example is artificial and such high motion is very unlikely to happen in typical videoconferencing situations. Fast motion is likely when a subject first enters or leaves the scene, and it would last for a very brief period of time (less than 1 sec).

3.4.3 Model Selection Considerations

Our experiments have shown that the spatio-temporal model-assisted coding approach can potentially create discontinuous motion defects, as discussed in Sections 3.2.3. This is attributed to the various temporal scalabilities of the model. It is not desirable to have extremely different temporal scalabilities in adjacent areas (e.g., 1 fps vs. 30), regardless of their QP difference in the spatial dimension. This is especially important for the face area, including the eyes and lips, since it is the focus of human observers.



(a) (b)

Figure 3-19: Decoded output: (a) The encoding defect in STMAC for an artificial sequence with extremely rapid motion, (b) An example scene of motion-adaptive STMAC

In Figure 3-10, for example, the face appears elongated compared with the actual scene. This is due to the fact that the lip area refresh is being done at every frame, while the eye area refresh is at 6 fps in this example. The eye area is therefore not being tracked on time. If, however, we use a two-area model (in which face has finer QP and more frequent frame refresh rate and background has coarser QP and less frequent frame refresh rate) there are no discontinuity artifacts. But this is not the best policy for improving compression and/or quality. In contrast, if we use the four area structure we can save the maximum number of bits (or improve quality), but sometimes experience "moving area discontinuity." Hence the number of areas and the overall structure of the model play an important role in achieving a reasonable balance among the various trade-offs[1]. In our experiments, we restrict ourselves on the 4S-4T and 4S-2T models that can be generated by our automatic face detection algorithm.

[1]Model selection is, of course, also dependent on what areas can we robustly detect in real-time.

3.5. Concluding Remarks

STMAC addresses the need to perform better bit allocation for very low bit rate coding (less than 64 Kbps). The motivation is that artifacts are systematically present at such rates; the overall user experience, however, can be significantly improved if we take into account human perception characteristics. Of particular importance are the eye region which must be rendered very finely and the lip region which must be replenished very frequently.

The first component of STMAC is a simple face model that can be used to robustly identify, in real-time, the location of important face components (face, eyes, mouth). Accuracy is not extremely important, as the model is only used as a guide to bit allocation, not rendering at the receiver. Equipped with this model, we can formulate a general rate control architecture by considering two balance equations: one operating in both space and time, and one operating only in space. These equations are driven by the available total bit rate, and the desired emphasis that the encoder's designer wishes to place in the various model areas (both spatially and temporally). The equations provide specific bit budgets for each frame of a given period (e.g., 30 frames). These budgets are then used in an area-selective adaptation of the basic rate control algorithm (here TMN5).

Our experiments demonstrate that indeed the approach provides improved results compared with baseline H.263 TMN5 encoding. The perceptually important areas are improved, whereas the temporal resolution is much higher. This substantial improvement comes at the cost of a more degraded background, and the possibility of observing motion discontinuities caused by the different temporal scales used in the various parts of the model. The model breakdown prevention, which forces STMAC back to the traditional frame-based H.263, ensures that gross inaccuracies are avoided.

One of the key benefits of the approach is that it can be applied without any modification to the decoder, and is hence fully compatible with H.263. This is also important when model breakdown occurs, i.e., the scene content does not contain a head-and-shoulder sequence and hence the model detection algorithm fails. The encoder can always fall back to standard H.263 encoding thus achieving acceptable performance.

We should also point out that the application of the technique is not limited to H.263, but can be applied to any block-based transform coding scheme. The syntactic features of the scheme are, however, important, as they can affect both the mechanism and the overhead associated with establishing different frame rates for the various model areas. H.263 is attractive in this respect because of its very low cost macroblock repetition option (COD flag), but it is also somewhat restrictive because of the differential encoding of QP values that forces absolute differences between successive macroblocks to be up to 2.

Chapter 4

Optimal Buffered Compression and Coding Mode Selection for MPEG-4 Shape Coding

4.1. Introduction

MPEG-4, a project of the International Standardization Organization (ISO), is an emerging coding standard that supports new ways for communication, access and manipulation of digital audio-visual data. MPEG-4 will offer a flexible framework and an open set of tools supporting a range of both novel and conventional functionalities. The aim is a generic audio-visual coding system with acceptable consumer quality, markedly better than possible existing standards and products actually available [77].

The video part of the MPEG-4 specification provides two core components that are not available in any other standard. The first component is object-based representation, as opposed to a pixel or frame based representation, that allows scene composition and interactivity with content. The representation of objects consists of Video Objects (VO), and Video Object Planes (VOP). The VOs correspond to entities in the bitstream that the user can access and manipulate (e.g., cut and paste), are independently coded, and saved on separate bitstreams. Instances of a

VO in a given time are called VOPs. The second component is a certain degree of flexibility in the design of a system that leads to an open, yet efficient, standard. The notion of a toolbox is used to allow a flexible design of coding algorithms based on the requirements of specific applications [30].

The MPEG-4 video encoder is composed of two main parts: shape coding, and the traditional texture coding for the same VOP. The texture coding as well as motion estimation parts of the encoder are similar in principle to those used in other state-of-the-art standards. In the past, the problem of shape representation and coding has been thoroughly investigated in the fields of computer vision, image understanding, image compression and computer graphics. However, this is the first time that a standardization effort is adopting a shape representation for coding purposes [30].

There are two types of shape data in MPEG-4: grey scale and binary shape information. The grey scale shape information has a similar structure to that of binary shape with the difference that every pixel can take on a range of values (usually 0 to 255) representing the degree of transparency of that pixel. Binary shape information corresponds to grey shape information with values of 0 and 255.

To compress both grey and binary shape data, we fundamentally use the same technique: context-based arithmetic encoding (CAE) as defined by MPEG-4. The only difference in handling data between grey scale and binary encodings is that the grey scale shape data needs an additional process for texture data compression (i.e., grey scale) on top of binary data compression. That is, we need to divide the grey scale shape data into binary shape (i.e., support) and texture data, thereby applying a texture compression algorithm on the texture data of the grey shape. Since the texture coding part is not our main interest in this chapter, we concentrate on binary shape compression.

Lossy shape coding techniques were reported in several recent papers [47, 80]. The reports mainly described polygon/spline representation approaches that provide optimality in the operational rate-distortion sense. MPEG-4's CAE shape coder is, on the other hand, a binary bitmap-based shape coder [47]. In this chapter, a framework of optimality in the operational rate-distortion sense is provided based on "optimal buffered compression" for binary bitmap-based shape data.

Optimal buffered compression was recently proposed in [60] to provide optimal buffer control strategies for video sequences using a finite buffer environment. The authors formalized the description of the buffer-constrained adaptive quantization problem. They then formulated the optimal solution for a given set of admissible quantizers used to code a discrete nonstationary signal sequence in a finite buffer. In the MPEG-4 video context, optimal buffered compression can be thought of as the optimal solution for texture coding. The importance of MPEG-4 as an industry standard with extensive future use in interactive multimedia systems suggests further investigation on the optimal buffered compression issue on shape information as well.

This chapter addresses the shape counterpart of optimal compression under buffer constraints. We first formulate the buffer-constrained adaptive quantization problem for shape coding. We then propose algorithms for optimal and fast, but sub-optimal, solutions. In addition, a low bit rate tuned algorithm is proposed for very low bit rate applications such as wireless and/or Internet video.

The structure of the chapter is as follows. In Section 4.2., the optimal buffered compression problem is described. Section 4.3. explains the background for MPEG-4 shape coding, and provides a quantitative problem formulation; the optimal algorithm is identified and shown to be quite complex. A fast approximation algorithm and a low bit rate tuned algorithm are presented respectively in Sections 4.4.

and 4.5., and simulation results follow in Section 4.6.. A discussion and concluding remarks are given in Section 4.7..

4.2. Optimal Buffered Compression with Mode Selection

4.2.1 Motivation and Related Work

One of the findings through the core experiment process of MPEG-4 is that the portion of shape coded bits over the entire video is less than 20 percent (especially at target bit rates greater than 75 kbps). Therefore, lossless shape coding is a reasonable candidate for dealing with shape data in MPEG-4. However, there are still cases where lossy shape coding is desirable. For example, an HDTV signal has larger resolution as well as higher quality than those of regular video. In this case, some distortion around the shape boundary is not that harmful to human perception since the block dimension of coding unit is quite small within a HDTV resolution. Therefore, dealing with lossy shape coding for high bit rate video may be meaningful under certain conditions. When a low bit rate video, on the other hand, is considered such as in wireless and/or Internet video, the absolute number of bits of shape data should be reduced as much as possible due to its bandwidth requirement [84]. Therefore, dealing with lossy shape coding for low bit rate video may be useful for certain applications unless there are unacceptable error patterns on block boundaries [84]. In addition, if an intelligent rate control method were not devised for MPEG-4 shape coding, some part of bit stream such as the one used for conversion ratio (CR) would be wasted.

Recently, lossy shape coding techniques were reported in a couple of papers [47, 80]. Several polygon/spline representation approaches that provide optimality in operational rate-distortion sense were proposed. They approximated the boundary by a polygon/spline and considered the problem of finding the polygon/spline which

leads to the smallest distortion for a given a number of bits. The authors also addressed the dual problem of finding the polygon/spline which leads to the smallest bit rate for a given distortion. The papers presented fast and efficient methods for a couple of different measures including maximum operator and summation operator for shape data.

MPEG-4's CAE shape coder is, on the other hand, a binary bitmap-based shape coder [47]. More recently, a shape coding control algorithm was devised to reduce the amount of bits used for shape data under low bit-rate conditions [84] for CAE shape coders. To control the number of bits for shape information, the authors proposed to vary the value of "threshold" based on the current mode of operation. Once, therefore, the threshold is chosen by the algorithm heuristically, the shape coding is executed. This heuristic technique has been adopted by MPEG committee in July 1997 as part of the video Verification Model (VM8).

In this chapter, a framework of optimality in operational rate-distortion sense is proposed based on "optimal buffered compression" for binary bitmap-based shape data. The idea of optimal shape coding is that a virtual quantization parameter and a coding mode are determined based on the current buffer occupancy instead of threshold. In addition, a careful consideration for small size images such as QCIF is given to prevent coded images from being unacceptable to human perception. That is because a CR=$\frac{1}{4}$ makes coded image irritating to human observers for QCIF size, as was reported in [61] through the context of MPEG-4 CAE. Thus, a low bit rate tuned algorithm is introduced as well.

4.2.2 Problem Definition

Recently, optimal trellis-based buffered compression was proposed in [60] to provide optimal buffer control strategies for video sequences in a finite buffer environment.

Figure 4-1 depicts the traditional optimal buffered compression approach. Originally, optimal buffered compression was defined to select only the optimal quantizer. In this chapter, the optimal buffered compression is expanded to select optimal quantizer and coding mode as follows.

Problem Definition: Given a set of quantizers, a sequence of blocks to be quantized, and a finite buffer, select the optimal quantizer and coding mode for each block so that the total cost measure is minimized and the finite buffer never overflows.

Consider the allocation for N blocks, and suppose there are total M combinations of quantizers and coding modes available to code each block. For example, the number of combinations, M, is 6 when the number of quantizers is 3 (say, QP=1, 2, and 3) and the number of coding modes is 2 (say, intra_mode and inter_mode). Note that the "quantizer index" from 1 to M for such a combination element can be in an arbitrary order. For example, a quantizer index set {1, 2, 3, 4, 5, 6} can be corresponding to a set {(2, intra_mode), (3, inter_mode), (3, intra_mode), (1, intra_mode), (2, inter_mode), (1, inter_mode)}. Let d_{ij} and r_{ij} be, respectively, the distortion and the number of bits produced by the coding of block i with quantizer index j, and let r be the channel rate in bits per block. Define an admissible solution x as a selection of one quantizer index for each block, i.e., a mapping from $\{1, 2, ..., N\}$ to $\{1, 2, ..., M\}$, $x = \{x(1), x(2), ..., x(N)\}$, where each $x(i)$ is the quantizer index for block i. Therefore, $(r_{1x(1)}, ..., r_{Nx(N)})$ and $(d_{1x(1)}, ..., d_{Nx(N)})$ are, respectively, the rate and distortion for each block and a given choice of a quantizer index mapping x.

Now, define the buffer occupancy at stage i, B_i for a given admissible solution x. The buffer occupancy cannot be negative at any stage (i.e., underflow means the buffer occupancy is 0). Let $B_1 = r_{1x(1)} + B_0, B_2 = max(B_1 + r_{2x(2)} - r, 0)$ and, in

general

$$B_i = max(B_{i-1} + r_{ix(i)} - r, 0) \tag{4.1}$$

where the buffer occupancy at each block instant is increased by the coding rate of the current block and decreased by the channel rate. B_0 is the initial buffer state.

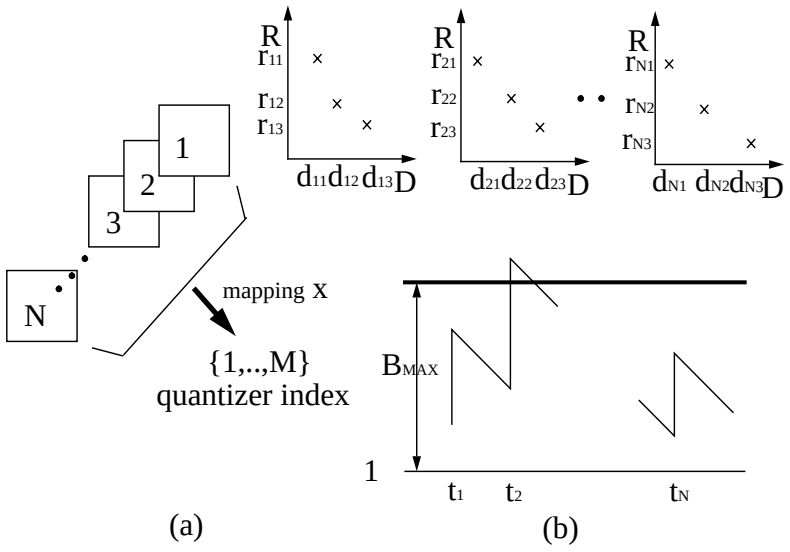


Figure 4-1: (a) Each block in the sequence has a different R-D characteristic (for a given choice of quantizers for the blocks in the sequence, we can obtain R-D points to form the composite characteristic), (b) R at t_2 is not a feasible solution with the chosen buffer size (buffer is limited).

General Formulation : (Integer Programming)

The problem is to find the mapping x that solves

$$\min_{x(i), i=1,...,N} (\sum_{i=1}^{N} d_{ix(i)}), \tag{4.2}$$

subject to

$$B_i \leq B_{max}, \forall i = 1, ..., N$$

where B_{max} is the buffer size.

In the MPEG-4 video context, previous optimal buffered compression can be thought of as the optimal solution for texture coding. The following sections address the shape counterpart of optimal buffered compression within an extended context of optimal coding mode selection.

4.3. Optimal Rate Control for MPEG-4 Shape Coding

4.3.1 MPEG-4 Shape Coding Overview

A binary alpha plane can be encoded in INTRA mode for I-VOPs and in INTER mode for P-VOPs and B-VOPs. The principal method used is block-based CAE with block-based motion compensation. For a detailed explanation of MPEG-4 shape coding, we refer to the MPEG-4 Video Specification [2] and Verification Model [1]. In this section, we concentrate on the size conversion process which is considered as a virtual quantizer in the context of optimal buffered compression.

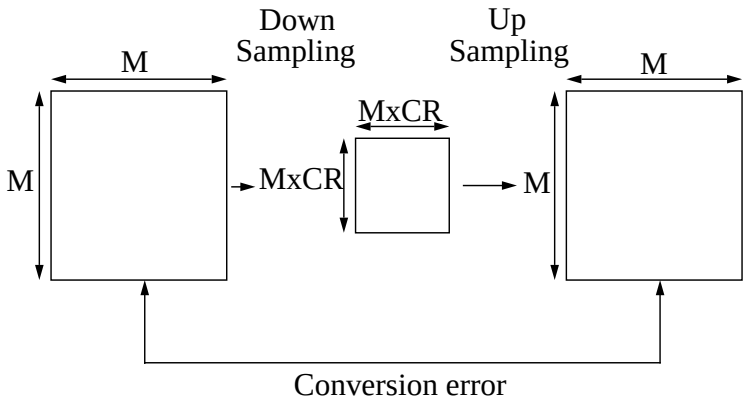


Figure 4-2: Size conversion.

Current rate control and rate reduction in MPEG-4 is realized through size

conversion of the binary alpha information as shown in Figure 4-2. In this process, the determination of the conversion ratio (CR) is done based on a given distortion threshold *alpha_th*. That is, it is necessary to ascertain whether a Binary Alpha Block (BAB) has an acceptable quality under the size conversion process with a specific CR. Here, BAB is defined as a set of 16×16 binary pixels as illustrated in Figure 4-3. The criterion is based on a 4×4 pixel block (PB) data structure. Each BAB is composed of 16 PBs.

Given the current original BAB and some approximation of it (i.e., BAB'), we define an "acceptable quality" function ACQ as follows.

$$ACQ(BAB') \stackrel{def}{=} MIN(acq_1, acq_2, ..., acq_{16}) \tag{4.3}$$

where,

$$acq_i = \begin{cases} 0, & \text{if } SAD\_PB_i > 16 \times alpha\_th \\ 1, & \text{otherwise} \end{cases}$$

and $SAD\_PB_i(BAB, BAB')$ is defined as the sum of absolute differences for PB_i, where an opaque pixel has the value 255 and a transparent pixel has the value 0. The parameter *alpha_th* has values {0, 16, 32, 64,..., 256}. If $alpha\_th = 0$, then encoding will be lossless. A value of $alpha\_th = 256$ means that the accepted distortion is maximal (i.e., in theory, all alpha pixels could be encoded with an incorrect value).

For the size conversion process, let us consider the down sampling case first. For CR=1/2, if the average pixel value in a 2×2 pixels block is equal to or greater than 128, the pixel value of the down sampled block is set to 255, otherwise to 0. For CR=1/4, if the average pixel value in a 4×4 pixel block is equal to or greater than 128, the pixel value of the down-sampled block is set to 255, otherwise to 0.

Second, we consider the up-sampling case. When CR is different from 1, up-

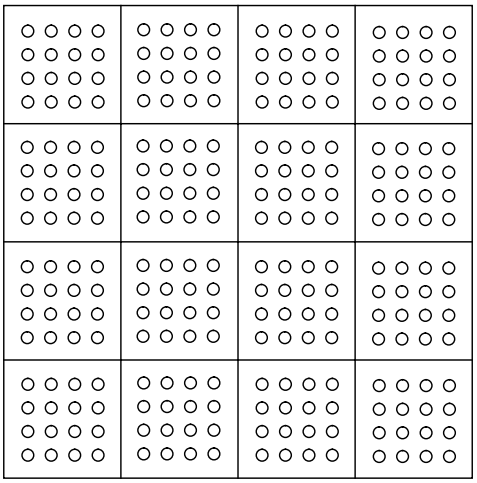


Figure 4-3: A BAB consists of 16 PBs.

sampling is carried out for the BAB. The value of the interpolated pixel (let P1 be the top-left point, P2 top-right, P3 bottom-left, and P4 bottom-right) is determined by calculating the filter context of neighboring pixels. For the pixel value calculation, the value of "0" is used for a transparent pixel, and "1" for an opaque pixel.

The values are given by:

P1: if$(4 \times A + 2 \times (B + C + D) + (E + F + G + H + I + J + K + L) > Th[C_f])$ then '1'

else '0'.

P2: if$(4 \times B + 2 \times (A + C + D) + (E + F + G + H + I + J + K + L) > Th[C_f])$ then '1'

else '0'.

P3: if$(4 \times C + 2 \times (B + A + D) + (E + F + G + H + I + J + K + L) > Th[C_f])$ then '1'

else '0'.

P4: if$(4 \times D + 2 \times (B + C + A) + (E + F + G + H + I + J + K + L) > Th[C_f])$ then '1'

else '0'.

Here, the 8-bit filter context, C_f, is calculated as follows:

$$C_f = \sum_k c_k \cdot 2^k \tag{4.4}$$

The positions of A, B, C, D, E, F, G, H, I, J, K, L and the c_k are defined and depicted for each P1, P2, P3 and P4 in Figure 4-4. Based on the calculated C_f, the threshold value ($Th[C_f]$) can be obtained from a look-up table given in the MPEG-4 Verification Model document [1]. Note that after interpolation the pixels in the low-resolution image (A- L) are not contained in the pixels of the upsampled image (i.e., all pixels in the upsampled image are interpolated). When the BAB is on the left (and/or top) border of the VOP, the left (and/or top) borders are extended from the outermost pixels inside the BAB. In the case where CR=1/4, the BAB is interpolated into the size of CR=1/2, and then interpolated again into CR=1.

The value assigned to CR for a specific BAB is based on $ACQ(BAB')$. If this quality is acceptable, that CR is adopted for that BAB.

Once CR is determined, now size conversion is done with that CR value for the BAB. After size conversion, each BAB is coded according to one of seven different modes, all lossless, as listed below.

1. MVDs==0 && No Update
2. MVDs!=0 && No Update
3. all_0

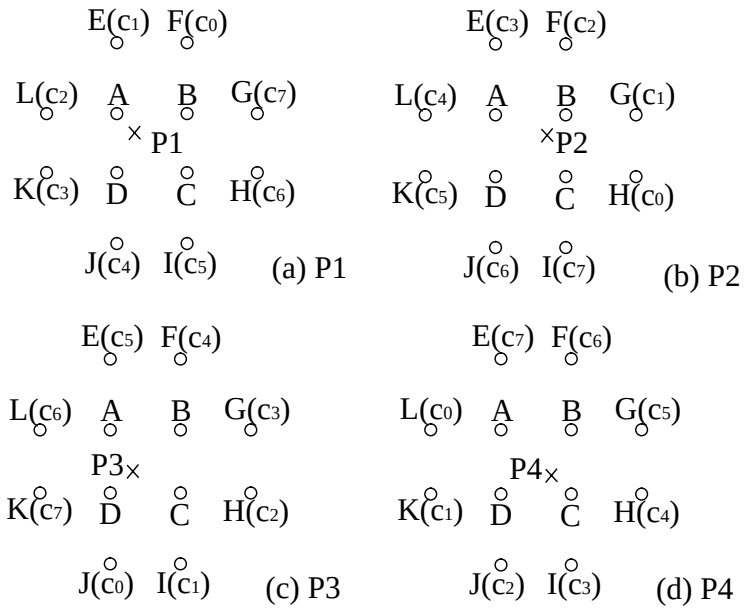


Figure 4-4: Interpolation filter and interpolation construction.

4. `all_255`

5. `intraCAE`

6. `MVDs==0 && interCAE`

7. `MVDs!=0 && interCAE`

Here, MVs and MVPs are defined as a shape motion vector and a shape motion vector predictor, respectively. MVDs stands for motion vector difference of shape, and is determined by MVs and MVPs (i.e., MVDs=MVs-MVPs). In I-VOPs, only the coding modes, `all_0`, `all_255` and `intraCAE` are allowed. MVPs is determined by candidates of MVs and texture motion vectors (MV) around the macroblock corresponding to the current shape block. They are located and denoted as shown in Figure 4-5 where MV1, MV2 and MV3 are rounded to integer values. By looking into MVs1, MVs2, MVs3, MV1, MV2 and MV3 in this order, MVPs is determined

by taking the first encountered MV that is valid (not zero). If no candidate MV is valid, MVPs is regarded as 0.

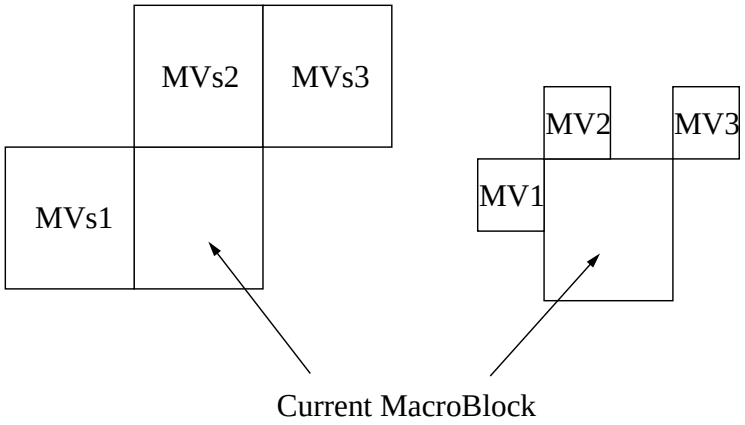


Figure 4-5: Candidates for MVPs.

Based on MVPs determined above, MVs is computed by the following procedure: the MC error is computed by comparing the predicted BAB (by MVPs) and the current BAB. If the computed MC error is less or equal to $16 \times AlphaTH$, where $AlphaTH$ is a threshold used when comparing two 4×4 sub-blocks, the MVPs is directly employed as MVs, and the procedure terminates. If the above condition is not satisfied, MV is searched around the prediction vector MVPs. The search range is +/- 16 pixels around MVPs along both the horizontal and vertical directions. The MV that minimizes the sum of absolute differences (SADs) is taken as MVs and this is further interpreted as MVDs for shape.

We refer, once again, to the MPEG-4 Verification Model for a detailed explanation of the BAB coding mode decision algorithm.

4.3.2 Optimal Buffered Compression for MPEG-4 Shape Coding

Since the MPEG-4 standard only specifies the decoder (in fact, the syntax of a compliant bitstream), modules in encoders such as rate control can be arbitrarily designed. In this section, we describe an optimal rate control algorithm, which monitors the occupancy of the encoder buffer.

In a basic MPEG-4 shape coding system, the determination of CR value and BAB coding mode is based on $alpha\_th$ and a specific mode decision algorithm as designed in the Verification Model document. The idea of optimal shape coding, in this chapter, is that the CR value and BAB coding mode are determined based on current encoder buffer occupancy with a consideration of R-D characteristics.

Note that in the rate control algorithm in the current Verification Model, only the quality measure is taken into account; there are no buffer constraints. Therefore, a problem occurs when the encoding buffer is considered (i.e., in constant bit rate applications). To introduce a proper rate control framework, we apply the optimal buffered compression concept to MPEG-4 shape coding by considering the size conversion process as a "virtual quantizer."

In the size conversion process, distortion can possibly be introduced in two levels; VOP-level size conversion and the following BAB-level size conversion. If this is thought of as a "virtual quantizer," we can apply optimal buffered compression concepts to shape coding. Note that in this formulation the $alpha\_th$ value is replaced by (VOP(BAB(i)).CR, BAB(i).CR ,coding_mode), thus eliminating $alpha\_th$. In the above expressions, $BAB(i)$ means i-th BAB, and $VOP(BAB(i))$ indicates the VOP in which $BAB(i)$ is located. Therefore, the problem formulation can be given as follows.

Shape Coding Formulation : (Integer Programming)

Let x be $\{x(1), x(2), ..., x(N)\}$ where $x(i)$ = (VOP(BAB(i)).CR, BAB(i).CR ,

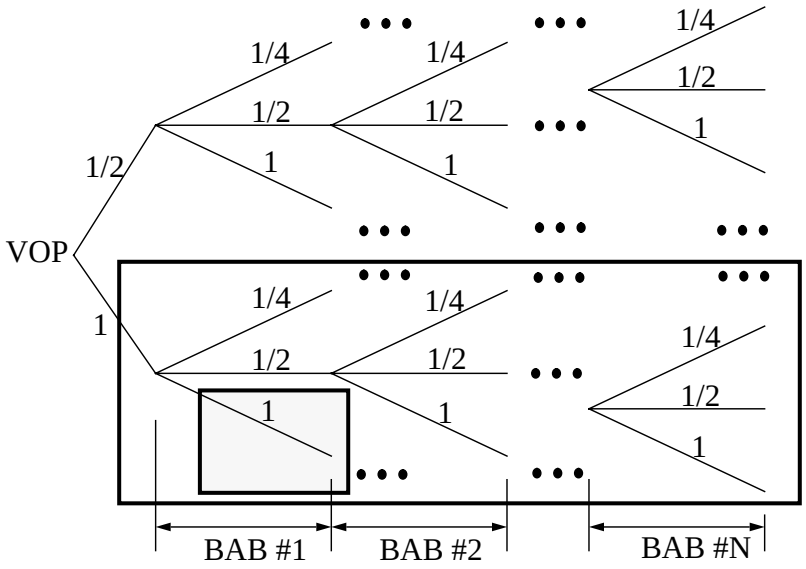


Figure 4-6: One path determines a quality and rate of output shape data (Branch value means VOP.CR and BAB.CR).

coding_mode).

The problem is to find the mapping x that solves

$$\min_{x(i),i=1,...,N} (\sum_{i=1}^{N} d_{ix(i)}), \tag{4.5}$$

subject to

$$B_i \leq B_{max}, \forall i = 1, ..., N$$

where B_{max} is the maximum buffer size. In the above expressions, $BAB(i)$ means i-th BAB, and $VOP(BAB(i))$ means VOP in which $BAB(i)$ is.

Figure 4-6 shows all the possible paths of encoding CRs from the first BAB to the last BAB. A certain path in the figure implies a quality and rate of a specific output shape data sequence. Each branch has two values involved: VOP.CR and BAB.CR. In this chapter, we only consider VOP.CR=1. The problem of optimal shape coding is to decide a certain path which gives minimum overall distortion under B_{max} as

defined in Equation (4.5). Note that in this problem we try to minimize the added distortion (i.e., global minimum). If we consider that the distortion is independent and additive with respect to each macroblock, we can use dynamic programming to sequentially eliminate suboptimal solutions. We can grow a tree where each stage represents one block and where the different states represent a possible cumulative bit use up to that stage. Then we can rule out solutions for which there is an alternative providing less total distortion for the same rate. If two paths converge to the same node in the tree (i.e., the two solutions use the same number of bits for blocks considered), then we need only keep the minimum cost path [60, 59].

4.4. Fast Approximation Algorithm

For the theoretically optimal solution, the past buffer size and data should be preserved while the coding procedure is operating. However, this is impractical, since the algorithm makes the coding delay extremely long. In addition, to consider all the modes of BAB with various values of CR requires a considerable amount of computation. To make the algorithm more feasible and make computational demands realistic, we propose a greedy "marginal buffer reservation" algorithm as a fast approximation algorithm.

4.4.1 Greedy Algorithm

The aspects of fast approximation algorithms about optimal buffered compression point in two directions: *Lagrange multiplier* method and *greedy* method as shown in recent studies [60, 83, 83]. The greedy method was reported to achieve a nearly optimal performance with much relaxed burden of computational complexity in [83]. The proposed greedy method in [83] gives much less computational complexity than the Lagrange multiplier method in [60].

We take greedy method approach in this chapter based on following two reasons: firstly, the greedy method shows to achieve a very close performance to optimal algorithm as shown in [59, 60, 83, 72, 71]; therefore, excessive computation is not necessary to improve a very small PSNR increase for most of applications. Secondly, strict optimality is not the most important purpose in video compression, but "human perceptual factors."

To take greedy method means that we choose a CR which gives minimum distortion at each macroblock stage. This decision rule will make the additive objective function not only a local minimum but also a nearly global minimum. For example, if there is no buffer limitation involved, the path which has CR=1 at every macroblock (the lowest path in the figure) is the optimal path. In reality, each branch path is chosen based on the occupancy of the encoder buffer. If a branch is not allowed due to current buffer occupancy, other branches which generate fewer bits are considered. The decision is made only by the buffer status at the current BAB. It is important to note that the distortion measure is independent even when the INTER mode is selected: the quality of a BAB with INTER mode is not affected from the best match BAB in the previous frame. The best matched area is used for updating the probability table to apply CAE coding to efficiently encode BABs in a lossless manner.

The optimal path can thus be approximated by the following rule: at each macroblock, a CR which generates smallest distortion is chosen unless the buffer is overflowed. If there are more than one branches which produce the same distortion, the CR which produces fewer bits is chosen. Note that recent texture part for "optimal buffered compression," which is proposed in [60], was computational intensive since the number of quantizers is usually not small (i.e., the QP value ranges from 1 to 31). However, the shape part in this chapter is quite feasible because the number

of quantizers is just 3 (i.e., 1, 1/2, and 1/4).

Once CR is chosen at that specific macroblock, we further need to decide what "lossless" coding mode must be used for that BAB. To ensure bitstream compatibility in the MPEG-4 context, we should use (some or all of) the same 7 BAB coding modes (`MVDs==0 && No Update`, `MVDs!=0 && No Update`, `all_0`, `all_255`, `intraCAE`, `MVDs==0 && interCAE`, `MVDs!=0 && interCAE`).

Let's discuss the decision algorithm in detail in order to make the bitstream compliant with the current MPEG-4 specification. First `all_0` and `all_255` modes are considered. Note that these modes generate smaller bits than "`No Update`" mode when all data in a BAB are 255 or 0. That is why we consider "`all_0`" and "`all_255`" first before `No Update mode`. The `all_0` and `all_255` modes are only considered at CR = 1. "`No Update`" mode is used when MC_BAB is completely matched to BAB. "`intraCAE`" is carried out after previous modes are tried. In addition, "`interCAE`" is carried out from the second frame after previous modes are tried. Basically, the quality is independent on BAB modes, because the encoding is always lossless. That is, the quality distortion comes only from the size conversion process. Since all BAB modes give us the same quality in spite of producing different amount of bits, the best policy for us in order to choose the BAB mode is to select the one which gives us the smallest number of bits as long as the buffer is not overflowed.

Since there is no distortion in CR=1 case (i.e., the best quality), the encoding mode which gives the smallest bits should be chosen among BAB coding modes at CR=1. The only concern may be given when a certain encoding mode at CR=$\frac{1}{2}$ or CR=$\frac{1}{4}$ also produces no distortion. Since the size is smaller at CR=$\frac{1}{2}$ or CR=$\frac{1}{4}$, the generated bits by the CAE are definitely smaller than that of CR=1.

In CR=$\frac{1}{2}$ case, distortion is usually introduced. Therefore, the encoding mode

which makes the lowest distortion should be a candidate among BAB coding modes at CR=$\frac{1}{2}$ as long as the buffer is not overflowed. When there are more than one which have the same distortion, a mode which generates the smallest bits is chosen. The concern, once again, should be given when CR=$\frac{1}{4}$ produces a smaller distortion than that of CR=$\frac{1}{2}$.

The same consideration should be also applied to CR=$\frac{1}{4}$ case. In CR=$\frac{1}{4}$ case, distortion usually maximally happens. Therefore, the encoding mode which makes the lowest distortion should be considered as a candidate among BAB coding modes at CR=$\frac{1}{4}$ as long as the buffer is not overflowed. When there are more than one which have the same distortion, a mode which generates the smallest bits is chosen.

The final decision must be made, considering all level of CR values, based on the optimal path decision rule mentioned earlier. Consequently, the optimal branch decision for each BAB is made according to the following pseudo-code.

```
if(ALL0(BAB)) {
            all_0 mode;
}
else if(ALL255(BAB)) {
            all_255 mode;
}
else if(MC_BAB == BAB) {
            No Update mode;
}
else {
```

$$mode_1 = MODE_{min}(D_{intraCAE}\texttt{(CR=1)}, D_{interCAE}\texttt{(CR=1)},$$
$$D^T_{intraCAE}\texttt{(CR=1)}, D^T_{interCAE}\texttt{(CR=1)});$$
$$mode_2 = MODE_{min}(D_{intraCAE}\texttt{(CR=}\tfrac{1}{2}\texttt{)}, D_{interCAE}\texttt{(CR=}\tfrac{1}{2}\texttt{)},$$

$D^T_{intraCAE}$(CR=$\frac{1}{2}$) ,$D^T_{interCAE}$(CR=$\frac{1}{2}$));

$mode_3 = MODE_{min}$($D_{intraCAE}$(CR=$\frac{1}{4}$),$D_{interCAE}$(CR=$\frac{1}{4}$),

$D^T_{intraCAE}$(CR=$\frac{1}{4}$) ,$D^T_{interCAE}$(CR=$\frac{1}{4}$));

if(D_{mode_3}(CR=$\frac{1}{4}$) is the smallest distortion) {

Use $mode_3$;

}

else if(D_{mode_2}(CR=$\frac{1}{2}$) is the smallest distortion) {

Use $mode_2$;

}

else if(D_{mode_1}(CR=1) is the smallest distortion) {

Use $mode_1$;

}

else {

Backward mode;

}

}

Here, $MODE_{min}$ is the choice function for a "encoding mode" among its candidates. The encoding mode means a pair of CR and CAE methods. For example, (CR=1, `intraCAE`) is an encoding mode. The $MODE_{min}$ is defined to select an encoding mode which comes with the minimum distortion or buffer size parameter, and not to return any encoding mode when all parameters are infinite values. In addition, $MODE_{min}$ is defined to select the mode which generates the smallest number of bits (i.e., the lowest buffer size just after that mode is applied) when more than one distortion parameters are of the same value. For example, in above $mode_1$, $MODE_{min}$ function selects (CR=1, `intraCAE`) as $mode_1$ if `intraCAE` at

CR=1 gives us the lowest buffer size among input parameters, as long as the buffer is not overflowed. That is possible since distortion is all the same at various encoding modes with fixed CR values. For algorithm description, we also define the following terms. $B_{intraCAE}$(CR=1) and $B_{interCAE}$(CR=1) mean the current buffer sizes just after `intraCAE(CR=1)` and `interCAE(CR=1)` are applied respectively. Similar meanings can be defined for the expression of $B_{intraCAE}$(CR=$\frac{1}{2}$), $B_{interCAE}$(CR=$\frac{1}{2}$), $B_{intraCAE}$(CR=$\frac{1}{4}$), and $B_{interCAE}$(CR=$\frac{1}{4}$). And $B^T_{intraCAE}$(CR=1), $B^T_{interCAE}$(CR=1), $B^T_{intraCAE}$(CR=$\frac{1}{2}$), $B^T_{interCAE}$(CR=$\frac{1}{2}$), $B^T_{intraCAE}$(CR=$\frac{1}{4}$), and $B^T_{interCAE}$(CR=$\frac{1}{4}$) are all the same expressions for the buffer sizes of CAE-ed "transposed data." The above expressions are used in the next section. Note that if the selected coding condition involves transposition of the BAB, then "ST" flag in the bitstream is set "1." Otherwise, it is set to "0." On the other hand, $D_{(encode)}$(CR=cr) means the distortion when (encode) method is used for the compression at (CR) size conversion. We define the value of D as infinite when there is no encoding mode which makes the current buffer occupancy under the maximum buffer size. $D_{intraCAE}$(CR=1), $D_{interCAE}$(CR=1), $D_{intraCAE}$(CR=$\frac{1}{2}$), $D_{interCAE}$(CR=$\frac{1}{2}$), $D_{intraCAE}$(CR=$\frac{1}{4}$), and $D_{interCAE}$(CR=$\frac{1}{4}$) are defined as distortion just after the BAB are CAE-ed as such. And $D^T_{intraCAE}$(CR=1), $D^T_{interCAE}$(CR=1), $D^T_{intraCAE}$(CR=$\frac{1}{2}$), $D^T_{interCAE}$(CR=$\frac{1}{2}$), $D^T_{intraCAE}$(CR=$\frac{1}{4}$), and $D^T_{interCAE}$(CR=$\frac{1}{4}$) are all distortion for just after the transposed data are CAE-ed.

The final step in BAB mode decision is to compare the shortest INTRA code with shortest INTER code. For this comparison, it is necessary to add the number of bits for MVDs to the INTER code length. Note that, once again, the same CR produces the same distortion. Under the same distortion, the shortest code length is the best.

Note that in interCAE mode motion vectors are necessary. However, they cannot

be obtained based on the technique described in the MPEG-4 Verification Model document since the notion of optimal buffered compression doesn't allow us to use *alpha_th* (i.e., motion vectors in interCAE mode in the current MPEG-4 Verification Model should be determined based on the *alpha_th* threshold, as we mentioned in the previous section). In order to make the notion of proposed algorithm compatible with the current VM bitstream, we propose to set $alpha\_th = 0$ which means that the mode where the search range of MV is +/- 16 pixels around MVPs is always "turned on." Note that +/- 16 pixels area is quite large. Therefore, we expect that in most cases the motion vectors of shape coding are close to the motion vectors of texture coding.

If any of these values at a certain macroblock stage cannot avoid buffer overflow, the MB of the previous step should be reconsidered in the branch for the same computation in order to find a feasible optimal path. In theory, the buffer size and past data should be kept until the entire coding procedure ends, because there is overflow possibility in any future macroblocks. This basically means that the coding delay is extremely large. We, therefore, further consider a fast approximation algorithm to make the algorithm more practical in the next section.

4.4.2 An Assumption on CR and Distortion

To reduce the computational complexity more, we make the assumption that a reconstructed (i.e., down and up size conversion) image from CR=$\frac{1}{2}$ has smaller distortion than that from CR=$\frac{1}{4}$. We also assume that this is true between the original image and the reconstructed image from CR=$\frac{1}{2}$. Based on this assumption, we can search, based on the current buffer size, in the following order: (1,1), (1,1/2), (1,1/4). That is, we consider the higher quality image first. Therefore, once we meet such an image which gives non-overflow buffer condition, we don't need to go further;

the algorithm for CR ends here. In reality, this assumption is almost always true. If this assumption is used, the algorithm will not require distortion computation as will be shown in following sections. This nice property comes from the fact that at the same CR value the distortion is the same whatever BAB coding mode is selected.

4.4.3 An Observation in Finite Memory Environment

The backward unit of the greedy algorithm in the previous section is another computationally intensive part. In addition, we cannot allow infinite memory for the optimal path computation. Therefore, to restrict the number of backward steps it is important to devise a simpler algorithm. The greedy algorithm can be modified to exploit a finite number of backward steps when the current buffer cannot be lower than B_{max} with any of CR values. The CR value and encoding mode decision for each BAB is made according to the following pseudo-code.

```
if(ALL0(BAB)) {
            all_0 mode;
}
else if(ALL255(BAB)) {
            all_255 mode;
}
else if(MC_BAB == BAB) {
            No Update mode;
}
```

else if(Min{$B_{intraCAE}$(CR=1), $B_{interCAE}$(CR=1), $B^T_{intraCAE}$(CR=1), $B^T_{interCAE}$(CR=1)} < B_{max}) {

Use $MODE_{min}$($B_{intraCAE}$(CR=1), $B_{interCAE}$(CR=1),

$B^T_{intraCAE}$(CR=1),$B^T_{interCAE}$(CR=1));

}

else if(Min{$B_{intraCAE}$(CR=$\frac{1}{2}$),$B_{interCAE}$(CR=$\frac{1}{2}$),$B^T_{intraCAE}$(CR=$\frac{1}{2}$),$B^T_{interCAE}$(CR=$\frac{1}{2}$)}
< B_{max}) {

Use $MODE_{min}$($B_{intraCAE}$(CR=$\frac{1}{2}$),$B_{interCAE}$(CR=$\frac{1}{2}$),
$B^T_{intraCAE}$(CR=$\frac{1}{2}$),$B^T_{interCAE}$(CR=$\frac{1}{2}$));

}

else if(Min{$B_{intraCAE}$(CR=$\frac{1}{4}$),$B_{interCAE}$(CR=$\frac{1}{4}$),$B^T_{intraCAE}$(CR=$\frac{1}{4}$),$B^T_{interCAE}$(CR=$\frac{1}{4}$)}
< B_{max}) {

Use $MODE_{min}$($B_{intraCAE}$(CR=$\frac{1}{4}$),$B_{interCAE}$(CR=$\frac{1}{4}$),
$B^T_{intraCAE}$(CR=$\frac{1}{4}$),$B^T_{interCAE}$(CR=$\frac{1}{4}$));

}

else if("the number of backward" <= a pre-fixed number) {

Backward mode;

}

else {

Choose CR = $\frac{1}{4}$ and perform CAE;

}

Note that the assumption on CR and distortion is used here in order to make the algorithm simpler. We found that this finite memory restriction on the greedy algorithm seems to be effective when shape data patterns are simple. However, if shape data patterns are complex and only a few steps of finite back-tracing are allowed, buffer overflow happens. The results are shown in Figures 4-8- 4-13 (see Section 5). An important observation is that the steady state quality is almost independent of the maximum buffer size. If we compare $B_{max} = 1000$ and $B_{max} =$

500, the steady state quality looks similar except from the first several frames. Based on this observation, we propose the following fast approximation algorithm.

4.4.4 Marginal Buffer Reservation Algorithm

We observe that the maximum buffer size doesn't make much difference in steady state quality. Therefore, using a slightly smaller buffer will not make much difference in steady state quality either. The idea of this algorithm is to take a smaller buffer size $B_{virtual}$ than the maximum buffer size B_{max}. And if there is no way to maintain the current buffer size under the $B_{virtual}$, then we allow the algorithm to generate more bits thus making the buffer size $B_{virtual}$ exceeded. Note that in this case we don't need to go backward. Since the algorithm always assigns CR=$\frac{1}{4}$ after buffer saturation, the occupancy will go under $B_{virtual}$ quickly. Therefore, the pseudo-code is exactly the same to that of finite memory modification when the maximum buffer size is changed to $B_{virtual}$ and the backward unit is removed. One nice property of this algorithm is that the computational complexity is much lower than that of the finite memory case.

The CR value and encoding mode decision for each BAB is made according to the following pseudo-code.

```
if(ALL0(BAB)) {
            all_0 mode;
}
else if(ALL255(BAB)) {
            all_255 mode;
}
else if(MC_BAB == BAB) {
            No Update mode;
```

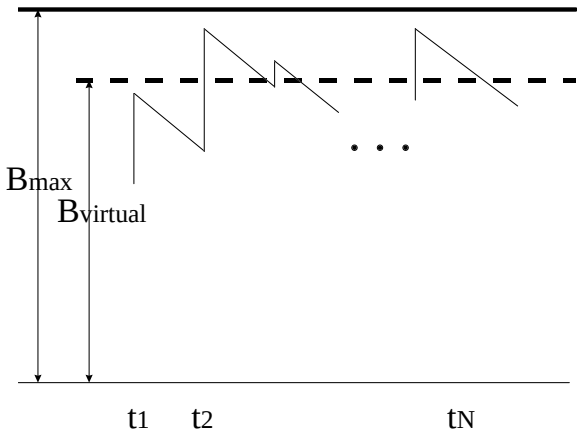


Figure 4-7: Marginal buffer reservation.

}

else if(Min{$B_{intraCAE}$(CR=1),$B_{interCAE}$(CR=1),$B^T_{intraCAE}$(CR=1),$B^T_{interCAE}$(CR=1)}

< $B_{virtual}$) {

Use $MODE_{min}$($B_{intraCAE}$(CR=1),$B_{interCAE}$(CR=1),

$B^T_{intraCAE}$(CR=1),$B^T_{interCAE}$(CR=1));

}

else if(Min{$B_{intraCAE}$(CR=$\frac{1}{2}$),$B_{interCAE}$(CR=$\frac{1}{2}$),$B^T_{intraCAE}$(CR=$\frac{1}{2}$),$B^T_{interCAE}$(CR=$\frac{1}{2}$)}

< $B_{virtual}$) {

Use $MODE_{min}$($B_{intraCAE}$(CR=$\frac{1}{2}$),$B_{interCAE}$(CR=$\frac{1}{2}$),

$B^T_{intraCAE}$(CR=$\frac{1}{2}$),$B^T_{interCAE}$(CR=$\frac{1}{2}$));

}

else if(Min{$B_{intraCAE}$(CR=$\frac{1}{4}$),$B_{interCAE}$(CR=$\frac{1}{4}$),$B^T_{intraCAE}$(CR=$\frac{1}{4}$),$B^T_{interCAE}$(CR=$\frac{1}{4}$)}

< $B_{virtual}$) {

Use $MODE_{min}$($B_{intraCAE}$(CR=$\frac{1}{4}$),$B_{interCAE}$(CR=$\frac{1}{4}$),

$B^T_{intraCAE}$(CR=$\frac{1}{4}$),$B^T_{interCAE}$(CR=$\frac{1}{4}$));

}

else {

Choose CR = $\frac{1}{4}$ and perform CAE;

}

4.5. A Low-Bit-Rate-Tuned Algorithm

To prevent coded images from being unacceptable at a low bit rate, in [61] it is recommended to maintain CR value as 1 or $\frac{1}{2}$. The idea of the low bit rate tuned algorithm is to take a much smaller $B_{virtual}$ size than the maximum buffer size B_{max} with smaller set of CR values which don't produce deteriorating perceptual quality. And if there is no way to maintain the current buffer size under the $B_{virtual}$, then we allow the algorithm to generate more bits thus making the buffer size $B_{virtual}$ exceeded. The algorithm assigns CR=$\frac{1}{2}$ after buffer saturation, and hence the occupying size will go under $B_{virtual}$ quickly.

The CR value and encoding mode decision for each BAB is made according to the following pseudo-code.

```
if(ALL0(BAB)) {
            all_0 mode;
}
else if(ALL255(BAB)) {
            all_255 mode;
}
else if(MC_BAB == BAB) {
            No Update mode;
```

}

else if(Min{$B_{intraCAE}$(CR=1),$B_{interCAE}$(CR=1),$B^T_{intraCAE}$(CR=1),$B^T_{interCAE}$(CR=1)}
< $B_{virtual}$) {

Use $MODE_{min}$($B_{intraCAE}$(CR=1),$B_{interCAE}$(CR=1),
$B^T_{intraCAE}$(CR=1), $B^T_{interCAE}$(CR=1));

}

else if(Min{$B_{intraCAE}$(CR=$\frac{1}{2}$),$B_{interCAE}$(CR=$\frac{1}{2}$),$B^T_{intraCAE}$(CR=$\frac{1}{2}$),$B^T_{interCAE}$(CR=$\frac{1}{2}$)}
< $B_{virtual}$) {

Use $MODE_{min}$($B_{intraCAE}$(CR=$\frac{1}{2}$),$B_{interCAE}$(CR=$\frac{1}{2}$),
$B^T_{intraCAE}$(CR=$\frac{1}{2}$),$B^T_{interCAE}$(CR=$\frac{1}{2}$));

}

else {

Choose CR = $\frac{1}{2}$ and perform CAE;

}

4.6. Experimental Results

The sequences in QCIF format referred to as "Akiyo" and "Children" were processed according to Version 8.0 of the MPEG-4 Video Verification Model. Note that we used our own implementation of a shape encoder and decoder, including specific implementations for size conversion and all CAE coding modes. Since we only deal with the shape coder, the texture coder was not used here. Therefore, in our experiment all necessary information about texture motion vectors were saved in a temporary file at the encoder, and retrieved from it at the decoder.

An important observation from our experiments is that the steady state quality of optimal buffered compression is insensitive to the buffer size B_{max}. Figures 4-8

and 4-9 show that the steady state quality, say 6th frames of (f), seem to be similar. The difference is only in the first few frames; they are best coded until the current buffer occupancy reaches the maximum buffer size B_{max}. Since the outgoing channel rates are the same (i.e., $r = 6$) in Figures 4-8 and 4-9, the buffer characteristics are to be the same after a certain buffer point. Note that these results are commonly observed based on both the finite backward modification and the marginal buffer reservation algorithm. In this specific example, 2 backward steps were allowed. Generally, the shape coding doesn't need many backward steps unless the channel rate is chosen unreasonably low.

The channel rate is the average number of bits we use for each BAB block. The steady state quality is strongly connected with the channel rate r. Figures 4-10 and 4-11 show the decoded outputs at $r = 7$, while Figures 4-12 and 4-13 demonstrate them at $r = 8$. Note that Figures at $r = 8$ are of high quality. This means that each BAB can be encoded by 8 bits on average for such quality. Results vary when the input sequences have different degrees of the pattern of complexity (the shape of Akiyo has relatively simple pattern). If we take more than 10 bits as the channel rate, we can have almost perfect decoded images for the Akiyo sequence. In this case, for each BAB the CR value of "1" is highly likely. If we take less than 5 bits as the channel rate, we will have distorted images. With the similar reason for each BAB the CR value of "$\frac{1}{4}$" is highly likely. Moreover, in this case, it is very difficult to make the encoding buffer not overflow, since the compressed data cannot be lower than such a low channel rate. Let's call that range of the channel rate "break down" region. In Akiyo sequence, meaningful distortion usually happens at channel rate r from 6 based on our experiments, while almost noiseless coding is adopted/applied from channel rate 9.

Binary data for a BAB is of 16×16 bits (i.e., 256). If we take the channel rate

(B_{max},r)	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame
$(1000,r=6)$	0	0	0.000315	0.005168	0.007457	0.007536
$(1000,r=7)$	0	0	0	0	0.000631	0.003472
$(1000,r=8)$	0	0	0	0	0	0
$(500,r=6)$	0.000315	0.007062	0.007536	0.007536	0.007930	0.007536
$(500,r=7)$	0	0.002840	0.002998	0.003472	0.003393	0.003472
$(500,r=8)$	0	0	0	0.000552	0.001183	0.000907

Table 4.1: Dn from the finite backward modification.

$(B_{virtual},r)$	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame
$(960,r=6)$	0	0	0.000907	0.005247	0.007102	0.007339
$(980,r=7)$	0	0	0	0	0.000986	0.003472
$(990,r=8)$	0	0	0	0	0	0
$(460,r=6)$	0.000828	0.006352	0.006668	0.006786	0.007102	0.007339
$(480,r=7)$	0	0.003432	0.003117	0.003156	0.003393	0.003472
$(490,r=8)$	0	0	0	0.000552	0.001302	0.000907

Table 4.2: Dn from the marginal buffer reservation algorithm with $B_{max} = 1000$ and $B_{max} = 500$.

6 for Akiyo sequence, the compression ratio is 42.66 to 1 in steady state. If we take the channel rate 8, the compression ratio is 32.00 to 1 in steady state. Note that the channel rate (i.e., 6, 7, 8) contains overhead. However, in our experiment, motion vector data are not included in the output bits, but are in a temporary file, as we explained earlier. Actually, the added bits as a differential motion vector data are not many, because the differential motion vector is very close to "0" based on our proposed algorithm. Therefore, the compression ratios will still be around those numbers, when we consider the entire MPEG-4 codec (i.e., texture and shape codec).

Note that these arguments hold for Figures 4-14 to 4-19 as well. The general characteristics are the same as those in Figures 4-8 to 4-13. The only difference is that the latter uses the marginal buffer algorithm.

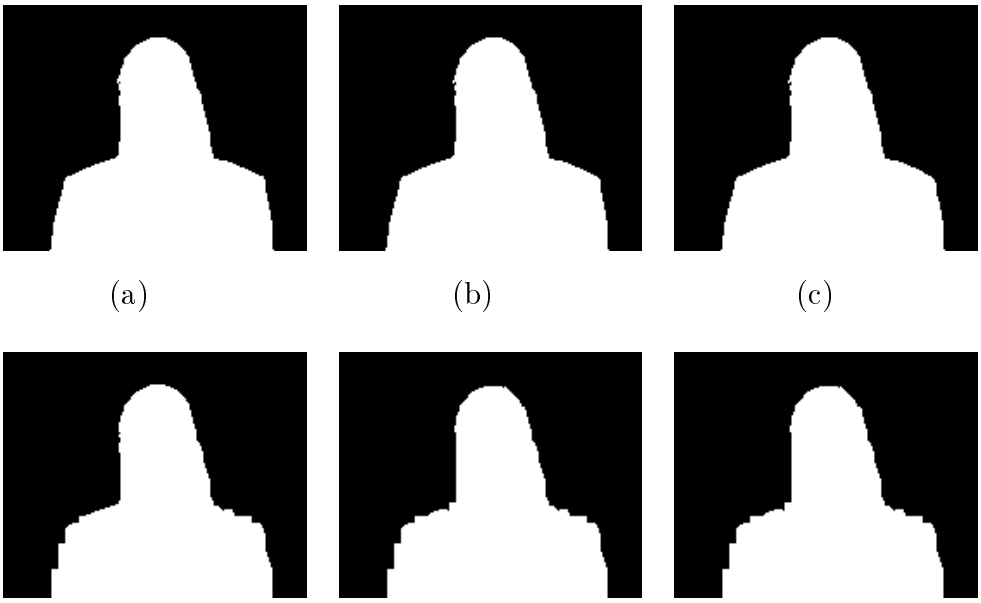
Table 4.6. shows the PSNR obtained from the finite backward modification. If we take a look at the (1000, $r = 6$) and (500, $r = 6$) cases, the steady state PSNRs seem to be similar around 21. This explains, once again, that the maximum buffer size B_{max} doesn't affect the steady state quality. Some of the values are "infinite," which means that the current buffer occupancy doesn't approach B_{max}. Table 4.6. also shows the PSNR obtained from the marginal buffer reservation algorithm. Note that the PSNR differs in the first few frames between the finite backward modification and the marginal buffer reservation algorithm. For example, in the (500, $r = 6$) and (460, $r = 6$) cases, the PSNRs for frames later than 2nd frame are stablized around 21.

Figures 4-20 and 4-22 depict the occupancy of the encoding buffer for the finite backward modification. An important aspect is that in a lower channel rate the encoding buffer is overflowed from time to time. It is very important to understand not only that finite backward modification is computational intensive but also that there is relatively weak guarantee that the encoding buffer is not overflowed. Therefore, to ensure that the channel rate is set to be reasonable is a basic assumption for using the algorithm. However, the buffer occupancy characteristics from the marginal buffer reservation algorithm, which is proposed as a fast approximation algorithm for the shape coding, seems a bit better. The trends always undergo the maximum buffer size B_{max} when the virtual buffer size $B_{virtual}$ is reasonably taken. The size of $B_{virtual}$ seems to be still small enough (50 bits margin out of 1000 bit size buffer). However, when the channel rate is lower, we must take a larger $B_{virtual}$. Figures 4-21 and 4-23 depict these facts.

Complicated shape data/pattern such as the ones found in Children sequence usually have a higher "break down" threshold. Therefore, an encoder requires a higher range of the channel rate to work as shown in Figures 4-24- 4-27. Since the

Children sequence is more complicated than Akiyo, a higher channel rate should be given.

Figures 4-26 and 4-27 depict the results of the low bit rate tuned algorithm. If we consider the $r = 15$ case, we don't see much difference between Figures 4-25 and 4-27. If we consider $r = 12$ case (a relatively low bit rate), on the other hand, we see some quality difference between Figures 4-24 and 4-26 (if you take a look at children's legs). An important aspect is that in a very low channel rate the encoding buffer fluctuates (sometimes it will brake down). Therefore, more margin is necessary for a virtual buffer than that of the fast algorithm. It is very important to note that low bit rate tuned algorithm distributes a big error in a spot into a small error over a large area in order to improve the perceptual quality.



(a) (b) (c)

(d) (e) (f)

Figure 4-8: Decoded outputs (MPEG-4 shape decoder $B_{max} = 1000$, $r = 6$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

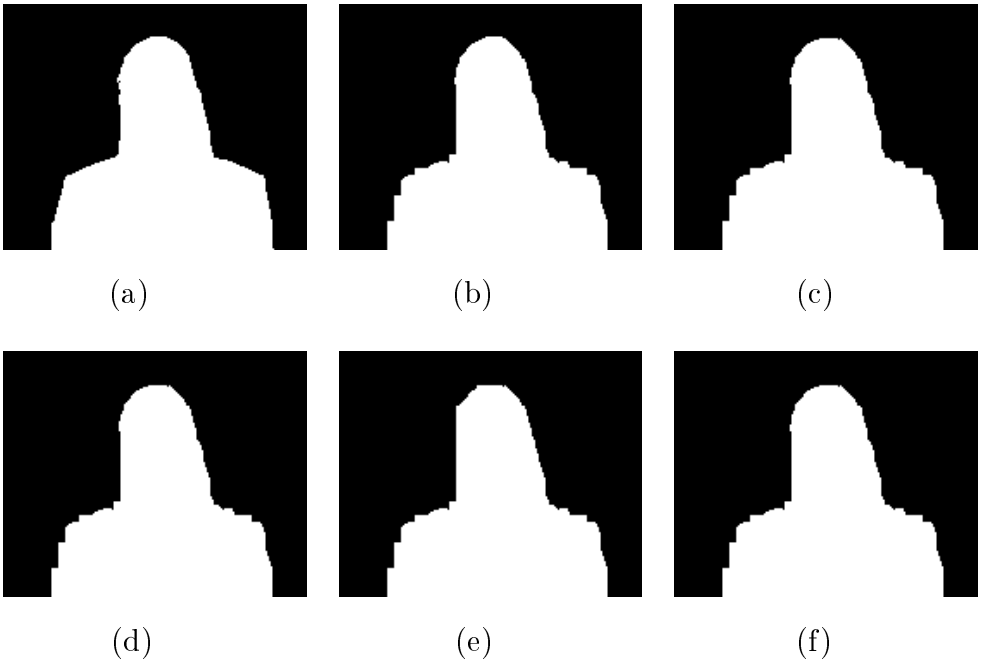


Figure 4-9: Decoded outputs (MPEG-4 shape decoder $B_{max} = 500$, $r = 6$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

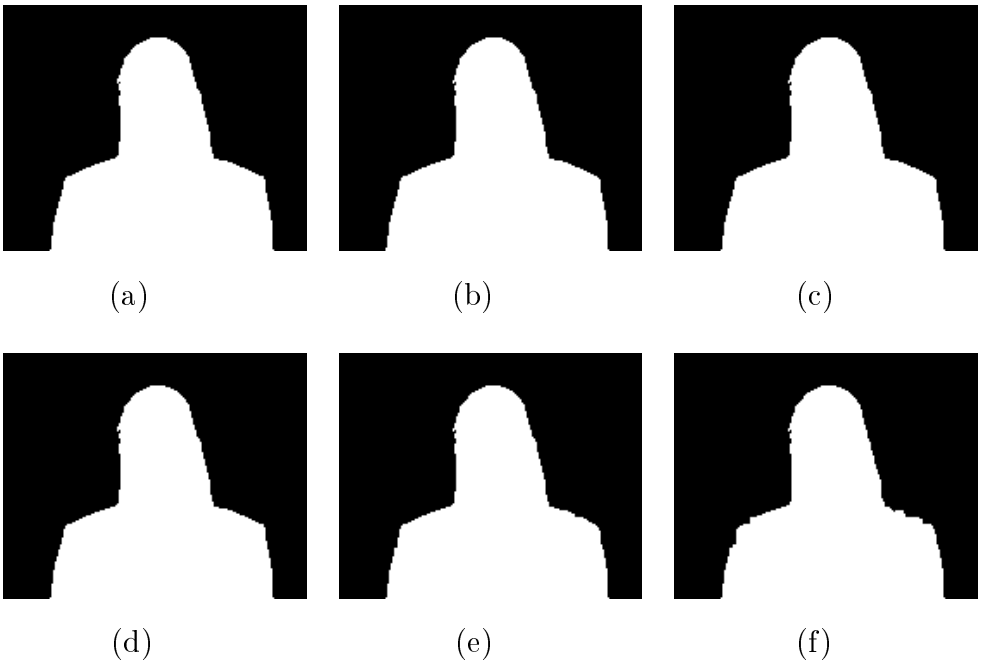
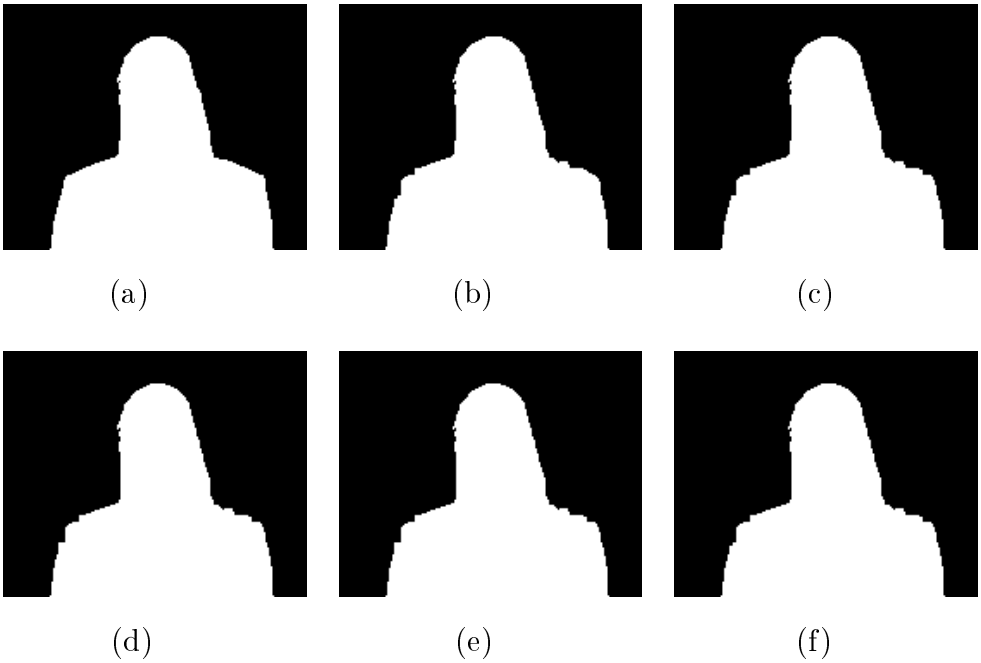


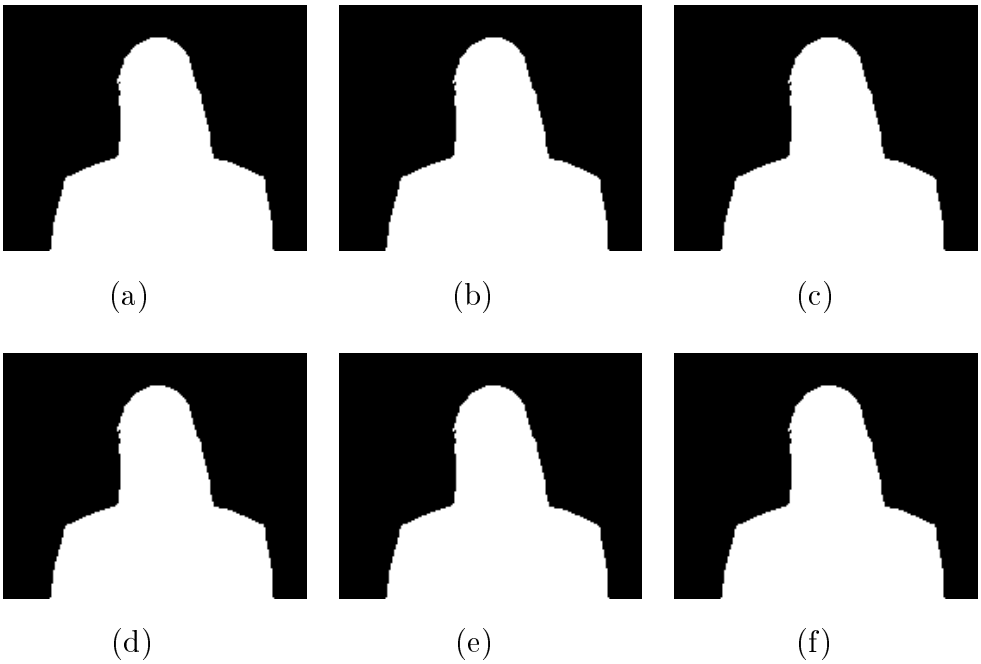
Figure 4-10: Decoded outputs (MPEG-4 shape decoder $B_{max} = 1000$, $r = 7$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 4-11: Decoded outputs (MPEG-4 shape decoder $B_{max} = 500$, $r = 7$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 4-12: Decoded outputs (MPEG-4 shape decoder $B_{max} = 1000$, $r = 8$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

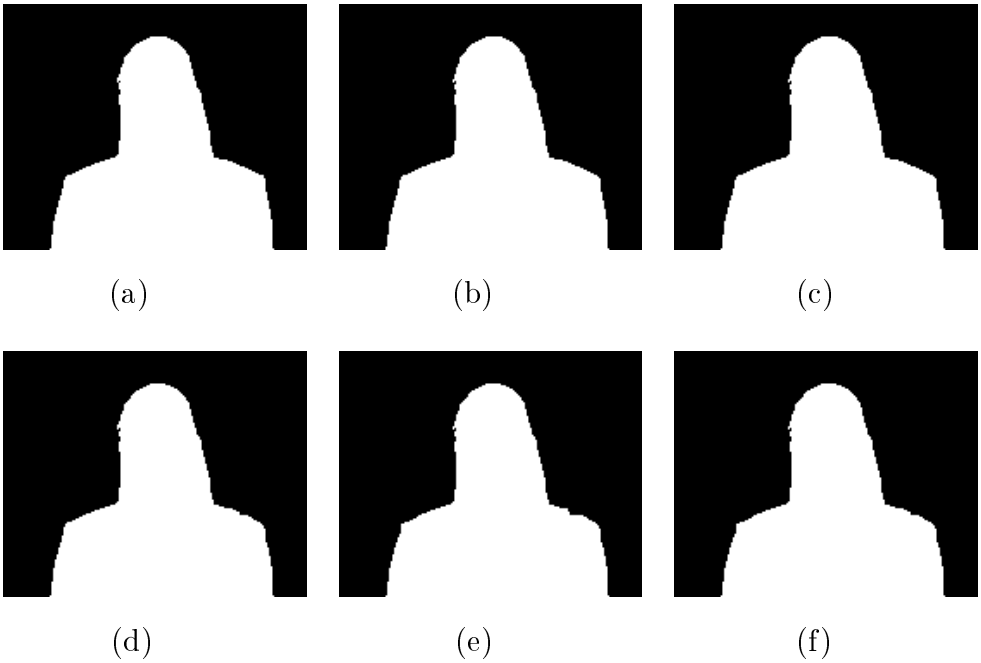


Figure 4-13: Decoded outputs (MPEG-4 shape decoder $B_{max} = 500$, $r = 8$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

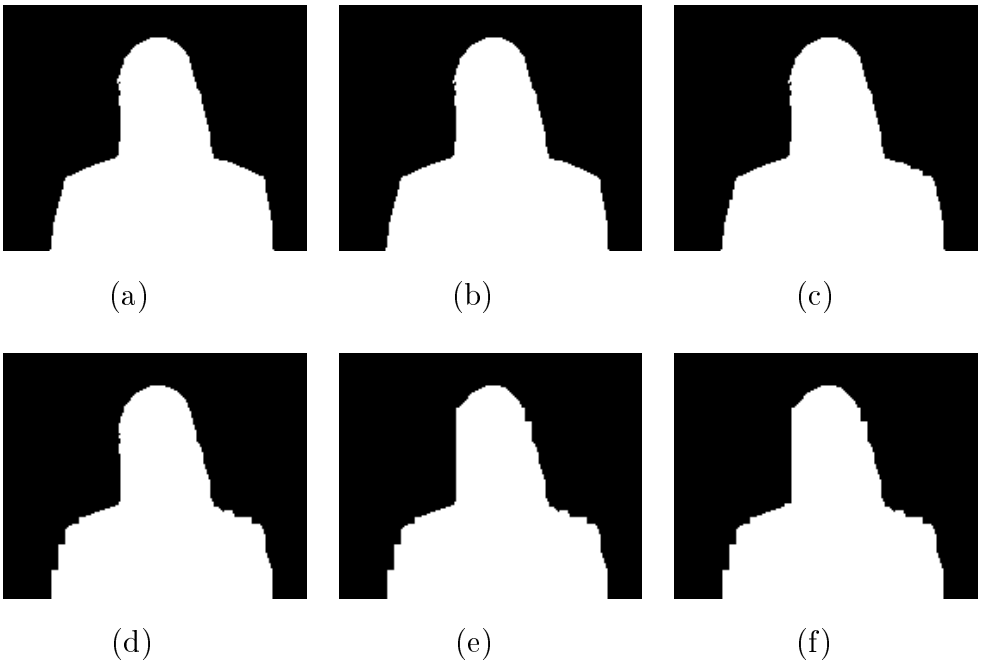
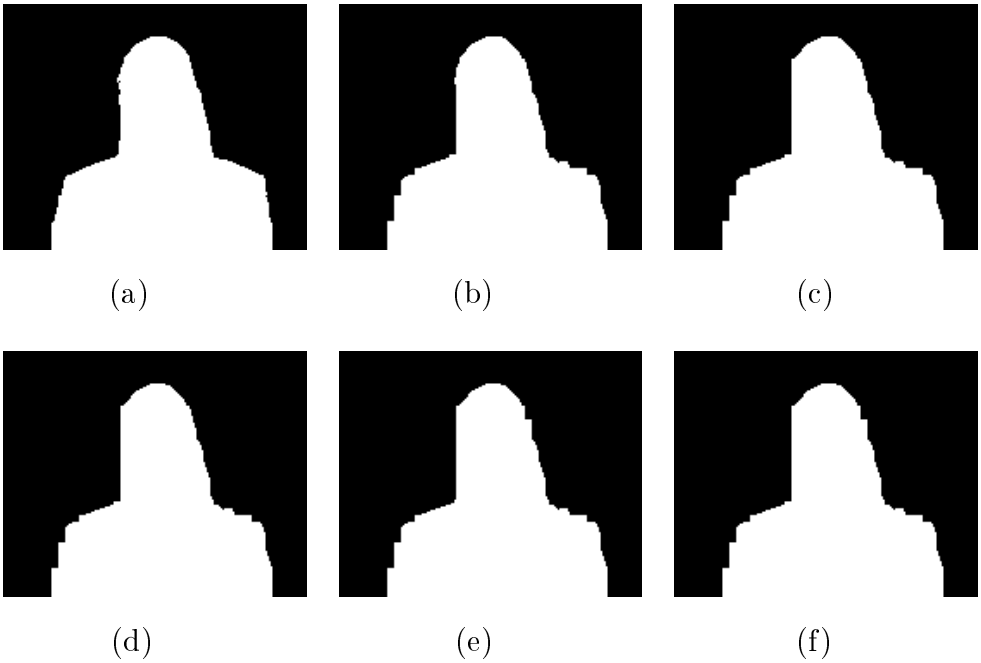


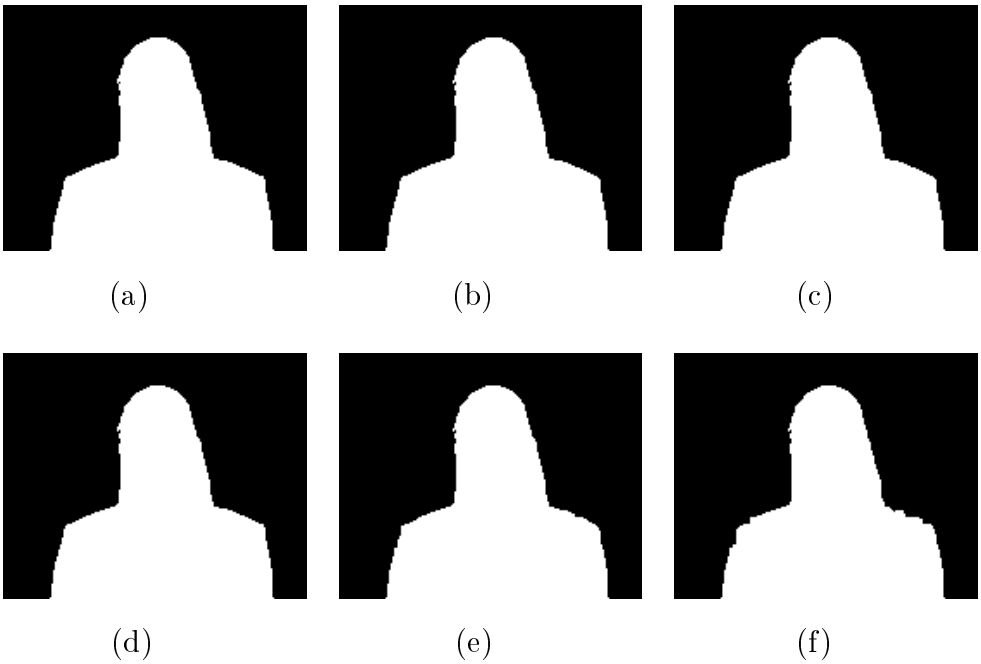
Figure 4-14: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 960$, $r = 6$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

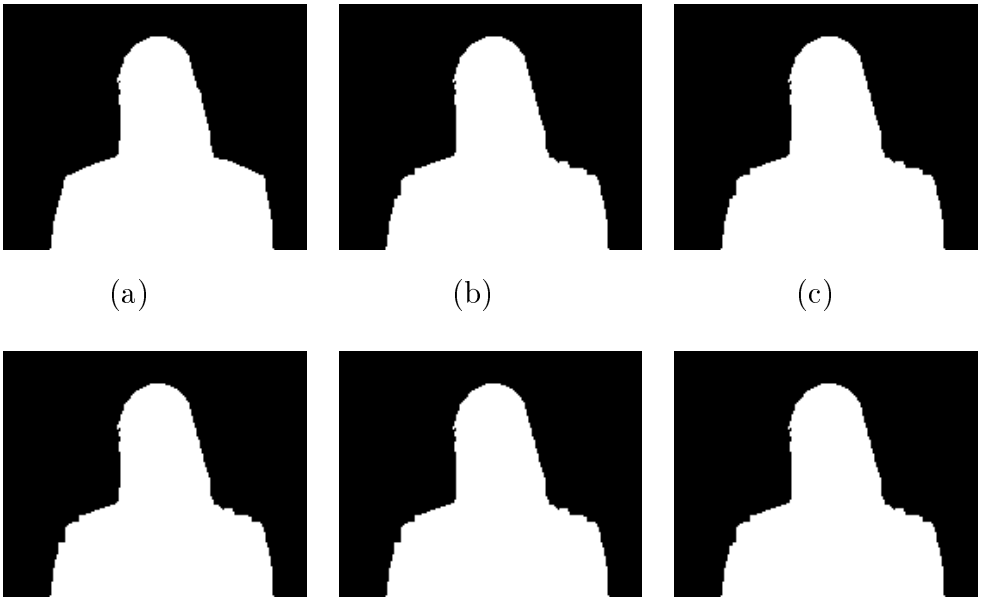
Figure 4-15: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 460$, $r = 6$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

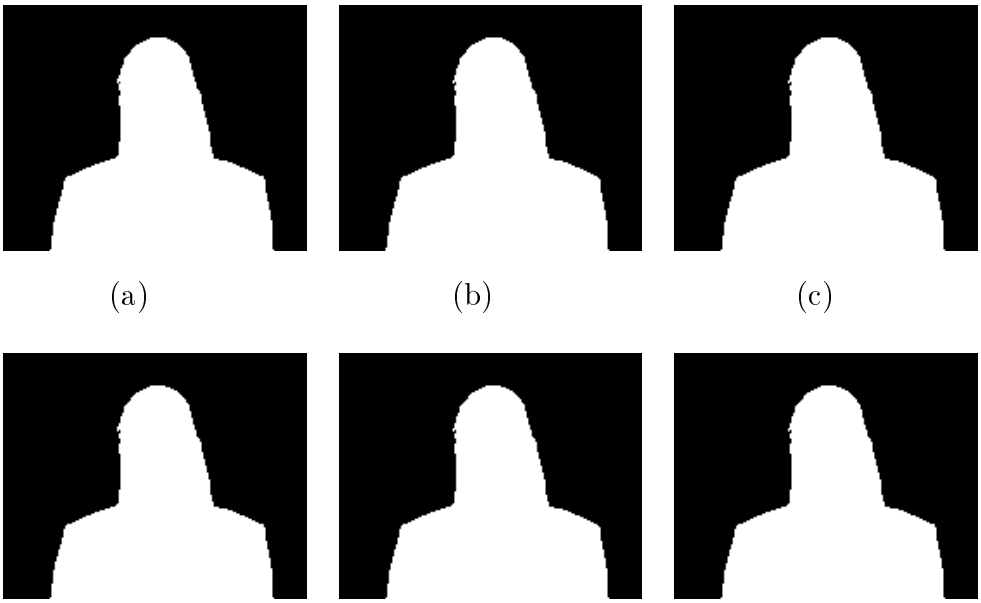
Figure 4-16: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 980$, $r = 7$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 4-17: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 480$, $r = 7$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 4-18: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 990$, $r = 8$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

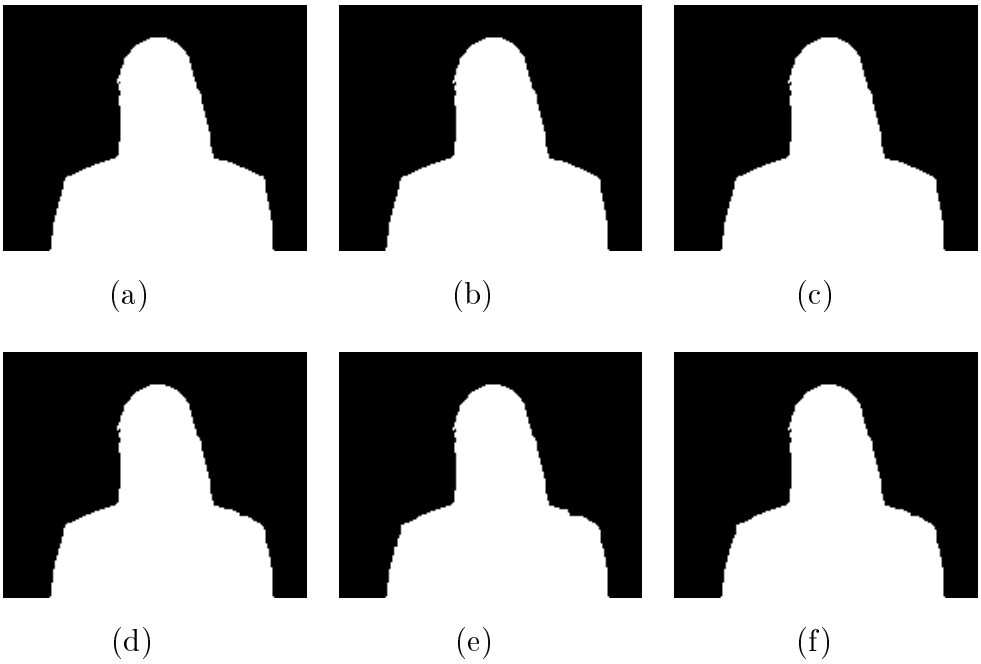
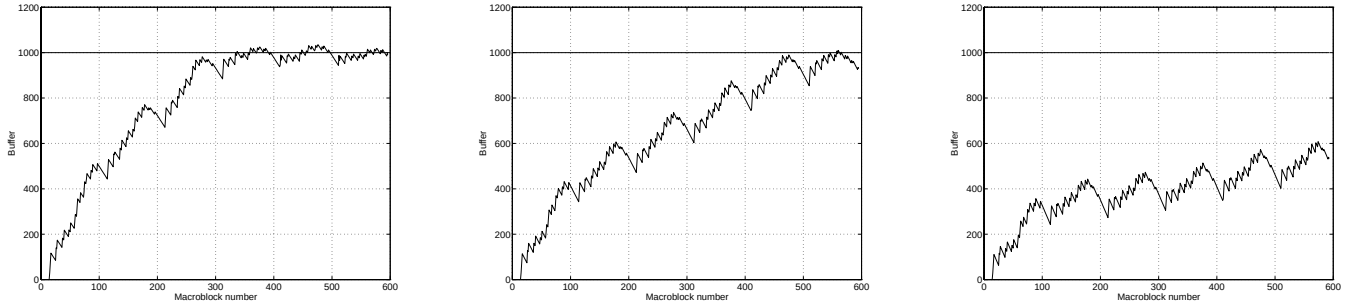


Figure 4-19: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 490$, $r = 8$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

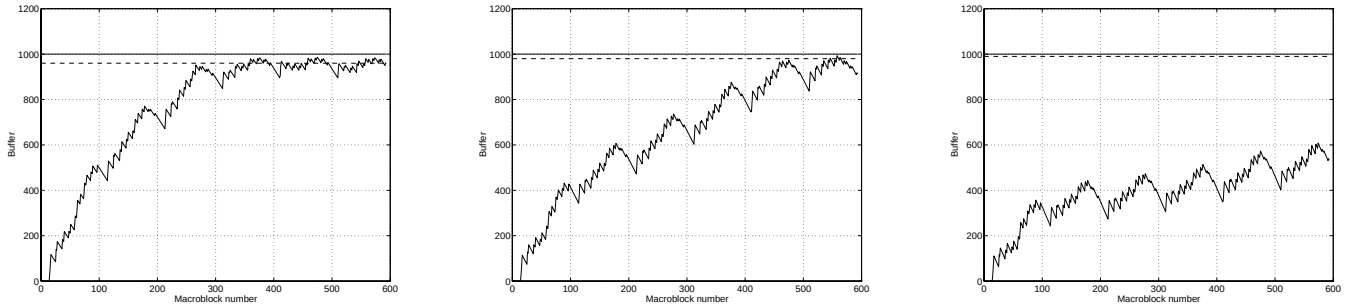
4.7. Concluding Remarks

In this chapter, we considered the problem of optimal buffered compression for MPEG-4 shape coding. The current MPEG-4 shape coding scheme, which is totally new to the coding standards arena, consists of two sub-steps. First, distortion is introduced by a size conversion process. Then, context-based arithmetic encoding is applied. Considering the size conversion process as a virtual quantizer, we formulated the buffer-constrained adaptive quantization problem for the shape encoder. We also developed an algorithm for the optimal solution under buffer constraints. For computational efficiency, we simplified the original algorithm to a fast approximation algorithm with additional consideration about marginal buffer reservation. Experimental results were given using an MPEG-4 shape coder confirming that the shape data were compressed efficiently and without overflow.



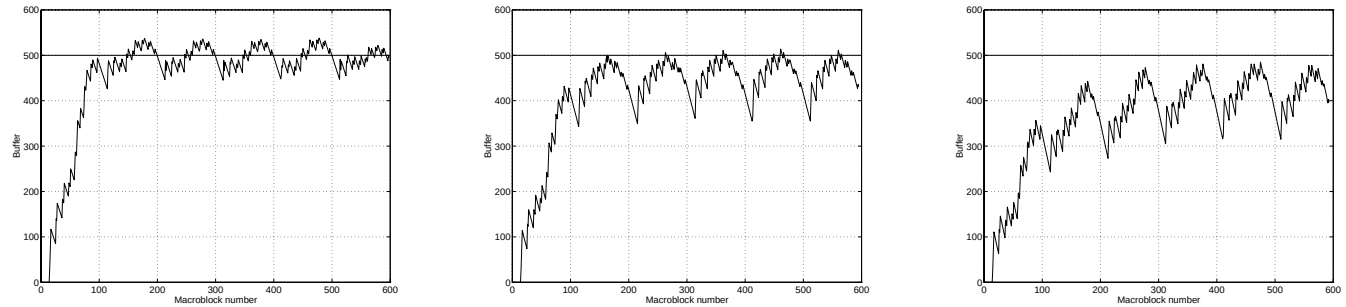
(a) (b) (c)

Figure 4-20: Buffer occupancy (MPEG-4 shape encoder $B_{max} = 1000$) : (a) $r = 6$, (b) $r = 7$, (c) $r = 8$.



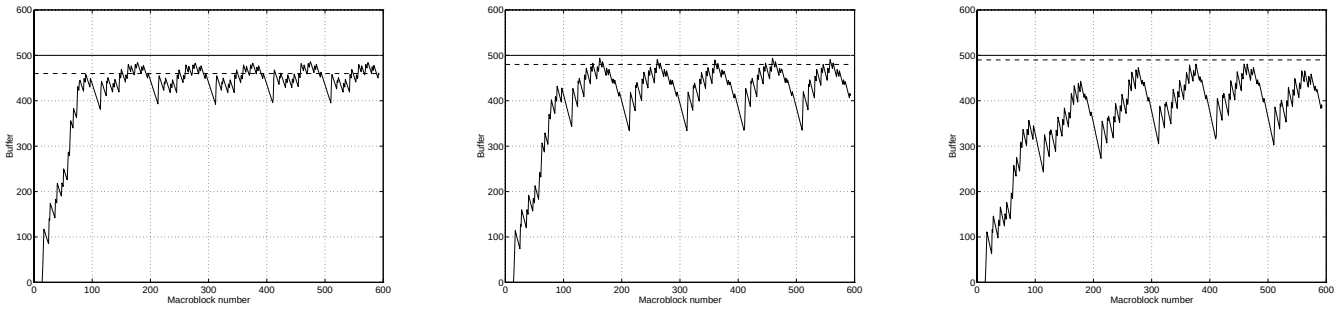
(a) (b) (c)

Figure 4-21: Buffer occupancy (MPEG-4 shape encoder $B_{max} = 1000$) : (a) $B_{virtual} = 960$ and $r = 6$, (b) $B_{virtual} = 980$ and $r = 7$, (c) $B_{virtual} = 990$ and $r = 8$.



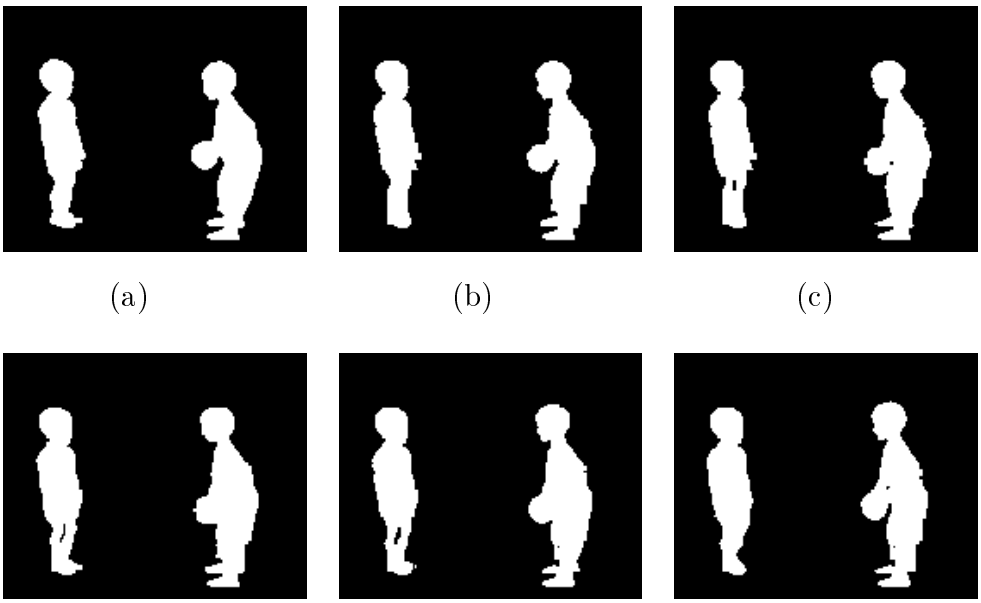
(a) (b) (c)

Figure 4-22: Buffer occupancy (MPEG-4 shape encoder $B_{max} = 500$) : (a) $r = 6$, (b) $r = 7$, (c) $r = 8$.



(a) (b) (c)

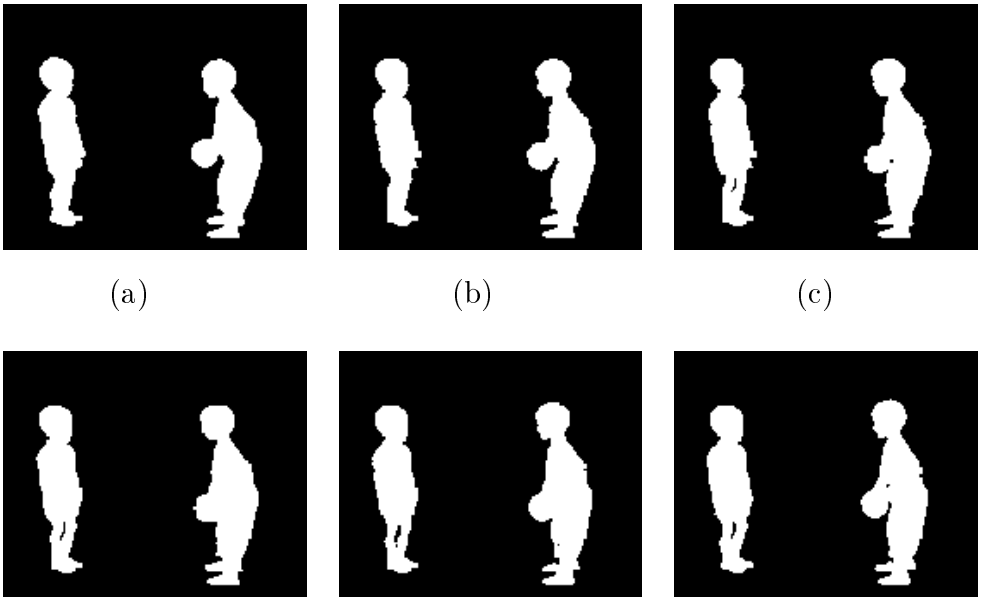
Figure 4-23: Buffer occupancy (MPEG-4 shape encoder $B_{max} = 500$) : (a) $B_{virtual} = 460$ and $r = 6$, (b) $B_{virtual} = 480$ and $r = 7$, (c) $B_{virtual} = 490$ and $r = 8$.



(a) (b) (c)

(d) (e) (f)

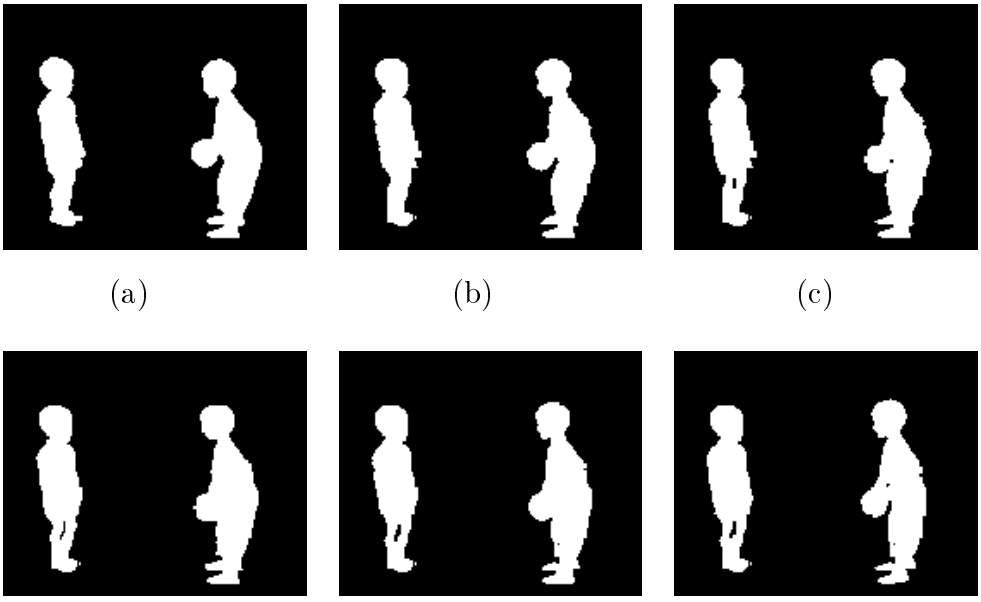
Figure 4-24: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500$, $r = 12$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

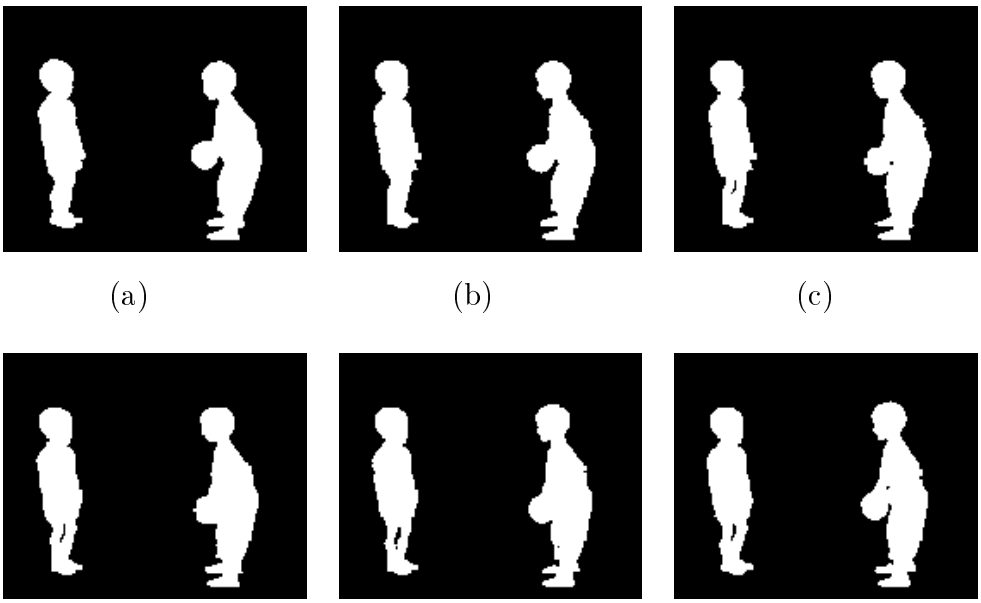
Figure 4-25: Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500$, $r = 15$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 4-26: Low-bit-rate-tuned decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500$, $r = 12$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 4-27: Low-bit-rate-tuned decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500$, $r = 15$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

Chapter 5

Spatio-Temporal Activity-Assisted Rate Control for MPEG-4 Object-Based Video Coding

5.1. Introduction

Video compression is one of the core technologies behind the multimedia revolution since the bandwidth of uncompressed video is usually much larger than the available channel capacity. With the wide range of bandwidth and storage requirements in existing networks, video compression algorithms should provide the highest visual quality at all bit rates. In fact, video lends itself to compression because natural scenes are of high perceptual, temporal, and spatial redundancies.

In object-based coding such as MPEG-4 video, foreground objects are usually homogeneous semantic elements such as human beings, animals, machines, etc.. Recent experiments show that human observers are usually more sensitive to visual cue areas in a foreground object [50, 51, 73, 75, 6, 62] since human eyes lock on to and track a particular foreground object such as a person's face. This property is especially effective in very low bit rate video coding. In such "noisy" situation (i.e., very low bit rate), this is important because human perception locates important semantic elements first. This was reported in the first competition stage of MPEG-

4 video development by a group of people at AT&T. In this chapter, we call this property "human structural perception (HSP)."

For example, blockiness and blurriness around talking person's eyes in visual communications at a very low bit rate make communicating people very unpleasant. Thus, minimizing errors in these areas such as eyes is desirable. This is in particular important aspect in object-based video where semantic elements are captured and generated as objects which perhaps have different importance to human perception, object-by-object. Therefore, this technique to assign more bits to perceptually important areas is preferable while preserving the same average bit rate with traditional rate control algorithms.

For foreground human object, generally HSP works well, while for background object, generally human visual perception (HVP) dominates. This is because people don't lock on to and track background objects, but a foreground object. For very low bit rate video, generally HSP works well, while for high quality video, generally HVP dominates. This is because people try to locate important semantic elements first for noisy situation. Therefore, it is better to apply different policy based on different situation about where to spend more bits.

If the semantics in scenes are not obvious, activity can replace the role of models due to the fact that visual cue areas in objects are usually active. This is especially true for objects about human beings (such as in human talking sequences) and animals. In this chapter we assume that input sequences have more important visual cues in active areas and that, to use of it, only human foreground objects are our target. Note that the same method can be applied to background objects, where HVP dominates, simply with assigning more bit budget to non-activity blocks.

Previous research on spatially adaptive coding schemes based on "activity levels" are effective block coding [37] and classified vector quantization [70]. In these

papers, the adaptation is used to classify an image block after it has been extracted from the image. More recent technique that does not involve direct extraction of blocks, but rather, is a global method to identify the active areas of human images (i.e., foreground object) to be coded more finely, was presented in [73]. It presented a modified JPEG algorithm that provides better visual quality than the Q-factor scaling method commonly used with JPEG implementations. The quantization step sizes are adapted to the activity level of the block, and the activity selection is based on edge-driven quadtree decomposition of the image [73].

On the other hand, adaptive coding schemes based on activity levels for "video" are in a new challenging area since techniques developed for still images cannot be easily applied for motion pictures. In order to sustain a reasonable target bit rate and to prevent the encoding buffer from being over/under-flowed, resources should be timely and accurately managed through quantization parameters (QPs). For example, if a certain activity area is coded finely, areas for other activity areas should be suffering to maintain a constant output target bit rate. In the MPEG-style video compression standard [41, 40, 1], compression ratio can be adjusted by uniformly scaling the quantization matrix by a multiplicative factor QP. And values of QPs correspond to a buffer's or virtual buffer's occupancy of the encoder in many cases. A Higher QP gives better compression but increased blockiness, while a lower QP gives better image quality but worse compression. MPEG suggests luminance and chrominance quantization matrices resulting from human perception studies and requires that a single matrix be used for all blocks in an image component. The quantization matrix is defined by standard but it can be also supplied by the user being stored or transmitted with the compression image.

Using activity-selective quantization in anchor frames of video sequences, for example, will not only improve the visual quality of foreground objects at low bit

rates but it will also reduce the "mosquito" noise prevalent in active areas of the objects. Mosquito noise is the high-frequency temporal noise that appears in low bit rate videos and since it is most visible around the active areas of the video, finer quantization in these areas is expected to reduce mosquito noise in the foreground objects. For head-and-shoulders video conferencing sequences, for example, activity selective quantization is expected to reduce the mosquito noise around the edges of the mouth and eyes of the speaker, thus improving the overall visual quality of the video conferencing sequences [73].

There are many methods proposed for maintaining an outcoming bit rate constant. Optimal buffered compression was recently proposed in [60] to provide optimal buffer control strategies for video sequences using a finite buffer environment. The authors formalized the description of the buffer-constrained adaptive quantization problem. They then formulated the optimal solution for a given set of admissible quantizers used to code a discrete nonstationary signal sequence in a finite buffer. The most harmful fact to incorporate activity-assistance into optimal buffered compression framework is that the optimal buffered compressed rate control only monitors the occupancy of buffer. Therefore, encoders assign quite rough QP values even for high active MBs when buffer is nearly full.

In this chapter, a technique to assign more bits to visually important MBs or BABs within the optimal buffered compression framework is introduced for MPEG video coding while preserving the same average bit rate with traditional rate control algorithms. It takes on the form of spatio-temporal activity-assisted rate control in adapting the DCT coefficient quantization step sizes to the activity level of the block. In Section 5.2. and 5.3. we first derive temporal and spatial balance equations. And then Spatio-Temporal Buffer Rate Modulation (STBRM), a technique that selectively allocates bits to areas of different activity in terms of space and time

for video sequences, is proposed. Section 5.4. presents a rate control mechanism which is proper to STBRM and provide a way to classify MBs or BABs into several categories for measuring "activity level." We discuss why this technique is more appropriate for MPEG-4 video coding. In Section 5.6., experimental results are given using an MPEG-4 shape and texture codec. A discussion and concluding remarks are given in Section 5.7..

5.2. Temporal Buffer Rate Modulation

In the next three sections, we call a unit of coding simply a "block." It implies a BAB in MPEG-4 shape coding, or an MB in MPEG-4 texture coding and other conventional standards. And we will as well assume that the time shot of pictures is rectangular shape (i.e., frame) in these sections. The shape will be generalized into arbitrary one (i.e., VOP) without loss of generality later.

In order to be able to spend more bits in certain frames of interest while staying within a prescribed bit budget (or avoiding buffer overflow), it is necessary to spend less bits on the remaining frames. For convenience, we define the problem to be a distribution of different bit budgets into frames of different interest in one Group Of Pictures (GOP). That is, we assume that this distribution is applied periodically to each GOP.

Regular MPEG rate control formula maps a low buffer occupancy to finer QP values while it does a high buffer occupancy to rougher QP values. Since buffer occupancy relates to a outgoing buffer rate (i.e., target bit rate), a virtual buffer occupancy shall as well be evolved through a virtual (i.e., modulated) buffer rate. Therefore, blocks in improved or deteriorated quality will be obtained if the rate control formula maps a virtual buffer occupancy into a QP. The idea of buffer modulation is to deceive the rate control algorithm of encoders into considering a

modulated outgoing buffer rate at different frames, thus making it assign a modified QP. This is depicted in Figure 5-3. For example, if we set a virtual buffer rate as 32 kbps (say, for actual 30 kbps), the finer QPs are given to that specific coding unit due to more starving buffer condition. Intuitively, all the frames cannot take advantage of buffer rate modulation because some other frames should be suffering to sustain a prefixed average target bit rate. Therefore, a temporal balance equation must be found before applying the concept.

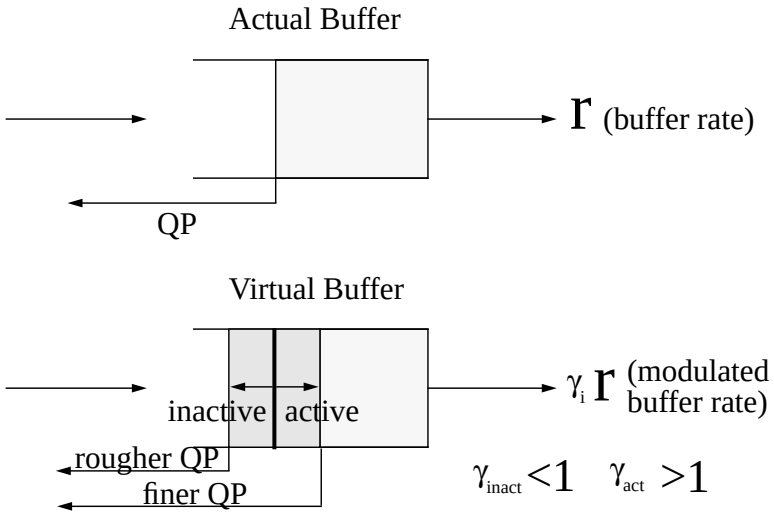


Figure 5-1: Actual and virtual buffers vs. actual and virtual buffer rates.

The purpose of temporal buffer rate modulation is to assign different "quality levels" at different frames, that differently impact on a viewer, of a video sequence. This approach in fact makes the content of the encoding buffer fluctuate intentionally. To derive a temporal balance equation, we classify frames to several quality levels. We assume that all the frame require different quality in a GOP and that a GOP is composed of total N frames, and name them as $1, 2, ..., N$ in order. We assume that there are K frames of interest $\mathcal{T}^1, \mathcal{T}^2, ..., \mathcal{T}^K$ out of N frames, with cor-

responding frame number $\mathcal{K} = \{\text{j} \mid \text{j is the frame number of K frames of interests}\}$. We assume that the coding of each macroblock uses β bits on the average, while the total number of blocks in a frame is A. Let $\beta^1, \beta^2, ..., \beta^N$ denote the target average number of bits per macroblock for the coding of each of the frames. Note that in general we would require $\beta^j > \beta$ for $j \in \mathcal{K}$, i.e., an improved quality within the frames of interest. Let also $\mathcal{T}^0$ denote the frames that belongs to none of the frames of interest, with a corresponding frame number $j \notin \mathcal{K}$ and average number of bits per macroblock β^0. In order to satisfy the given average bit budget, the following relation (i.e., balance equation) must hold in a GOP:

$$\sum_{j=1}^{N} \beta^j A = \sum_{j \in \mathcal{K}} \beta^j A + \sum_{j \notin \mathcal{K}} \beta^j A = N \ \beta \ A. \tag{5.1}$$

If the parameters $\beta^1, \beta^2, ..., \beta^N$ are given, it follows from Equation (5.1) that

$$\beta^0 = \frac{N\beta - \sum_{j \in \mathcal{K}} \beta^j}{N - K} \tag{5.2}$$

This formula defines the "equivalent average quality" for the image frame that is exterior to all frames of interests (henceforth called exterior region for brevity), and is uniquely specified by the desired average coding quality of the coded regions and frame length. In most cases, it is more convenient to express Equation (5.2) in terms of the relative average qualities $\gamma^j \equiv \beta^j/\beta$, j=0, ..., N:

$$\gamma^0 = \frac{N - \sum_{j \in \mathcal{K}} \gamma^j}{N - K} \tag{5.3}$$

Note that if $\gamma^j > 1$ for all $j > 1$, then $\gamma^0 < 1$ as should be expected.

The potential problem is that in order to analyze the activity of frames, we may need to use extensive buffering which is very bad. If we don't do that, it may be

hard to properly budget your bit allocation strategy in the temporal dimension. To avoid extensive buffering in temporal domain, we propose a heuristic activity decision policy in this section based on "monotonicity property"; we assign more bits in anchor frames.

An MPEG-style codec takes on a form of dependent coding frameworks, where the set of available R-D operating points for some coding units depends on the particular choice of R-D point for other coding units [59, 60, 72, 71]. Assume that the quantizer grades are ordered as monotonically increasing from finest to coarsest. Then, a dependent coding system has the monotonicity property, which simply implies that a "better" (i.e., finer quantized) predictor will lead to more efficient coding of the residue (whose energy decreases as the predictor quality gets better), for a typical R-D characteristics of quantizer set. For thorough proof and explanation, we refer to [59, 60, 72]. For example, if we assign finer QPs in a I frame sustaining the same target bit rate, overall distortion along with I, P and B frames in a GOP is minimized. This is also true for P and B frame case since P frame is used for predictor. Experimental results involving MPEG verify this monotonicity property for all the typical cases [59, 60, 72, 71]. Note that this heuristic activity decision policy not only decreases overall distortion measure but also reduces the mosquito noise because the quality of anchor frames is improved [73]. The experimental results about monotonicity property can be also interpreted from the view point of activity levels. Highly active blocks in P frames are modified to be more or less inactive with residual data after motion compensation. Furthermore, active blocks in B frames have more chances to be less inactive with motion-compensation since B frames use two reference frames, I and P. Therefore, we adopt monotonicity property for architecting a heuristic activity decision policy on temporal buffer rate modulation.

For example, we try to classify frames into two types; I frames and other frames.

Here, frames of interests are I frames and let γ^I denote relative average quality for I frames. And let's assume that a GOP is composed of N frames. Then,

$$\gamma^0 = \frac{N - \gamma^I}{N - 1}. \tag{5.4}$$

The extension of two type classification into three types of frames can be done in a similar way. Note that two temporal classification can be used mainly for MPEG-4 video coding because it already takes care of it for P and B frames (i.e., no such thing for MPEG-1/2). That is, the absolute value of quantization step size is re-defined (i.e., rougher) in B frames even for the same value of QP.

5.3. Spatial Buffer Rate Modulation

The purpose of this section is to assign different "quality levels" at different portions, that bear perceptual significance to a viewer, in a frame. In order to be able to spend more bits in regions of interest while staying within a prescribed bit budget (or avoiding buffer overflow), it is necessary to spend less bits on the remaining image areas. We assume that there are M activity classes of interest $\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_M$ in an image, with corresponding areas $A_1, A_2, ..., A_M$ (in block units). We require that the regions are non-overlapping, i.e.

$$\mathcal{A}_i \cap \mathcal{A}_j = 0 \ when \ i \neq \ j. \tag{5.5}$$

The rectangular region encompassing the whole image is denoted by $\mathcal{A}$, and its area by A. We also assume that the coding of each macroblock uses β bits on the average, when the target budget rate is R and the buffer size is B_{max}. Let $\beta_1, \beta_2, ..., \beta_M$ denote the target average number of bits per macroblock for the coding of each of the regions of interest. Note that in general we would require $\beta_i > \beta$, i.e., an

improved quality within the regions of interest. Let also $\mathcal{A}_0$ denote the portion of the image that belongs to none of the regions of interest, with a corresponding area A_0 and average number of bits per macroblock β_0. In order to satisfy the given average bit budget, the following relation (i.e., spatial balance equation) must hold in a frame:

$$\sum_{i=0}^{M} \beta_i A_i = \beta A. \tag{5.6}$$

If the parameters $\beta_1, \beta_2, ..., \beta_M$ are given, it follows from Equation (5.6) that

$$\beta_0 = \frac{\beta A - \sum_{i=1}^{M} \beta_i A_i}{A - \sum_{i=1}^{M} A_i} \tag{5.7}$$

This formula defines the "equivalent average quality" for the image region that is exterior to all objects (henceforth called exterior region for brevity), and is uniquely specified by the desired average coding quality of the coded regions and their sizes. In most cases, it is more convenient to express Equation (5.7) in terms of the relative average qualities $\gamma_i \equiv \beta_i/\beta$, i=0, ..., M:

$$\gamma_0 = \frac{A - \sum_{i=1}^{M} \gamma_i A_i}{A - \sum_{i=1}^{M} A_i} \tag{5.8}$$

Note that if $\gamma_i > 1$ for all $i > 1$, then $\gamma_0 < 1$ as should be expected.

In this chapter, we take two activity class approach. In the case, relative average quality for inactive area is represented by:

$$\gamma_0 = \frac{A - \gamma_{active} A_{active}}{A - A_{active}}. \tag{5.9}$$

We have described spatial balance equation as simple as possible in this section. We refer readers to authors' previous research of model assistance in previous chapters. Note that our previous research focused on model-assisted quality classification

and foveation in the spatial domain [31]. However, there is no model-assistance any more in this chapter and, rather, it takes on the form of activity-assistance.

5.4. Rate Control for Buffer Rate Modulation

5.4.1 A Consideration for Rate Control Formula

Typically buffer occupancy based rate control schemes monitor the occupancy of buffer (i.e., actual or virtual buffers). Therefore, the relation is represented by:

$$QP_i = f_{rate}(B_{i-1}), \tag{5.10}$$

where B_{i-1} is the buffer occupancy prior to coding block i. It is important to note that the form of Equation (5.10) is, generally, not a proper rate control formula for the buffer rate modulation scheme due to its "insensitiveness" to the buffer rate. For example, say, a certain block is in an active class. Then, the buffer rate will be modulated higher for the specific block. However, the correspondence of a finer QP is not immediately performed because it takes time that the segment for current buffer occupancy jumps to a lower segment area. Therefore, we would like to have a rate control formula which has direct relationship between QP and buffer rate r as follows.

$$QP_i = f_{rate}(B_{i-1}, r) \tag{5.11}$$

where B_{i-1} is the buffer occupancy prior to coding block i, and r is the average target rate in bits per block.

The rate control operation [60, 59] of optimal buffered compression is a well-known to be a similar form to Equation (5.11):

$$QP_i = f_{optimal}(B_{i-1}, r), \tag{5.12}$$

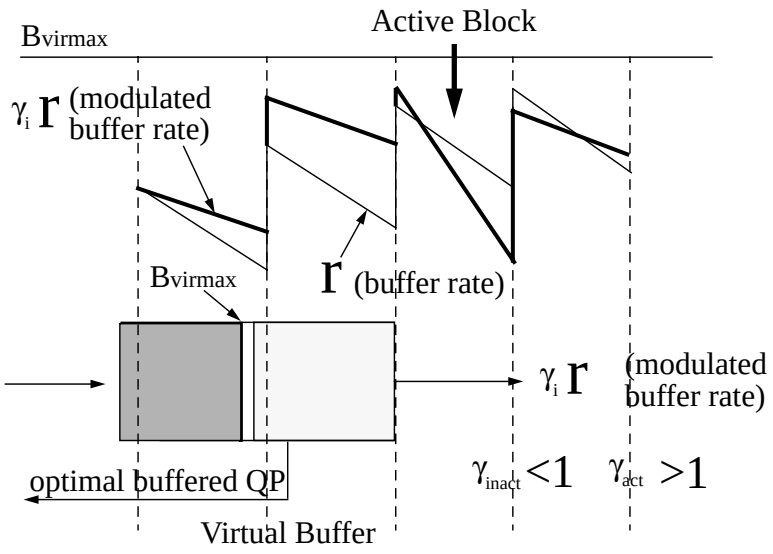


Figure 5-2: Direct relation between QP selection and buffer rate.

for MPEG-style texture coding, where B_{i-1} is the buffer occupancy prior to coding block i, and r is the average target rate in bits per block. Figure 5-2 depicts a direct effect of a buffer rate modulation to the selection of QP for that block. The output buffer evolution can be described by

$$B_i = B_{i-1} + c(i) - r, \tag{5.13}$$

where $c(i)$ is the number of bits spent to code the ith block and all its immediately preceding overhead information (headers, etc.). The function $c(i)$ depends on the input signal, as well as the current value of QP_i. Note that it is not a "rate control formula" any more, but a "rate control procedure." In the following subsection, we propose an optimal buffered compression technique for rate control procedure.

5.4.2 Rate-Modulated Optimal Buffered Compression

Recently, optimal trellis-based buffered compression was proposed by Ortega and Vetterli in [60] to provide optimal buffer control strategies for video sequences in a finite buffer environment. In order to convert rate control procedure to provide location-dependency we re-define the optimal buffered compression formulation considering buffer rate modulation.

Rate-Modulated optimal buffered compression is defined as follows.

Problem Definition: Given a set of quantizers, a sequence of blocks to be quantized, and a finite buffer with "location-dependent buffer rate," select the optimal quantizer for each block so that the total cost measure is minimized and the finite buffer never overflows.

Note that the core idea of the redefinition is to relax buffer rate condition in the original definition. Consider the allocation for N blocks, and suppose there are M quantizers available to code each block. Let d_{nm} and r_{nm} be, respectively, the distortion and the number of bits produced by the coding of block n with quantizer m, and let r be the channel rate per block. Define an admissible solution x as a selection of one quantizer for each block, i.e., a mapping from $\{1, 2, ..., N\}$ to $\{1, 2, ..., M\}$, $x = \{x(1; 0, 0), x(2; 1, 0), ..., x(N; f_f, f_b)\}$, where each $x(n; i, j)$ is the index of one of the M quantizers for block i in frame j (block index n), and f_f and f_b are, respectively, the final frame number and the final block number. Note that i and j can be derived from n since A is known (i.e., i(n) and j(n)). However, the notation is chosen to be redundant to help intuition. Therefore, $(r_{1x(1;0,0)}, ..., r_{Nx(N;f_f,f_b)})$ and $(d_{1x(1;0,0)}, ..., d_{Nx(N;f_f,f_b)})$ are, respectively, the rate and distortion for each block and a given choice of quantizers x.

The rate r now becomes location dependent, and is given by

$$r_n \equiv r_i^j \equiv \gamma_{S(n)}\gamma^{T(n)}r, \tag{5.14}$$

where the region index functions $\gamma_{S(n)}$ and $\gamma^{T(n)}$, defined in (5.3) and (5.8), associate the position of block i in frame j with the region in which it belongs to.

Now, define the buffer occupancy at stage n, B_n for a given admissible solution x. The buffer evolution can now be described by

$$B_n = max(B_{n-1} + r_{nx(n;i,j)} - r_n, 0) \tag{5.15}$$

where the number of bits spent $r_{nx(n;i,j)}$ is now region-dependent and the buffer occupancy at each block instant is increased by the coding rate of the current block and decreased by the channel rate. Note that $\gamma(i,j)$ is a spatio-temporal buffer rate modulation factor at block i in frame j.

General Formulation : (Integer Programming)

The problem is to find the mapping x that solves

$$\min_{x(n;i(n),j(n)),n=1,..,N} (\sum_{n=1}^{N} d_{nx(n;i(n),j(n))}), \tag{5.16}$$

subject to

$$B_n \leq B_{virmax}, \forall n = 1, ..., N$$

where B_{virmax} is the virtual buffer size.

It is important to note that this problem falls back to that of Ortega and Vetterli when $\gamma_i = 1$ for all i and $\gamma^j = 1$ for all j.

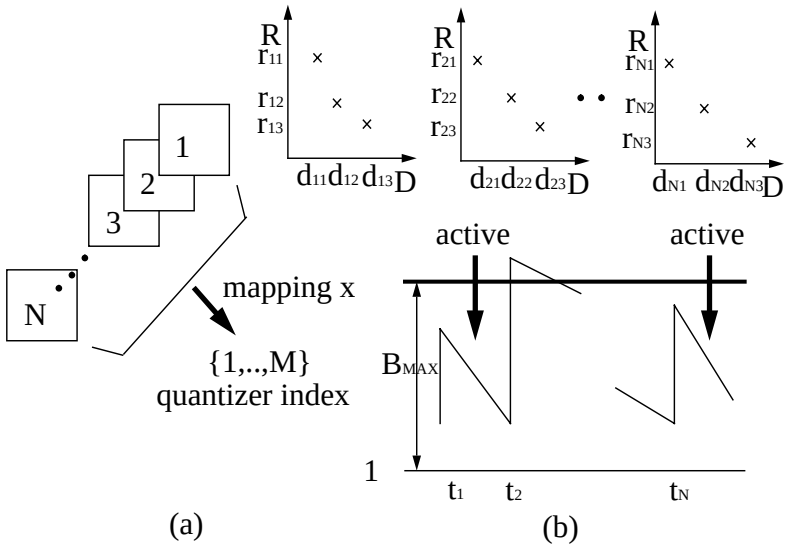


Figure 5-3: (a) Each block in the sequence has a different R-D characteristic (For a given choice of quantizers for the blocks in the sequence, we can obtain R-D points to form the composite characteristic), (b) R at t_2 is not a feasible solution with the chosen buffer size (Buffer is limited).

5.4.3 Activity Classification

There are tons of ways to classify frames and blocks into different activity groups. In this section, we just try to demonstrate a simple way to do it for texture and shape data. This measurement is not necessarily used, but for example. For instance, edge energy can be used as shown in [73]. In this chapter, we show the simplest case: 2 spatial classes and 2 temporal classes. First, the temporal classification is performed based on a type of VOP or frame such as I and others. And then, the spatial classification is performed under activity measurements (AM) of each coding block. For MPEG-4 texture coding, pixel energy variance is adopted for the activity classification as follows.

$$AM_{texture} = \sum_{j=0}^{15}\sum_{i=0}^{15}(x(i,j) - \overline{MB})^2 \tag{5.17}$$

$\overline{MB}$ is the average pixel value of that MB while $x(i,j)$ is each pixel value of a MB.

On the other hand, the measurement about activity of shape data in MPEG-4 is not obvious. In this chapter, we treat areas of delicate shape as being more active. As an example, we propose a method based on binary morphological operators in the following explanation.

Morphological processing refers to certain operations where an object is hit with a structuring element and thereby reduced to a more revealing shape. Most morphological operations can be defined in terms of two basic operations, "erosion" and "dilation" [42]. Let the sets X and G represent an object and a structuring element, respectively. Let G_x be the translation of G so that its origin is at x. Then the erosion of X with respect to G is defined as the set of all points x such that G_x is a subset of X:

$$X \ominus G = \{x : G_x \subset X\} \tag{5.18}$$

and the dilation of X with respect to G is defined as the set of all points x such that X and G have a non-empty intersection:

$$X \oplus G = \{x : G_x \cap X \neq \phi\} \tag{5.19}$$

An illustration of erosion and dilation is given in Figure 5-4. This figure demonstrates the erosion is a shrinking operation and dilation is an expansion operation.

Erosion and dilation can be used to build more complicated morphological operations such as *open, close, etc.*

The opening of X with respect to G is defined as a cascade of erosion and dilation operations on X with respect to G:

$$X \bigcirc G = (X \ominus G) \oplus G \tag{5.20}$$

Closing is the dual of opening:

$$X \odot G = (X \oplus G) \ominus G \tag{5.21}$$

The open operation has the property of smoothing contours and suppressing small islands, whereas the close operation has the property of blocking narrow channels. If we introduce opening-closing operation, it makes a mixed result of both two effects. Therefore, three operators will change original details much when the picture contains delicate edges. In other words, if we compare detailed original images with operationed images, there would be higher distortion for the images which have originally more delicate edges. So, we then decide activity levels based on the measurement.

Figure 5-4 depicts results of morphological operations. Figure 5-4 (a) is original high detail image. If the closing operation is done on high quality input, signal details are gone as shown in Figure 5-4 (b). And if the opening operation is done on high quality input, signal details are gone once again as shown in Figure 5-4 (c). In addition, if the opening-closing operation is done on high quality input, signal details are mixed gone as shown in Figure 5-4 (d).

To measure details of shape data, we provide the following measure:

$$AM_{shape} = \sum_{j=0}^{15}\sum_{i=0}^{15}(\mid x(i,j) - x_1(i,j) \mid + \mid x(i,j) - x_2(i,j) \mid + \mid x(i,j) - x_3(i,j) \mid). \tag{5.22}$$

Here, $x_1(i,j)$, $x_2(i,j)$, and $x_3(i,j)$ are morphological processed pixel value of that BAB while $x(i,j)$ is each pixel value of a BAB. We define index 1, 2, and

3 as closing, opening, and opening and closing operations respectively. If shape edges are delicate, the resultant measurement is relatively high. If shape edges are simple, on the other hand, the resultant measurement is relatively low. Therefore, the measurement presents delicacy of edges.

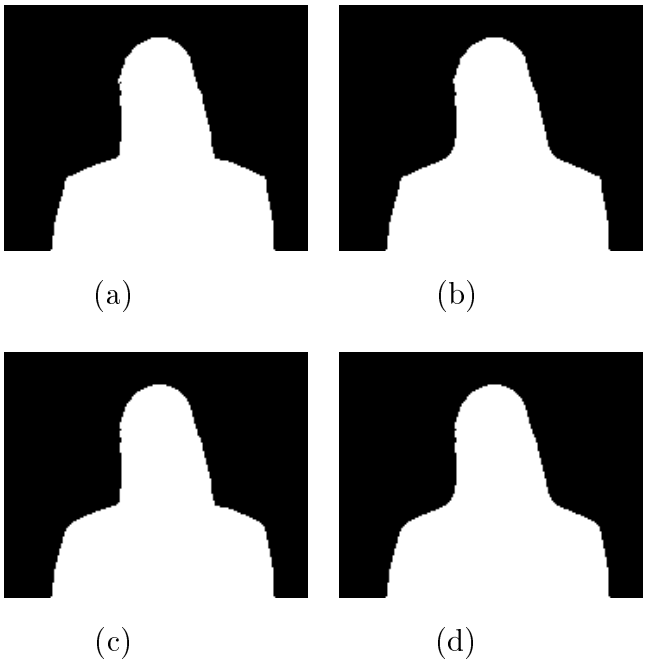


Figure 5-4: Results of morphological operation (BAB-based) : (a) decoded without restoration, (b) decoded with closing operation, (c) decoded with opening operation, (d) decoded with opening-closing operation.

5.4.4 Rough Bound of Buffer Fluctuation

Since the new rate control scheme deals with parameters in virtual buffer, it is important to predict characteristics of corresponding actual buffer. In particular, the upper bound of deviation D_U and the lower bound of deviation D_L are of interest. In optimal buffer compression approach usually steady state buffer occupancy, B_{sat}, is saturated just under maximum buffer size, B_{virmax}, with rate control algorithm operating.

Assume that the rate control procedure finds a stationary operating point where steady state buffer occupancy is B_{sat}. And also, we assume that blocks are classified into two groups based on activity: active and inactive, and that frames are done so into two groups based on monotonicity: frame I and others. Furthermore, let us say that total active area is αA in a frame where $\alpha \leq 1$ and that input video is frame type (or rectangular shape VOP). A I frame generates the highest buffer occupancy, and the accumulated bits in an actual buffer are the difference of going out bits in the virtual buffer and going out bits in the actual buffer. Since spatial modulation nullify the effect at the end of a frame, the maximum accumulation at the end of I frames only depends on temporal modulation. Therefore, the accumulated bits are $A\gamma^I r - Ar = A(\gamma^I - 1)r$ where A is the size of a frame. Note that this might not be the maximum occupancy sometimes; if the next frame P (i.e., a B frame is generating less bits than a P frame) generates more bits than going out bits in the actual buffer in αA area, that point is the maximum buffer occupancy (i.e., since there are two groups, the maximum buffer occupancy happens at the case of the first αA blocks being active). Note that we consider two temporal groups and two spatial groups. Therefore, $\gamma^I \geq 1$, $\gamma^0 \leq 1$, $\gamma_{act} \geq 1$, and $\gamma_0 \leq 1$. From the previous discussion the upper buffer of deviation is $max\{A(\gamma^I - 1)r, A(\gamma^I - 1)r + \alpha A(\gamma^0 \gamma_{act} - 1)r\}$.

A B frame generates the lowest buffer occupancy, and the accumulated bits in an actual buffer are the difference of removed bits in the virtual buffer and removed bits in the actual buffer. Since spatial modulation nullify the effect at the end of a frame, the maximum accumulation at the end of B frames only depends on temporal modulation as we discussed above. Therefore, the over-removed bits are $Ar - A\gamma^0 r = A(1 - \gamma^0)r$. Note that this might not be the minimum occupancy usually; if the next frame B (i.e., a B frame is generating less bits than a P frame) generates less bits than removed bits in the actual buffer in αA area, that point is

the minimum buffer occupancy (i.e., the minimum buffer occupancy happens at the case of the first $(1-\alpha)A$ blocks being inactive). From the previous discussion the lower buffer of deviation is $max\{A(1-\gamma_0)r, A(1-\gamma_0)r + (1-\alpha)A(1-\gamma^0\gamma_0)r\}$.

If we assume that $B_{sat} \approx B_{virmax}$ in the optimal buffered compression algorithm, the maximum size of the actual buffer should be larger than $B_{virmax} + max\{A(\gamma^I - 1)r, A(\gamma^I - 1)r + \alpha A(\gamma^0\gamma_{act} - 1)r\}$ and the absolute size of the encoder buffer should be larger than $max\{A(\gamma^I-1)r, A(\gamma^I-1)r+\alpha A(\gamma^0\gamma_{act}-1)r\} + max\{A(1-\gamma_0)r, A(1-\gamma_0)r + (1-\alpha)A(1-\gamma^0\gamma_0)r\}$.

We assumed above that the rate control procedure finds a stationary operating point where steady state buffer occupancy is B_{sat}. Note that stationary operating point can be obtained only when there are enough number of quantizers (i.e., 31 is usually enough such as in MPEG-1/2) and that outgoing buffer rate r is reasonable. For example, if QP is limited in a small number of quantizers, there are certain cases where buffer occupancy can not be lowered less than B_{virmax}. As another example, if buffer rate is extremely small (say, 1 bit per block), QP values cannot make any stationary operating point. If stationary operating point assumption is broken as previous examples, the maximum buffer size of the actual encoder ought to be larger than $B_{virmax} + max\{A(\gamma^I - 1)r, A(\gamma^I - 1)r + \alpha A(\gamma^0\gamma_{act} - 1)r\} + B_{abn}$ where B_{abn} is an extra contribution (i.e., overshoot) to the buffer occupancy due to abnormal buffer regulation.

5.4.5 A Fast Approximation: Buffer Suppression and Greedy Procedure

Optimal path can be found by an exhaustive search. If any of QP values at a certain block stage cannot avoid the buffer overflow, one step previous macroblock should be recalculated in the branch for the same computation in order to find the optimal path. In theory, the buffer size and past data should be kept until the entire coding

procedure ends, because there is overflow possibility in any future blocks. This basically means that the coding delay is extremely long, thus making the algorithm impractical.

To make the algorithm feasible and make computational demands realistic is the purpose of an approximation algorithm. The aspects of fast approximation algorithms about optimal buffered compression point in two directions: *Lagrange multiplier* method and *greedy* method as shown in recent studies [60, 83, 83]. The greedy method was reported to achieve a nearly optimal performance with much relaxed burden of computational complexity in [83]. The proposed greedy method in [83] gives much less computational complexity than the Lagrange multiplier method in [60].

We try greedy procedure approach based on following two reasons: firstly, the greedy procedure shows a very close performance to optimal algorithm as shown in [59, 60, 83, 72, 71]. Therefore, excessive computation is not necessary to improve a very small PSNR increase for most of applications. Secondly, strict optimality is not the purpose any more since activity-assistancy as a "human visual factor" was introduced in the chapter. In other words, computational simplexity and approximated optimal performance can be achieved by aforementioned greedy approach. To do so means that we choose a quantizer which gives minimum distortion at each block stage. This decision rule will make the additive objective function not only a local minimum but also a nearly global minimum.

On top of that, we propose a "buffer suppression algorithm," which has no backup, as an approximation algorithm to make computational demands more realistic. The idea of this algorithm is to take a virtual buffer size B_{virmax} in the virtual buffer. And if there is no way to maintain the current buffer size under the B_{virmax} ($\leq B_{max}$), then we allow the algorithm to generate more bits thus making the buffer

size B_{virmax} exceeded. Note that in this case we don't need to go backward for the optimal path. Since the algorithm assigns the worst QP value forcingly after buffer saturation, the content size will go under B_{virmax} quickly. One nice property of this algorithm is that the computational complexity is much lower than that of greedy algorithm (any backward allowed). We refer readers to [49] for further explanation.

Obviously, an actual system will operate with a regular, unmodulated output buffer which will be emptied at the constant rate r. Consequently, both two Equations (5.13) and (5.15) have to be tracked in order to avoid overflow or underflow (the latter is of importance only if alternate synchronization mechanisms are not available). The modulated virtual buffer is used to drive the evolution of QP via the greedy regulation $f_{greedy}(.)$, while the actual buffer is monitored to force block skipping in cases of overflow. When the virtual buffer overflows, no particular action is taken and QP is assigned its maximum value (i.e., buffer suppression). Note that since underflow is not included in this definition, given that the objective of minimizing distortion would tend to increase the coding bit rate and thus automatically penalize underflow.

In summary, buffer rate modulation allows one to force the rate control algorithm to spend a specified number of bits more in active regions/frames. It is important to note that techniques can be applied in general to any rate control scheme; modulating parameters which are considered external and fixed to the rate control algorithm they would simply operate it at a number of different target bit rates depending on the image area being coded. Furthermore, complete decoder compatibility is maintained as the region location information does not have to be transmitted to the decoder, and is only used in the encoder.

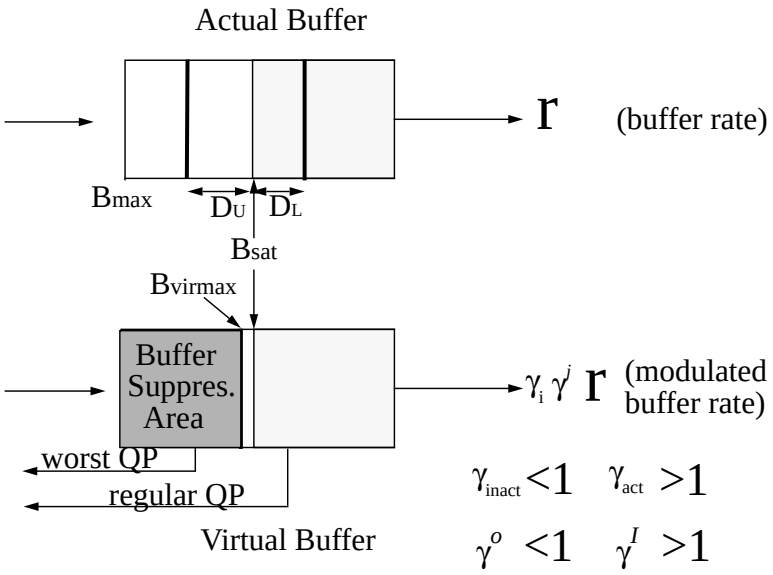


Figure 5-5: Buffer suppression and deviation.

5.4.6 Examples

Practical systems don't need to be used to prove the validity of the proposed scheme because the problem itself is exactly a resource allocation problem in operations research. In this section, we consider very artificial examples to compare performance boundaries of the proposed method and the optimal buffered compression.

We consider a quantization system with 5 different quantization parameters {1, 2, 3, 4, 5} which generates the following rate and the distortion about the type of "block 1" and the type of "block 2."

For the type of block 1,

$$r(1) = 61 \; and \; d(1) = 0.21, \tag{5.23}$$

$$r(2) = 51 \; and \; d(2) = 0.31,$$

$$r(3) = 41 \; and \; d(3) = 0.41,$$

$$r(4) = 31 \; and \; d(4) = 0.51,$$

$$r(5) = 21 \; and \; d(5) = 0.61.$$

For the type of block 2,

$$r(1) = 50 \; and \; d(1) = 0.10, \tag{5.24}$$

$$r(2) = 40 \; and \; d(2) = 0.20,$$

$$r(3) = 30 \; and \; d(3) = 0.30,$$

$$r(4) = 20 \; and \; d(4) = 0.40,$$

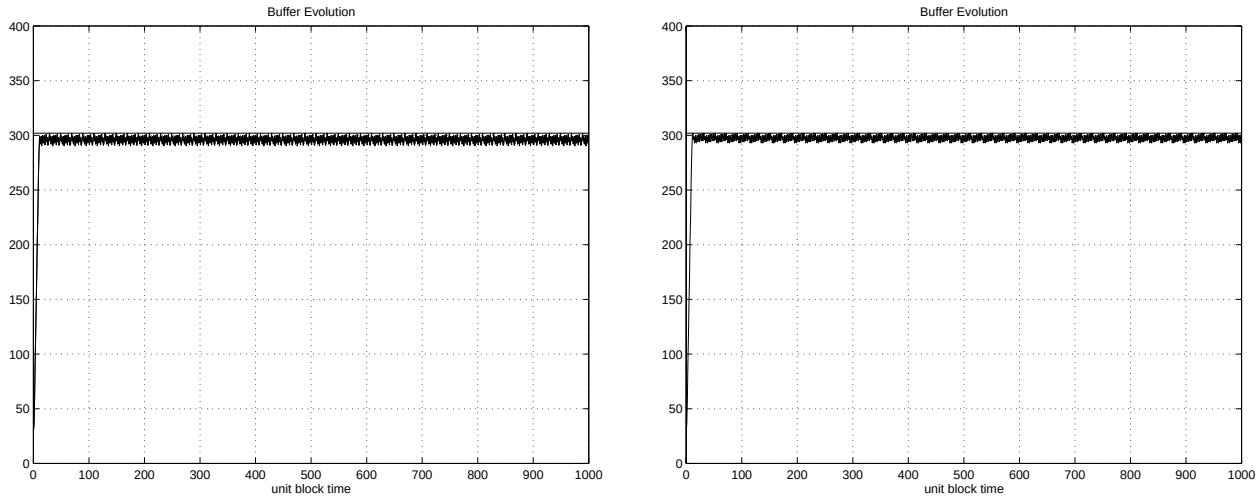
$$r(5) = 10 \; and \; d(5) = 0.50.$$

We assume that a frame is composed of two blocks where the first block is "block 1" and the second block is "block 2," and that such 500 frames are concatenated continuously. Therefore, the total number of blocks is 1000 and the total number of frames is 500. As we see in the rate distortion characteristics, block 1 is more active than block 2 as shown in Equation 5.23 and Equation 5.24. Figures 5-6-5-8 show buffer occupancy comparisons between the proposed method and the optimal buffered compression method. As we expected, our proposed method shows more fluctuation in the buffer by which more bits are given to active areas (i.e., block 1), while less bits are given to inactive areas (i.e., block 2). On the other hand, the optimal buffered compression method shows quite uniform fluctuation because it doesn't consider activity of each block. Note that the maximum buffer size of the optimal buffered compression is taken based on maximum occupancy of the proposed method for a fair comparison. Since a virtually maximum buffer is taken for the proposed method, an actual maximum buffer occupancy usually is over the

Distortion with block number	dist at 997	dist at 998	dist at 999	dist at 1000
V and O ($B_{max} = 302$)	435.79	436.19	436.70	437.00
Proposed ($B_{max} = 302$)	435.79	436.19	436.70	437.00
V and O ($B_{max} = 308$)	435.79	436.09	436.60	437.00
Proposed ($B_{max} = 308$)	435.69	436.19	436.60	437.00
V and O ($B_{max} = 313$)	435.69	435.99	436.60	436.90
Proposed ($B_{max} = 313$)	435.69	436.19	436.60	437.00

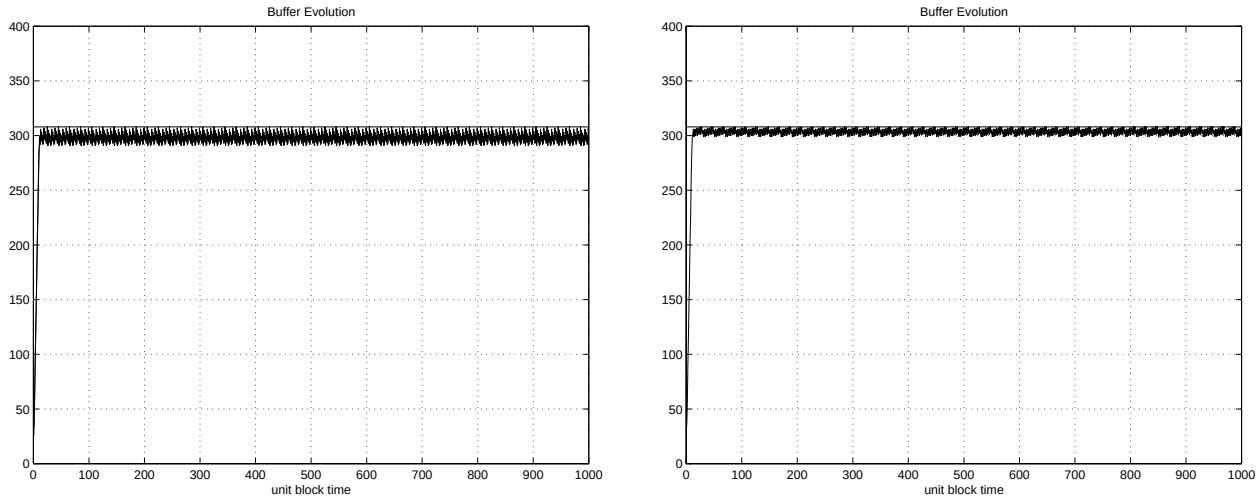
Table 5.1: Accumulated distortion with block number (50% of active area and γ_{active}=1.1, 1.3, and 1.5) respectively.

virtual maximum buffer size. Therefore the actual maximum buffer occupancy is given to the optimal buffered compression method as the maximum buffer size. In other words, the order of experiment for a fair comparison is: first, the proposed method is tested, and then, the optimal buffered compression is tested based on maximum buffer occupancy information of the proposed method. Note that maximum buffer sizes (i.e., B_{max}) are equally set up "pairwise" in the Table 5.1 so that a fair comparison can be performed. The table basically shows that the accumulated distortions have almost the same values in the proposed method and the optimal buffered compression after 500 frames (i.e., steady-state). This means that our proposed method can assign different bit budgets to different activity blocks with a similar achievement of the performance of the optimal buffered compression. Note that this problem is set up in a simple manner to show the property clearly. In practical systems blocks of the same rate distortion property will not be repeated throughout frames. The similar property of above simple case about rate distortion performance can be expected in more complicated systems as well.



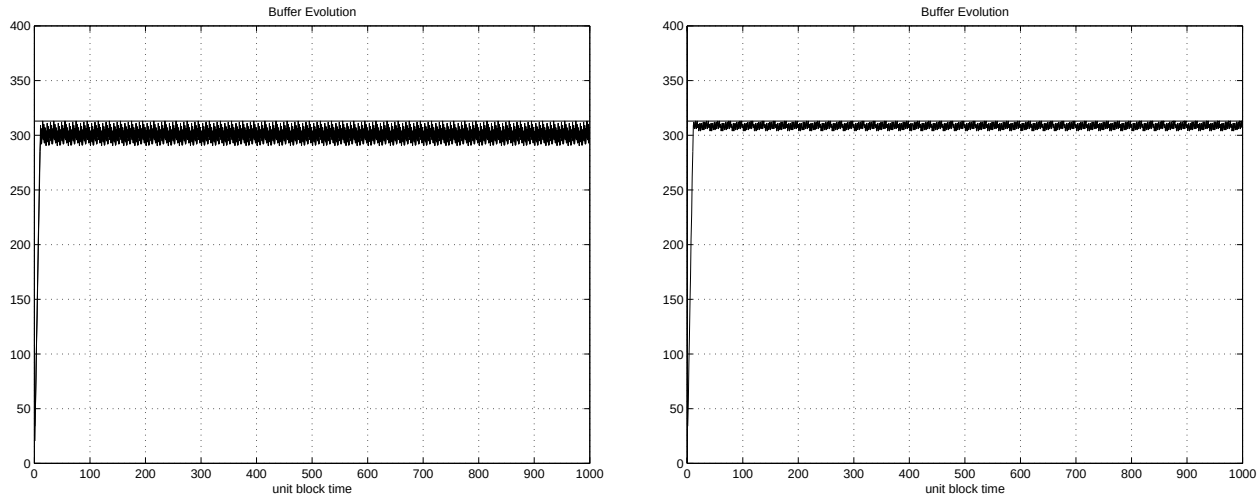
(a) (b)

Figure 5-6: Buffer occupancy comparisons between activity-assisted optimal buffered compression and optimal buffered compression for example 1): (a) $r = 27$ for $\gamma_{active} = 1.1$ and 50% of active area, (b) $r = 27$ Ortega and Vetterli



(a) (b)

Figure 5-7: Buffer occupancy comparisons between activity-assisted optimal buffered compression and optimal buffered compression for example 2): (a) $r = 27$ for $\gamma_{active} = 1.3$ and 50% of active area, (b) $r = 27$ Ortega and Vetterli



(a) (b)

Figure 5-8: Buffer occupancy comparisons between activity-assisted optimal buffered compression and optimal buffered compression for example 3): (a) $r = 27$ for $\gamma_{active} = 1.5$ and 50% of active area, (b) $r = 27$ Ortega and Vetterli

5.5. STBRM Technique for MPEG-4 Video Coding

The video part of the MPEG-4 specification provides two core components that are not available in any other standards. The first component is object-based representation, as opposed to a pixel or frame based representation, that allows scene composition and interactivity with content. The representation of objects is composed of Video Objects (VO), and Video Object Planes (VOP). The VOs correspond to entities in the bitstream that the user can access and manipulate (e.g., cut and paste), are independently coded, and saved on separate bitstreams. Instances of a VO in a given time are called VOPs. The second component is a certain degree of flexibility in the design of a system that leads to an open, yet efficient, standard. The notion of a toolbox is used to allow a flexible design of coding algorithms based on the requirements of specific applications [30].

5.5.1 Syntactic Restriction on Quantization in MPEG-4 Video Coding

In MPEG-1/2, the range of QP is from 1 to 31. To find the optimal path requires a lot of computation. Even with the simplied method, to check every QP's rate-distortion (31 QP values) is still a computational intensive procedure.

MPEG-4 video coding is applied independently to each object. Therefore, the quality variance inside one object is designed to be minimal. As a result, for MPEG-4 texture coding, only 5 different values of QP are allowed inside one VOP (i.e., DQUANT: -2, -1, 0, +1, +2) when a VOP level QP (i.e., 1, ..., 31) is given. Therefore, for the optimal path, 31 different QPs are tried at VOP level while only 5 DQUANTs are tried. This is once again a bit computational complex. In this section, we propose a fast approximation for VOP QP computation. To find the optimal path for 5 different QPs is not that harsh in terms of computation given a VOP QP.

In recent research [49], authors have showed that MPEG-4 shape coding can be translated into the problem of selecting virtual quantizer with respect to coding units. Since the translated problem only contains 3 different virtual quantizer (i.e., 1, 1/2, 1/4), once again applying the simplified method is not computational burden. Overall, the techniques developed in this chapter are more proper to MPEG-4 video coding.

We should note that there is a extra contribution B_{abn} to the buffer occupancy because the number of quantizer is too small (i.e., texture: 5, shape: 3). In particular, bits cannot be suppressed under B_{virmax} in I frames since generated bits for blocks in the frames are beyond controllable region owing to only 5 quantizers. If the number of incoming bits is the same as r, there would be no buffer increase (i.e., $B_{abn} \approx 0$). If the buffer increase is k portion of r in each block, the overshoot buffer size is approximated at the end of the frame by $B_{abn} \approx kAr$. Especially for I frames,

such overshoot in buffer happens. Experiments say that in MPEG-4 texture coding a k factor ranges from 2 to 3 at I frames. On the other hand, in MPEG-4 shape coding $k = 0$ can be seen (i.e., no serious overshoot happens in the actual buffer due to simplicity of shape data). Note that this doesn't do that harm in terms of algorithm operation. It only requires a bigger, but reasonable, size actual buffer to avoid unexpected overflow in the actual buffer. The relationship between the maximum size of virtual buffer and the maximum size of actual buffer is relative. Therefore, one parameter can be taken independently unless lower fluctuation is reasonably considered. We will discuss this more in the next section.

5.5.2 Spatial Activity-Assisted MPEG-4 Shape Coding

Since the MPEG-4 standard only specifies the decoder (the syntax of a compliant bitstream), modules in encoders such as rate control can be arbitrarily designed. In this section, we only provide the definition of optimal buffered compression algorithm, which was introduced in [49], with buffer rate modulation.

The current MPEG-4 binary shape coding process consists of two sub-steps. First, distortion is introduced in the size conversion process by the way shape data are scaled down and up, based on $alpha\_th$ value. Second, CAE is applied. Since arithmetic coding itself is lossless, we view the size conversion process as a "virtual quantizer."

In the size conversion process, distortion can possibly be introduced in two levels; VOP level size conversion and following BAB level size conversion. If this is thought of as a "virtual quantizer," we can formulate it for shape coding. Note that in this formulation the $alpha\_th$ value is replaced by $(VOP(BAB(i)).CR, BAB(i).CR,$ $coding\_mode)$ conceptually, thus making $alpha\_th$ removed. Therefore, the problem formulation can be given as follows.

Shape Coding Formulation : (Integer Programming)

Let x be $\{x(1), x(2), ..., x(N)\}$ where $x(i) = (VOP(BAB(i)).CR, BAB(i).CR,$ $coding\_mode)$.

The problem is to find the mapping x that solves

$$\min_{x(i), i=0,..,N} (\sum_{i=1}^{N} d_{ix(i)}), \tag{5.25}$$

subject to

$$B_i \leq B_{virmax}, \forall i = 1, ..., N$$

where B_{max} is the maximum buffer size. In the above expressions, $BAB(i)$ means i-th BAB, and $VOP(BAB(i))$ means VOP in which $BAB(i)$ is.

Figure 4-6 shows all possible paths of encoding CRs from the first BAB to the last BAB. A certain path in the figure implies a quality and rate of a specific output shape data sequence. Each branch has two values involved: VOP.CR and BAB.CR. In this chapter, we only consider VOP.CR=1. The greedy procedure is to decide a certain path which gives local minimum distortion under B_{virmax} as discussed in 5.4.5. If we consider that the distortion is independent and additive with respect to each block, such a local minimum path gives a nearly global minimum path. It is important to understand that distortion measure is independent even when the INTER mode is selected. Note that the quality of a BAB with INTER mode is not affected from the best match BAB in the previous frame. A certain area is used for updating probability table to apply CAE coding to efficiently encode BABs in a lossless manner.

The rate control operation of the encoder by the greedy procedure is represented by:

$$CR_i = f_{greedy}(B_{i-1}, r). \tag{5.26}$$

Note that no temporal modulation is necessary since there is no dependency among VOPs.

The only difference, compared with [49] is that in order to convert rate control regulation to provide location-dependent we use the concept of buffer rate modulation which was introduced in our previous research [33]. The idea is to modulate the target rate so that more bits are spent for BABs that are inside activity regions, and less for BABs that are not. The rate r now becomes location dependent, and is given by

$$r_i = \gamma_{S(i)} r, \tag{5.27}$$

where the region index function $S(i)$ associates the position of BAB i with the region in which it belongs to in a VOP. The buffer evolution can now be described by

$$B_i = B_{i-1} + c(i) - r_i, \tag{5.28}$$

where the number of bits spent $c(i)$ is now region-dependent.

Note that in general the γ_i do not have to be all positive. Although a negative modulated buffer rate is somewhat surprising and counterintuitive (as it means that the buffer is actually receiving bits from the channel instead of always transmitting them), it helps to severely constrain bit allocation in the exterior, particularly in pictures which are difficult to code.

We refer readers to [49] for more detailed explanation.

5.5.3 Spatio-Temporal Activity-Assisted MPEG-4 Texture Coding

The texture coding technique in MPEG-4 video is similar in several aspects to those used in other state-of-the-art standards [41, 40, 53, 39, 27] . The I VOP MBs and the residual MBs after motion compensation are coded one after the other using a

discrete cosine transform (DCT) scheme. DCT is performed separately on each of the luminance and chrominance planes in a given MB, totaling 6 blocks of size 8×8. The MBs that are completely outside of the VOP are not coded at all. Those MBs that lie completely inside the VOP are coded using a conventional DCT scheme. The 8×8 blocks of MBs lying partially on the VOP are first padded using repetitive padding as for the motion estimation, with difference that for residue blocks, the region outside of the VOP within the blocks are padded with zero values. If all the pixels in an 8×8 block are transparent, their values are replaced by zero. These blocks are then coded in a manner identical to the blocks inside of the VOP [30].

The DCT coefficients are quantized, zig-zag scanned, and entropy coded by a run-length Huffman method. The quantization step can be based on one of the two following two alternatives: H.263 and MPEG. For higher compression coefficiency, the DC as well as the first row and first column AC coefficients in the intra-coded MBs can be differentially coded using an adaptive predictive method. The prediction value used is chosen as the corresponding value of the block above or immediately to the left of the current block. This adaptive selection is based on comparison of horizontal and vertical gradients of corresponding DC values of already coded neighboring blocks [30].

For MPEG-4 texture coding, only 5 different values of QP are allowed inside one I or P VOP (i.e., DQUANT: -2, -1, 0, +1, +2) while only 3 different values of QP are used for one B VOP (i.e., DBQUANT: -2, 0, +2). DQUANT and DBQUANT define changes in the value of the VOP_quantizer. Note that if 0 is selected in DQUANT nothing is written in the bitstream while if 0 is selected in DBQUANT code "0" is written in it. For VOP level QPs, one needs to monitor actual buffer maximum size to take an action at possible overflow. If overflow happens at a specific QP value, a rougher QP value is retried from the first MB of the same

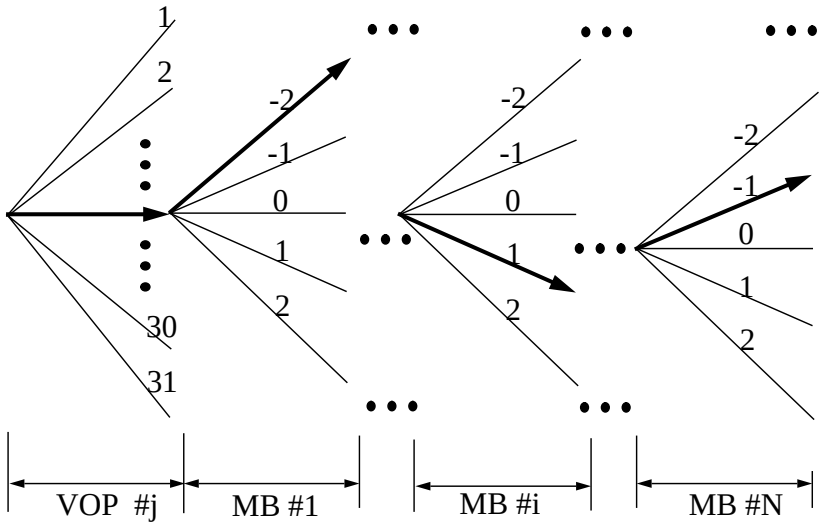


Figure 5-9: One path determines a quality and rate of output texture data (A Branch value means an QP value of each MB).

VOP. To obtain the optimal path, it is obvious that every value from 1 to 31 should be tried for VOP QP repeatedly. In other words, the VOP QP value of 2 is tried when the VOP QP value of 1 makes actual buffer overflow. If VOP QP value of 2 makes actual buffer overflow, the compression with the VOP QP value of 3 will be tested. This procedure will continue until no actual buffer overflow is happening. Since this is quite computational intensive, we propose an approximation algorithm. An estimation of the VOP QP is depicted in Figure 5-10. The idea is to follow exactly the same procedure of MB QP decision based on buffer evolution. To set up a buffer evolution model at the VOP level, outgoing bits (from the encoder buffer) and generated bits (into the encoder buffer) should be estimated. First, the amount of outgoing bits is estimated by area times unit r for each MB. Second, the estimation of generated bits is set by area times generated bits for the first MB. The basic assumption of this buffer evolution model is that the source is stationary (i.e., the average of generated bits over one VOP is the same to the generated bits for

first MB. The VOP QP is determined to provide the best quality given allowable buffer occupancy. We note that the deviation of generated bits (i.e., when above assumption is broken) can be absorbed by MB level QP (i.e., -2, -1, 0, +1, +2).

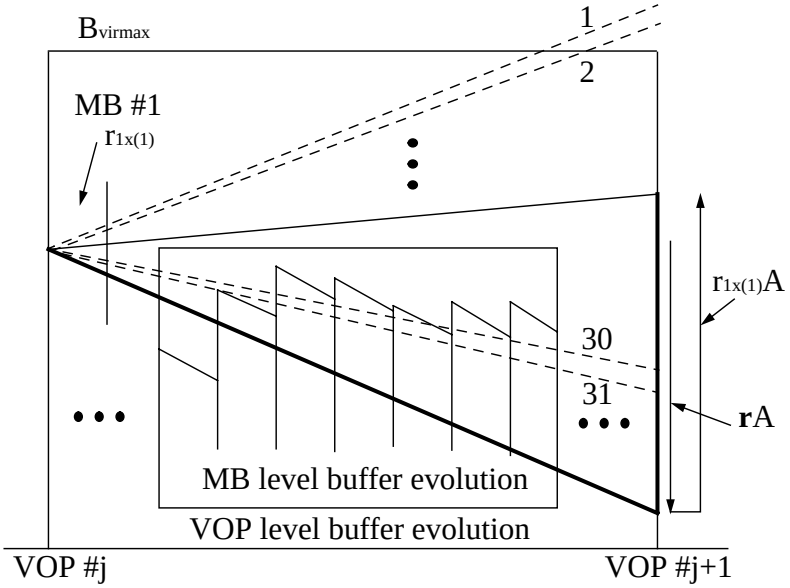


Figure 5-10: A fast approximation with an estimated VOP QP.

The MPEG-4 texture doesn't require the problem redefinition in (5.16). Therefore, we refer readers to original papers of trellis-based buffered compression [60] for further discussion.

The rate control operation of the encoder by the greedy procedure is represented by:

$$QP_n = f_{greedy}(B_{n-1}, r) \tag{5.29}$$

The idea is to modulate the target rate so that more bits are spent for MBs that are inside activity regions, and less for MBs that are not. The rate r now becomes

location dependent, and is given by

$$r_n = \gamma_{S(n)} \gamma^{T(n)} r, \tag{5.30}$$

where the region index function $S(n)$ and $T(n)$ associates the position of MB n with the region in which it belongs to in a VOP. The buffer evolution can now be described by

$$B_n = B_{n-1} + c(n) - r_n, \tag{5.31}$$

where the number of bits spent $c(n)$ is now region-dependent.

5.6. Experimental Results

The sequences in QCIF format referred to as "Akiyo" and "Children" were processed according to the version 8.0 of the MPEG-4 Verification Model. Note that we used our own implementation of a shape encoder and decoder, including specific implementations for size conversion and all CAE coding modes. For texture coding part, we modified a public domain code (Momusys) of the version 8.0 of MPEG-4 Verification Model. The simulations are done separately for shape and texture codings because we have two different implementations. For example, shape coder doesn't interact directly with the texture coder. Therefore, in our shape experiment all necessary information about texture motion vectors were saved in a temporary file at the encoder, and retrieved from it at the decoder.

The experimental results of shape coding are in Figures 5-12- 5-13. The channel rate is the average number of bits we use for each BAB block. The steady state quality is strongly connected with the channel rate r. Figures 5-11 and 5-12 show the decoded outputs of Akiyo at $r = 6$. The difference is that the first one is the result through a method of Ortega and Vetterli while the second one is through that

of ours. We can see the hair of left part in our method while there is no such shape in results of Oterga and Vetterli. On the other hand, in 2nd frame of Figure 5-12 there is an error in the right-most part of the head. This is happening because too many bits are used for delicate parts of the image; thus, bits are not enough for even smooth areas. This makes overall distortion increase. Figure 5-15 (a) depicts such a characteristics. For relatively simple shapes, Ortega and Vetterli's results are usually better.

Figures 5-13 and 5-14 show the decoded outputs of Children at $r = 12$. The difference is that the first one is the result through a method of Ortega and Vetterli while the second one is through that of ours. We can see the details of right person's legs in our method while there is no such shape in results of Ortega and Vetterli. For this sequence, the decoded shape of overall picture is better in our method than in that of Ortega and Vetterli. Figure 5-14 (b) depicts such a characteristics. For relatively complex shapes, our results are usually better. The most important implication of the graph is that the performance of the optimal buffer compression technique proposed by Ortega and Vetterli is achieved with subjectively enhanced visibility.

The experimental results of texture coding are in Figures 5-16- 5-19. The channel rate is the average number of bits we use for each MB. Figures 5-16 and 5-17 show the decoded outputs of Akiyo at $r = 50$. The difference is that the first one is the result through a method of Ortega and Vetterli while the second one is through that of ours. In this sequence, it is hard to tell the quality difference except high frequency areas such as eyes. We can see slight quality improvement around eye areas of Figure 5-17 (f), compared with Figure 5-16 (f). Since it is hard to tell, PSNR table is provided in Table 5.2. With parameter change, a slightly better result in terms of subjective quality can be seen in our method than that in Ortega

Sequence	Ortega and Vetterli	ours ($\gamma^I = 1.2$)	ours ($\gamma^I = 1.5$)	ours ($\gamma^I = 1.8$)
Akiyo ($r = 100$)	40.165897	40.493671	40.574001	40.642414
Children ($r = 100$)	29.389746	29.449678	29.167364	29.442242
Akiyo ($r = 50$)	36.687477	36.846695	36.974091	37.108620
Children ($r = 50$)	25.819033	25.949129	25.939619	25.834099

Table 5.2: Average Y-PSNR comparison for 50 frame duration (15% of active area and $\gamma_{active} = 1.7$) at various parameters.

and Vetterli.

Figures 5-18 and 5-19 show the decoded outputs of Children at $r = 50$. The difference is that the first one is the result through a method of Ortega and Vetterli while the second one is through that of ours. In this particular sequence, it is hard to tell the quality difference by bare eyes. PSNR table provided show a slight difference in PSNR. With parameter change, sometimes better results can be seen in our method, while other times Ortega and Vetterli's results can be better; usually better PSNR can be found through our metho d.

5.7. Concluding Remarks

A natural extension of the Model-Assisted Coding (MAC) is Activity-Assisted Coding (AAC); if the model in scenes is not obvious, activity can replace the role of models. In this chapter, we have introduced a technique to assign more bits to visually important unit blocks while preserving the same average bit rate with traditional rate control algorithms. We first derived temporal and spatial balance equations. And then Spatio-Temporal Buffer Rate Modulation (STBRM), a technique that selectively allocates bits to areas of different activity in terms of space and time for video sequences, was proposed. We presented as well a rate control mechanism which is proper to STBRM in the form of optimal buffered compression, providing a way to classify MBs (or Binary Alpha Blocks-BABs) into several

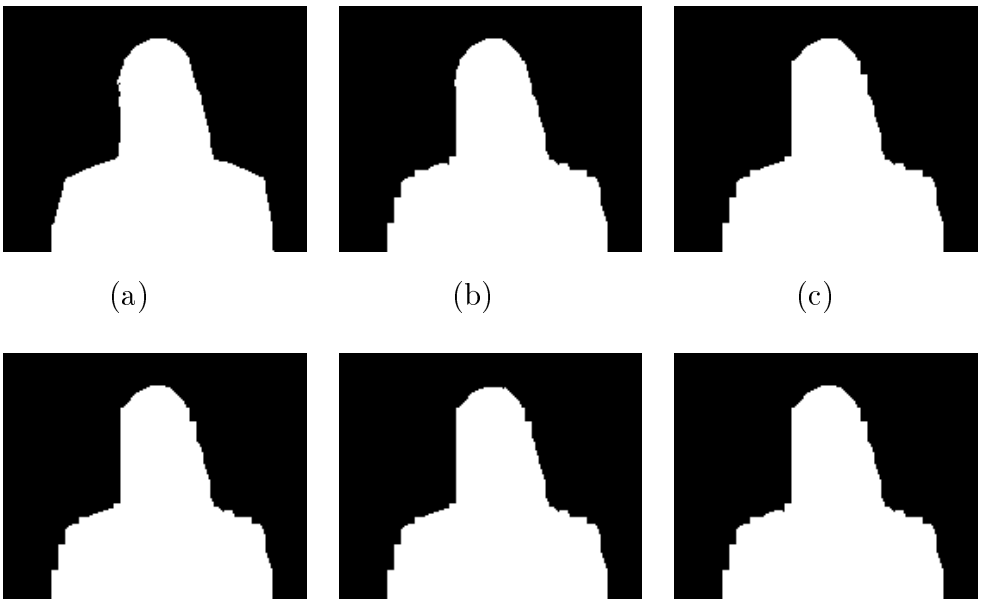


Figure 5-11: Decoded outputs (MPEG-4 shape decoder $B_{max} = 550$, $r = 6$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.

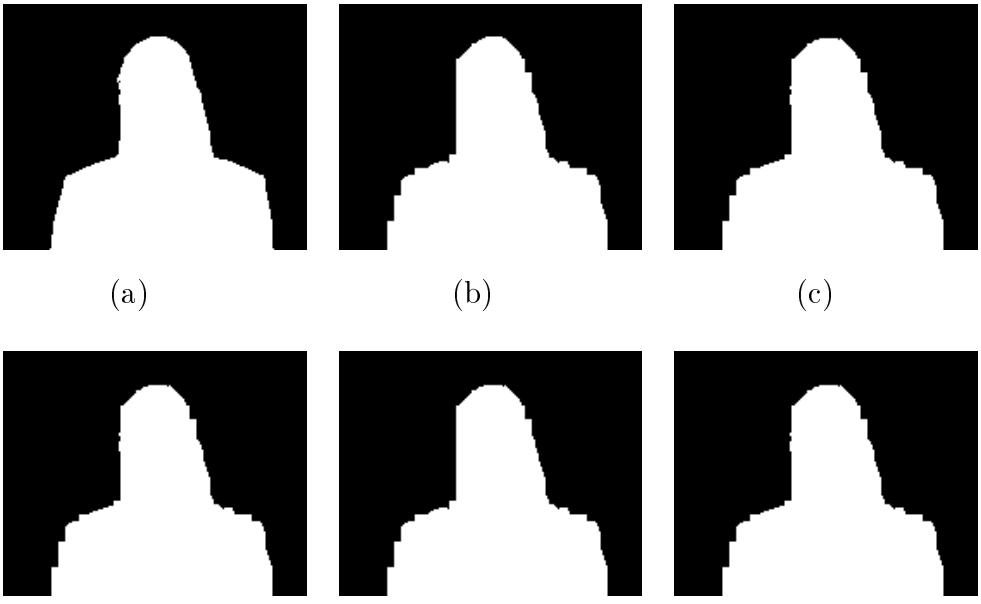
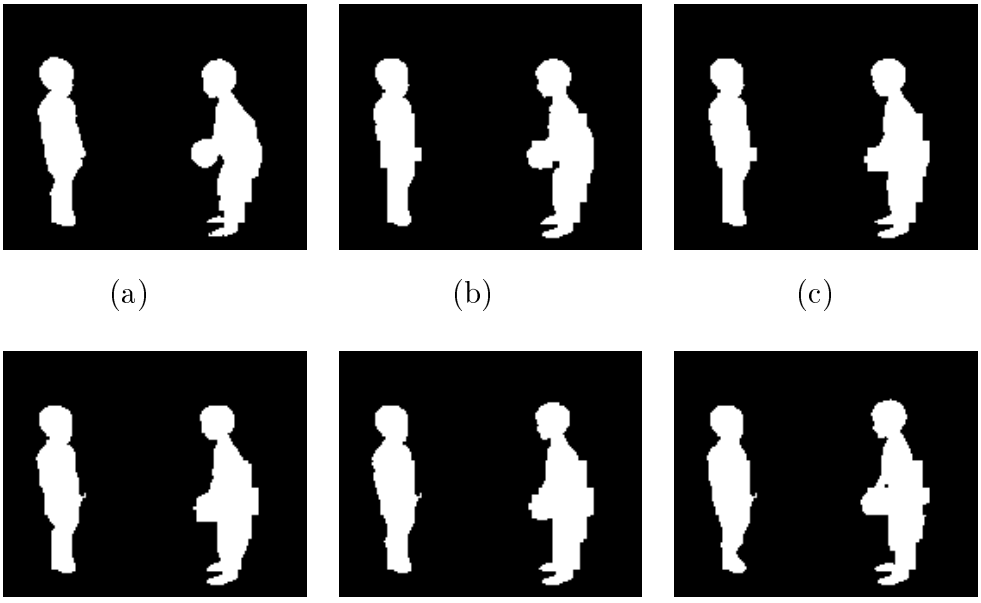


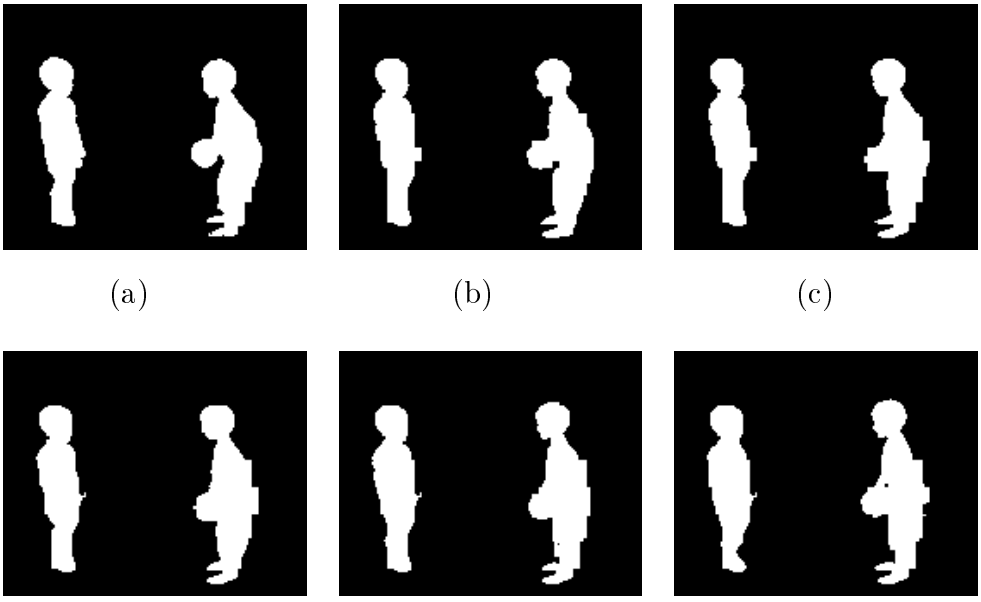
Figure 5-12: Decoded outputs (MPEG-4 shape decoder $B_{max} = 550$, $r = 6$, 15%, $\gamma_{active} = 1.2$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

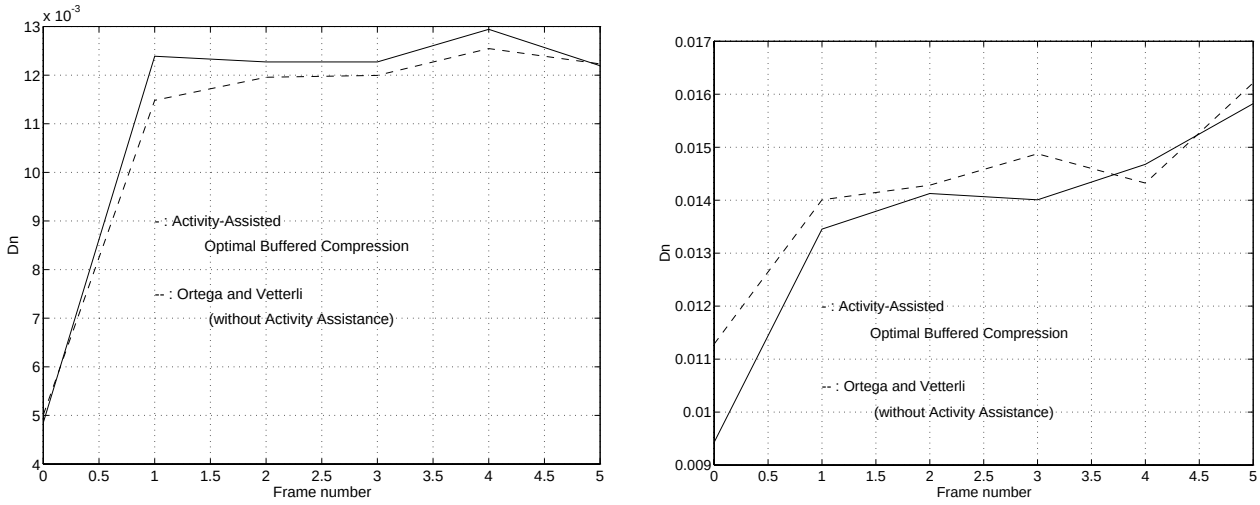
Figure 5-13: Decoded outputs (MPEG-4 shape decoder $B_{max} = 550$, $r = 12$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



(a) (b) (c)

(d) (e) (f)

Figure 5-14: Decoded outputs (MPEG-4 shape decoder $B_{max} = 550$, $r = 12$, 15%, $\gamma_{active} = 1.2$) : (a) 1st frame, (b) 2nd frame, (c) 3nd frame, (d) 4rd frame, (e) 5rd frame, (f) 6th frame.



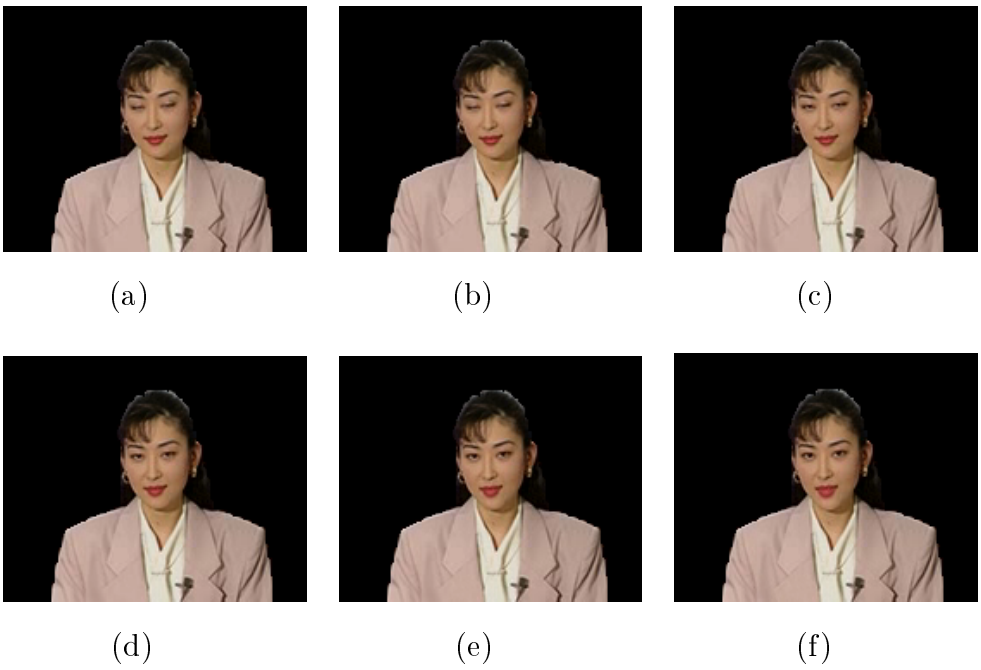
(a) (b)

Figure 5-15: Dn comparisons between activity-assisted optimal buffered compression and conventional buffered compression: (a) Akiyo ($r = 6$), (b) Children ($r = 12$) for $\gamma_{active} = 1.2$ and 15% of active area

(a) (b) (c)

(d) (e) (f)

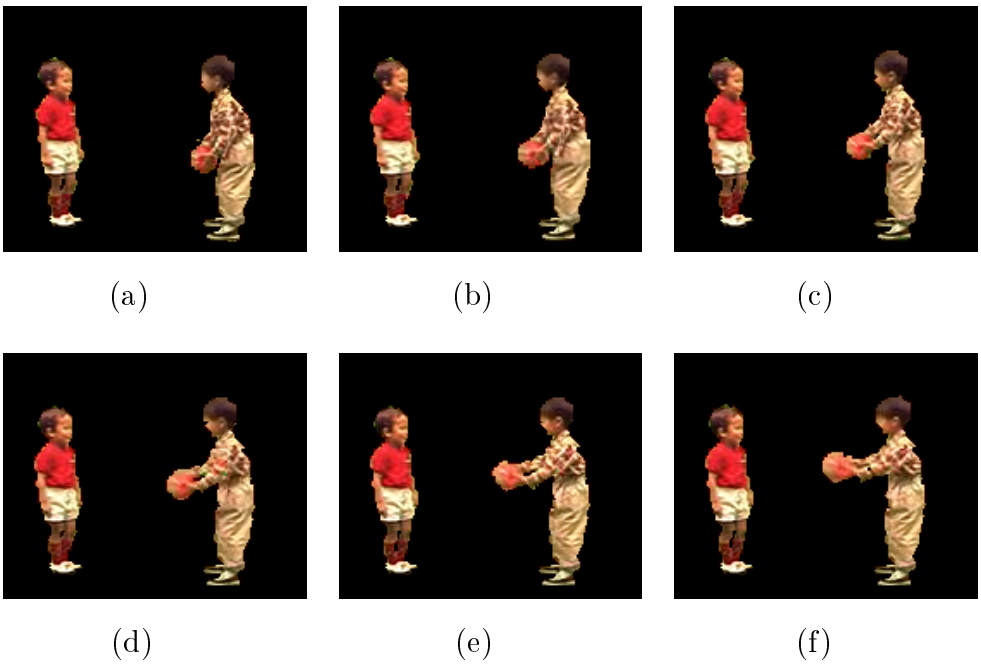
Figure 5-16: Decoded outputs of Akiyo sequence (MPEG-4 texture decoder $B_{max} = 40000$, $r = 50$) : (a) 5st frame, (b) 6nd frame, (c) 7nd frame, (d) 8rd frame, (e) 9rd frame, (f) 10th frame.



(a) (b) (c)

(d) (e) (f)

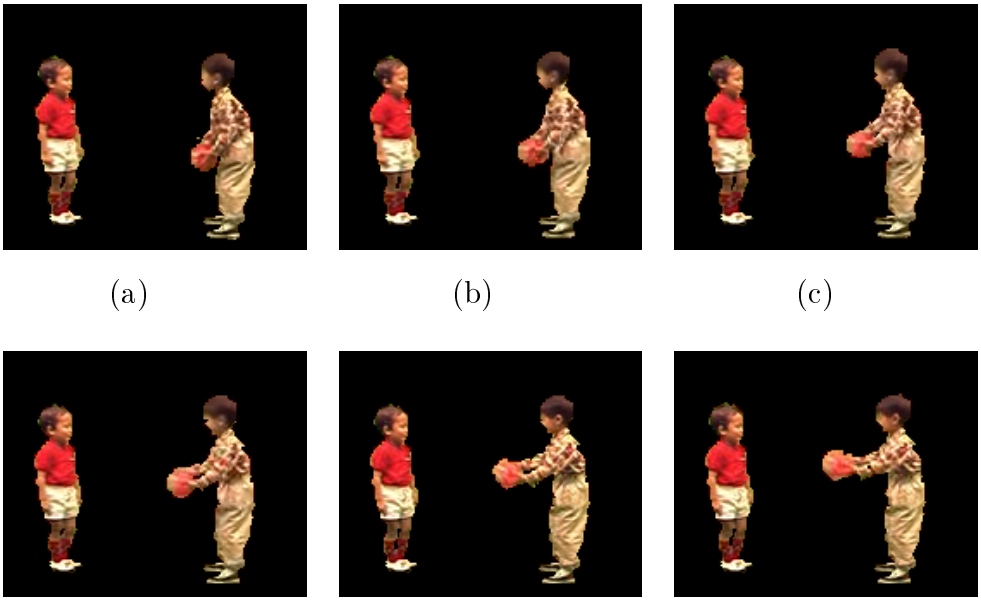
Figure 5-17: Decoded outputs of Akiyo sequence (MPEG-4 texture decoder $B_{max} = 40000$, $r = 50$, 15%, $\gamma_{active} = 1.8$) : (a) 5st frame, (b) 6nd frame, (c) 7nd frame, (d) 8rd frame, (e) 9rd frame, (f) 10th frame.



(a) (b) (c)

(d) (e) (f)

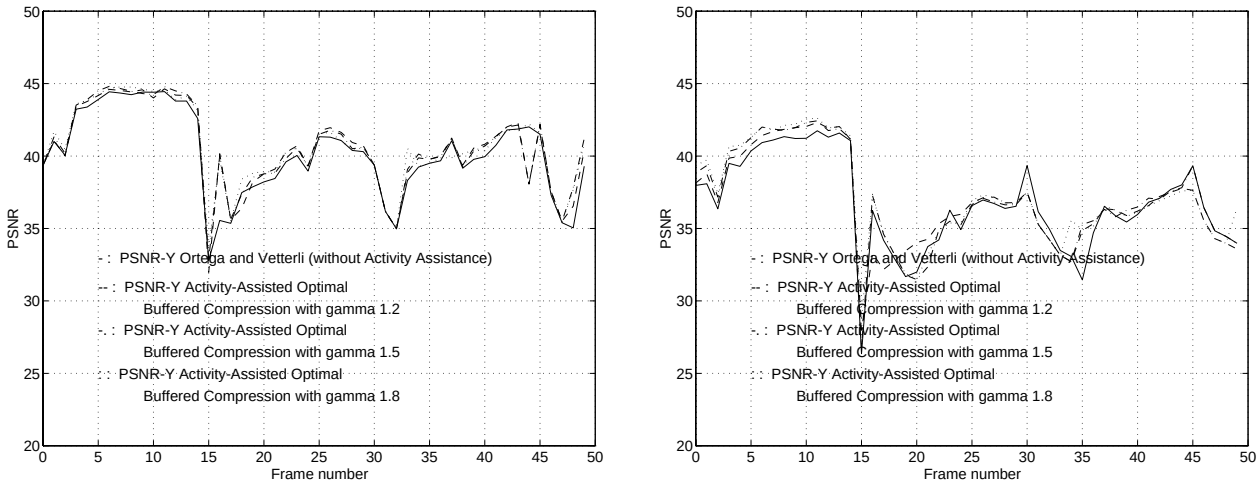
Figure 5-18: Decoded outputs of Children sequence (MPEG-4 texture decoder $B_{max} = 40000$, $r = 50$) : (a) 5st frame, (b) 6nd frame, (c) 7nd frame, (d) 8rd frame, (e) 9rd frame, (f) 10th frame.



(a) (b) (c)

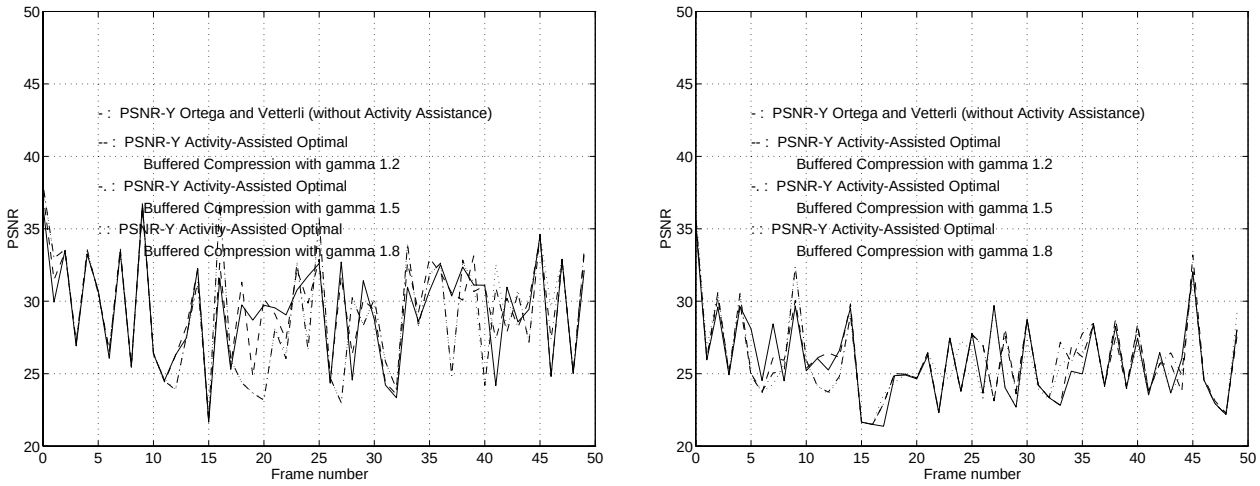
(d) (e) (f)

Figure 5-19: Decoded outputs of Children sequence (MPEG-4 texture decoder $B_{max} = 40000$, $r = 50$, 15%, $\gamma_{active} = 1.8$) : (a) 5st frame, (b) 6nd frame, (c) 7nd frame, (d) 8rd frame, (e) 9rd frame, (f) 10th frame.



(a) (b)

Figure 5-20: PSNR comparisons between activity-assisted optimal buffered compression and optimal buffered compression for Akiyo (arbitrary shaped texture): (a) $r = 100$, (b) $r = 50$ for $\gamma_{active} = 1.7$ and 15% of active area



(a) (b)

Figure 5-21: PSNR comparisons between activity-assisted optimal buffered compression and optimal buffered compression for Children (arbitrary shaped texture): (a) $r = 100$, (b) $r = 50$ for $\gamma_{active} = 1.7$ and 15% of active area

categories measuring "activity levels." The most important implication of the work was to show a potential way that the performance of the optimal buffered compression technique proposed by Ortega and Vetterli can be achieved with subjectively enhanced visibility under certain conditions. We have demonstrated this property by experimental results through MPEG-4 texture and shape codecs. The technique itself is, however, applicable to generic video coding standards such as MPEG-1, MPEG-2 and H.261.

Chapter 6

Conclusions and Future Work

6.1. Conclusions

We have introduced the concept of Spatio-Temporal Model-Assisted Compatible (STMAC) coding, a technique that selectively encodes different areas in ways that optimize according to human perception of space and motion. STMAC addresses the need to perform better bit allocation for very low bit rate coding (less than 64 Kbps). We were motivated by the fact that artifacts are systematically present at such rates; the overall user experience, however, can be significantly improved if we adjust coding to take into account human perception characteristics. Of particular importance on representations of the human face are the eye region, which must be rendered very finely, and the lip region, which must be replenished very frequently.

The first component of STMAC is a simple face model that can be used to identify robustly, in real-time, the location of important facial components (face, eyes, mouth). As we have shown, accuracy is not extremely important, for the model is only intended for use as a guide to bit allocation, not for rendering at the decoder. Equipped with this model, we can formulate a general rate control architecture by considering two balance equations: one operating in both space and time, and one operating only in space. These equations are driven by the total available bit rate,

and the desired emphasis that the encoder's designer wishes to place in the various model areas (both spatially and temporally). The equations provide specific bit budgets for each frame of a given period (e.g., 30 frames). These budgets are then used in an area-selective adaptation of the basic rate control algorithm (here TMN5).

Our experiments indeed demonstrate that the approach provides improved results compared with baseline H.263 TMN5 encoding. The perceptually important areas have improved in that the temporal resolution is much higher. Substantial improvement does come at the cost of a more degraded background and the increased possibility of observing motion discontinuities caused by the different temporal scales used in the various parts of the model. Yet, model breakdown prevention, which forces STMAC back to the traditional frame-based H.263, ensures that gross inaccuracies are avoided.

One of the key benefits of our approach is that it can be applied without any modification to the decoder, and is hence fully compatible with H.263. Compatibility is also important when model breakdown occurs – i.e., the scene content does not contain a head-and-shoulder sequence, forcing the model detection algorithm to fail – in which case, the encoder can always fall back to standard H.263 encoding, thus achieving acceptable performance.

We wish to point out that the application of the technique is not limited to H.263, but can be applied to any block-based transform coding scheme. The syntactic features of the scheme are important, however, as they can affect both the mechanism and the overhead associated with establishing different frame rates for the various model areas. H.263 is attractive in this respect because of its very low cost macroblock repetition option (COD flag); but, it is also somewhat restrictive because of the differential encoding of QP values that forces absolute differences between successive macroblocks to be up to 2.

We have also proposed an optimal buffered compression algorithm for shape coding as defined in the forthcoming MPEG-4 international standard. The current MPEG-4 shape coding scheme, which is totally new to the coding standards arena, consists of two sub-steps. In the first step, distortion is introduced by a size conversion process. Context-based arithmetic encoding is applied in the second step. Considering the size conversion process as a virtual quantizer, we formulated the buffer-constrained adaptive quantization problem for the shape encoder. We also developed an algorithm for the optimal solution under buffer constraints. For computational efficiency, we simplified the original algorithm to a fast approximation algorithm with additional consideration for marginal buffer reservation. Experimental results were given using an MPEG-4 shape coder, confirming that the shape data were compressed efficiently and without overflow.

A natural extension of Model-Assisted Coding (MAC) is Activity-Assisted Coding (AAC); should the model in a scene not be obvious, activity makes a useful replacement. Our technique assigns more bits to perceptually important areas based on activity, yet preserves the same average bit rate of existing rate control algorithms. To review, we first derived temporal and spatial balance equations, and then presented Spatio-Temporal Buffer Rate Modulation (STBRM). We also presented a rate control mechanism that is proper to STBRM in the form of optimal buffered compression, thus providing a way to classify MBs (or Binary Alpha Blocks-BABs) into several categories measuring activity levels. The most important implication of the work was to show one way that the performance of the optimal buffered compression technique proposed by Ortega and Vetterli can be achieved with subjectively enhanced visibility. We have demonstrated this property with experimental results through MPEG-4 texture and shape codecs. The technique itself, however, is applicable to generic video coding standards such as MPEG-1, MPEG-2

and H.261.

6.2. Future Work

6.2.1 Model-Assisted Coding with Encoder/Decoder Compatibility

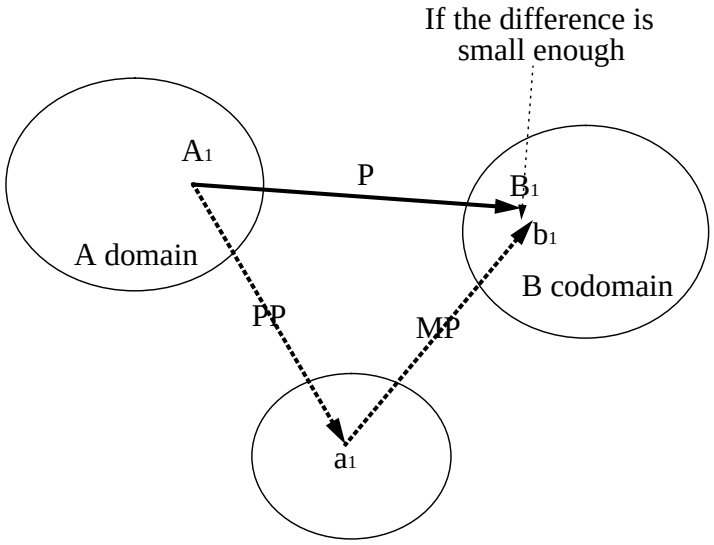


Figure 6-1: Processing approximation.

One merit of our STMAC very low bit rate coding system is its ability to use a traditional H.263 decoder without any modification, which is why we call it "compatible coding." Is there any way that we can achieve results very similar to STMAC's without any modification on the encoder and the decoder together? We think so, and we call this potential encoder/decoder set a "compatible codec."

We will say A is input domain, and B is the output codomain (solution domain) we want to achieve. We denote a general processing P as $A \xrightarrow{P} B$. If we do the processing P on one of elements of A, we can get one of elements of B. For example, $B_1 = P(A_1)$.

To avoid processing difficulty (P), sometimes it could be possible to change (let's call it PP) the input A_1 to another a_1, and then to apply modified MP (i.e., we assume that it requires much less complexity or that it has certain merits to apply). In a more straight forward formula, $a_1 = PP(A_1)$, and then, $b_1 = MP(a_1)$. If the result b_1 approaches what we expect, B_1, and if the processing complexity PP is small, it is desirable to apply MP, which has more merits than P. Hence, the problem ends up with finding a PP (i.e., preprocessing) and an MP (i.e., modified processing merits we want to achieve) for corresponding P. For our example, P is STMAC processing, and PP is preprocessing operations, such as filtering and cut-and-paste operation.

If we do spatial and temporal manipulation such as filtering and cut-and-paste operation on the input video before operating a regular H.263 encoder/decoder pair, it is possible to achieve the compatible codec. For example, we can smooth the background area out considerably, and do the temporal filtering or cut-and-paste operation for the background area. The input sequence itself then looks very similar to the output of STMAC coded data. If we apply this data into a regular H.263 encoder, we can save a substantial number of bits since high frequencies have been removed from the background area and temporal data changes are minimal from frame to frame; thus, motion compensation works well and gives optimal temporal compression. In other words, we can achieve the compatible codec through a processing approximation. Note that the assumption of the approach is that the encoder has the "not coded" mode for temporal compression.

6.2.2 Distortion Time Tradeoffs Codec in Software Architecture

At this time, the Discrete Cosine Transform (DCT) is the most popular transform used for image compression applications. Reasons for its popularity include its

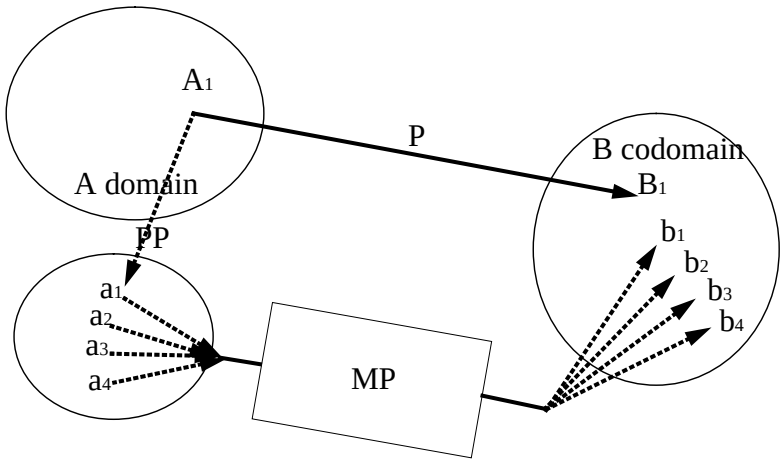


Figure 6-2: Processing approximation with rate distortion criterion.

good performance in terms of energy compaction for typical images as well as the availability of several fast algorithms for implementation.

One common factor in all fast algorithms proposed for hardware implementation of DCT is their aim at reducing the complexity of general forward or inverse DCT, regardless of the input to be transformed. For this purpose, complexity is estimated by the number of operations, which is the same for every input [54].

Recently, there have been several variable complexity algorithms proposed in image/video compression area [54, 55]. Most researchers focus on variable IDCT algorithms, under typical scenarios the image blocks on which IDCT must be performed are very sparse [54]. They look at the software encoder and decoder complexity and study the achievable performance for a given input to the codec. Since decoding speed depends on the number of zeros in the input, they then present a formulation that enables the encoder to optimize its quantizer selection so as to meet a prescribed decoding bit budget. Their approach leads to a complexity distortion

optimization technique that is analogous to well known techniques for rate distortion optimization. Their motivation comes from observing the increased importance of software implementations for various compression applications. This trend is likely to continue as faster general hardware becomes available and innovative uses of software become widespread [54, 55].

Our distortion time tradeoffs codec works the other way around. That is to say, to take advantage of a level of complexity among various complexity algorithms, we choose to perform a certain preprocessing on the input. If we want a low complexity algorithm, we perform significant manipulation on the input, which usually results in more distortion. On the other hand, if we want a high complexity algorithm, which usually results in a high quality output, we don't apply any preprocessing on the input images. Therefore, the problem is to devise a PP to take advantage of a variable complexity MP. For example, P can be STMAC processing, and PP can be a preprocessing operation such as filtering and cut-and-paste operation. Note that in this framework, we assume that the different complexity algorithms of a codec pair are well known to us.

Consider an example; a normal H.263 encoder uses a weighted motion estimation (ME) method that prefers a zero motion vector because motion vector data is quite expensive compared with other data in a compressed bitstream. Most implementations first check if the distortion measure (for example, absolute distance) at the zero motion vector in the ME algorithm is negative, because it doesn't need to continue computation for non zero motion vectors in that situation. Other examples include a Variable Length Coding (VLC). If the input is substantially smoothed, only a few coding steps are needed for VLC. On the other hand, if the input is very delicate, many steps of VLC procedures are needed.

To use some previously mentioned aspects, we can smooth the background area

out a lot, and do the temporal filtering or cut-and-paste operation of background area, causing the input sequence itself to look very similar to output of STMAC coded data. If we apply this data to a regular H.263 encoder, we can save a lot of "resources/steps" for VLC because the high frequency parts of the background area are already smoothed out. Furthermore, due to temporal data similarity from frame to frame, it is more likely to have many zero motion vectors. Thus, motion compensation can take advantage of a weighted ME method. Clearly then, we can achieve a high speed of software codec by introducing more blurriness or errors. In this example, the more we blur the input, the higher the speed we can achieve.

6.2.3 Spatio-Temporal Rate Control for Multiple MPEG-4 Objects

MPEG-4, an ongoing international standardization effort for multimedia communications, deals with objects. A basic assumption in MPEG-4 is that each object is compressed and transmitted independently. Independent objects are usually supposed to be kept in certain networked database systems, as shown in Figure 6-3. Compositing a certain set of objects, which are all from networked databases, at the user site is one likely scenario (such as for a multimedia presentation). We call this *non-real time scenario*. Compositing a certain set of objects – some of which are captured in real time while the rest are retrieved from networked databases – at the user site is a more generalized scenario. We call this *real time scenario*.

When it comes to multimedia applications, there are often bandwidth limitations at the user site. If there is a limited bandwidth situation with MPEG-4, the question is how to divide the bandwidth among the objects. This is a problem for "multiple object rate control," though, recently, a couple of ways to solve this problem were presented in papers [85, 78, 86]. In the future, we expect to expand our spatio-temporal rate control algorithm to multiple object environments.

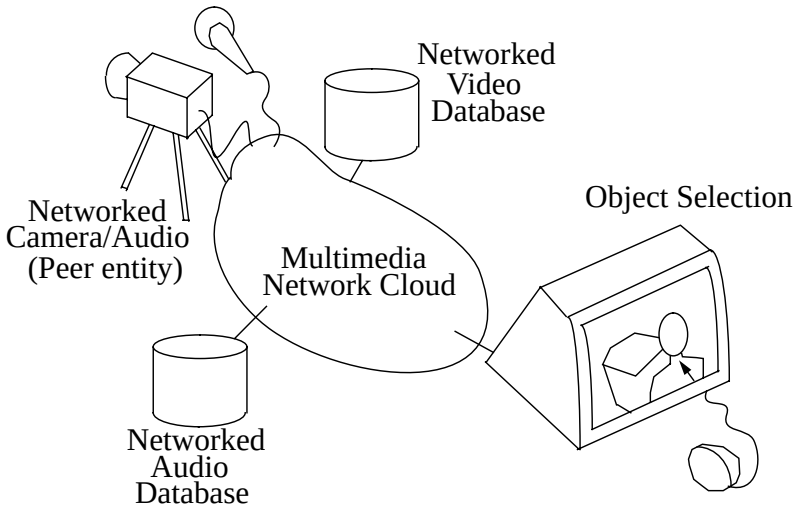


Figure 6-3: A generalized scenario for multiple object based coding.

In video conferencing images, it is clear that human beings look primarily at facial areas and that they feel comfortable if eyes and lips have nice quality in terms of space and time. In multiple object situations, however, it is still not clear what the most important object is.

We choose to focus on interactive and non-interactive applications. An interactive scenario is depicted in Figure 6-3. In MPEG-4, images and audio for objects come from various places and are composed into a scene; sometimes, a part is from networked databases that save pre-captured data, while another part comes from real time cameras. In such situations, users might want to assign relative importance to certain objects. For example, if a user doesn't like the given quality of a human-object in a certain scene, he will most likely want to see it more clearly than its background. In this case, users could change the relative importance of the human object interactively - a scenario known as "interactive." If interactivity from users is not possible, we call the scenario "non-interactive." The interactive control

policy is obvious; users can give p_i and γ_i to a certain object at their will. For example, if a user cares about a foregrounded human object as shown in Figure 6-3, he can give p_i as 30 (fps) and γ_i as a value higher than 1 (i.e., 1.5). In this case, other objects, such as the background, suffer in terms of quality due to bandwidth limitation. The non-interactive control is not obvious, however, so we must give our focus to it. To complete multiple object spatio-temporal rate control, the following three issues need to be addressed:

- How to assign p_i and γ_i to each object;
- How to adapt rate control algorithms in object scaling scenarios;
- How to assign bandwidth to non-real time objects.

For the first problem, the best approach may be to find a function based on human perception or based on given conditions such as

$$p_i = f(area,\ speed,\ other\ human\ visual\ factors) \tag{6.1}$$

and

$$\gamma_i = f(area,\ speed,\ other\ human\ visual\ factors). \tag{6.2}$$

The solution for non-interactive services is a special case of interactive services with a previously selected p_i and γ_i. Prefixed importance factors (p_i and γ_i) are given, based on a certain model. For example, human visual system characteristics can be a model for non-interactive service multi-object coding. The model can be adopted and adapted by encoders or publishers according to their needs as well. For example, if a publisher or a movie maker wants to emphasize rapidly moving

objects, a rougher QP and a higher fps can be assigned to those specific objects. How we assign a set of relative importance factors will be an issue.

For the second problem, the best direction may be to find a solid function for the β:

$$\beta = f(the\ number\ of\ objects,\ area\ of\ each\ object). \tag{6.3}$$

Note that β is the fundamental factor in the assignment of bit budgets to each frame, and that in video conferencing images, object motion is most likely, not object scaling.

For the third problem, the best direction may be to find a technique that dynamically varies the spatio-temporal properties of objects in the compressed domain. For example, if the object is relegated to the background and is retrieved from somewhere else, we probably do not need to keep the frame rate and quality high under a limited bandwidth at the user site.

References

[1] MPEG-4 Video Verification Model Version 8.0. *ISO/IEC JTC1/SC29/WG11*, July 1997.

[2] Final Committee Draft Coding of Audio-Visual Object: Visual. *ISO/IEC JTC1/SC29/WG11*, September 1998.

[3] D. Adolph and R. Buschmann. 1.15 Mbit/s coding of video signals including global motion compensation. *Signal Processing: Image Communication*, 3(2-3):259–274, June 1991.

[4] K. Aizawa, H. Harashima, and T. Saito. Model-based analysis synthesis image coding (MBASIC) system for a person's face. *Signal Processing: Image Communication*, 1(2):139–152, October 1989.

[5] K. Aizawa and T. S. Huang. Model-based image coding: Advanced coding techniques for very low bit rate applications. *Proceedings of the IEEE*, 83(2):259–271, February 1995.

[6] G. L. Anderson and A. N. Netravali. Image restoration based on a subjective criterion. *IEEE Trans. on Systems Man Cybern.*, SMC-6:845–853, 1976.

[7] M. R. Banham and J. C. Brailean. A selective update approach to matching pursuits video coding. *IEEE Trans. on CSVT*, 7(1):119–129, February 1997.

[8] R. Bellman. Dynamic Programming. *Princeton University Press*, 1957.

[9] D. P. Bertsekas. Dynamic Programming. *Prentice-Hall*, 1987.

[10] M. Bierling. A differential displacement estimation algorithm with improved stability. *Proceedings of SPIE*, 594:170–174, December 1985.

[11] M. Bierling. Displacement estimation by hierarchical block matching. *VCIP-88*, 1001:942–951, December 1988.

[12] M. Buck and N. Diehl. *Model-based image sequence coding.* in: Motion Analysis and Image Sequence Processing-Chap. 10, Kluwer Academic Publishers, Dordrecht, 1993.

[13] CCITT. Draft revision of recommendation h.261:. *Video codec for audio visual services at* $p \times 64$ *kbit/s*, November 1989.

[14] C.-T. Chen, F.-C. Jeng, M. Kawashima, S. Singhal, and A. Wong. Hybrid extended mpeg video coding algorithm for general video applications. *Signal Processing: Image Communication*, 5:21–37, 1993.

[15] L. Chiariglione. MPEG and multimedia communications. *IEEE Trans. on CSVT*, 7(1):5–18, February 1997.

[16] M. F. Chowdhury, A. F. Clark, A. C. Downton, E. Morimatsu, and D. E. Pearson. A switched model-based coder for video signal. *IEEE Trans. on CSVT*, 4(3), June 1994.

[17] C.-T. Chu. Core experiment report: motion compensation at VOP level. *ISO/IEC JTC1/SC29/WG11 MPEG96/M1103, Tampere*, July 1996.

[18] C.-T. Chu. Motion compensation at VOP level. *ISO/IEC JTC1/SC29/WG11 MPEG96/M0871, Firenze*, March 1996.

[19] C.-T. Chu. Object-based coding algorithm in low and very low bit rate video coding. *PhD Thesis, Columbia University*, 1997.

[20] C.-T. Chu, D. Anastassiou, and S.-F. Chang. Hybrid block-based/segment-based video coding at very low bitrate. *ISO/IEC JTC1/SC29/WG11 MPEG95/M0376, Dallas*, November 1995.

[21] C.-T. Chu, D. Anastassiou, and S.-F. Chang. Bi-directional object-based coding in video compression at very low bitrate. *Proceedings of the 3rd International Conference on Signal Processing, Beijing*, 1996.

[22] C.-T. Chu, D. Anastassiou, and S.-F. Chang. Hybrid block-based/object-based video coding at very low bitrate. *Proceedings of DCC*, page 431, April 1996.

[23] C.-T. Chu, D. Anastassiou, and S.-F. Chang. Hierarchical global motion estimation/compensation in low bitrate video coding. *Proceedings of IEEE Internatitonal Symposium on Circuits and Systems*, June 1997.

[24] C.-T. Chu, D. Anastassiou, and S.-F. Chang. Hybrid object-based/block-based coding in video compression at very low bitrate. *Signal Processing: Image Communication Journal, Special Issue on MPEG-4*, 10(1-3), July 1997.

[25] C.-T. Chu, D. Anastassiou, and S.-F. Chang. Motion-adapted content-based temporal scalability in very high compression ratio video coding. *Proceedings of DCC*, page 431, March 1997.

[26] T.-Y. Chung, K.-H. Jung, Y.-N. Oh, and D.-H. Shin. Quantization control for improvement of image quality compatible with MPEG-2. *IEEE Trans. on Consumer Electronics*, 40(4):821–825, November 1993.

[27] ISO/MPEG Test Model Editing Committee. Test Model 5. *ISO-IEC/JTC1/SC29/WG11/N0400*, April 1993.

[28] T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, 1991.

[29] W. Ding and B. Liu. Rate control of mpeg video coding and recoding by rate-quantization modeling. *IEEE Trans. on CSVT*, 6(1):12–20, February 1996.

[30] Touradj Ebrahimi. MPEG-4 Video Verification Model : A video encoding/decoding algorithm based on content representation. *Signal Processing : Image communication*, 3(1):26–40, June 1997.

[31] A. Eleftheriadis and A. Jacquin. Model-assisted coding of video teleconferencing sequences at low bit rates. *Proc. ISCAS '94*, May-June 1994.

[32] A. Eleftheriadis and A. Jacquin. Automatic face location detection and tracking for model-assisted coding of video teleconferencing sequences at low bit rates. *Image Communication journal*, 7(3):231–248, September 1995.

[33] A. Eleftheriadis and A. Jacquin. Automatic face location detection for model-assisted rate control in H.261 compatible coding of video. *Image Communication Journal, Special Issue on Coding Techniques for Very Low Bit rate Video*, 7(4-6):435–455, November 1995.

[34] M. Etoh, C. S. Boon, and S. Kadono. A template-based video coding with opacity representation. *IEEE Trans. on CSVT*, 7(1):172–180, February 1997.

[35] B. Falcidieno and T. L. Kunii. *Modeling in Computer Graphics*. Springer, Berlin, 1993.

[36] E. Francois, J.-F. Vial, and B. Chupeau. Coding algorithm with region-based motion compensation. *IEEE Trans. on CSVT*, 7(1):97–108, February 1997.

[37] J. I. Gimlett. Use of activity classes in adaptive transform image coding. *IEEE Trans. Communication Tech.*, COM-23:785–786, 1975.

[38] J. Hartung, A. Jacquin, H. Okada, J. Pawlyk, and J. Rosenberg. Object-based H.263 compatible video coding platform for conferencing applications. *IEEE Journal of Selected Areas in Communications*, 16(1), January 1998.

[39] ISO/MPEG. Coding of moving pictures and associated audio for digital storage media at up to 1.5 mbps. *ISO/IEC 11172-2*, 1992.

[40] ITU. Draft revision of recommendation H.261: video codec for audiovisual services at $p \times 64$ kbps. *Signal Processing:Image Communication*, 2(2):221–239, August 1990.

[41] ITU. Draft ITU-T recommendation H.263: video coding for low bitrate communication. *Expert's Group on Very Low Bitrate Video Telephony Draft*, July 1995.

[42] A. K. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall, Englewood Cliffs, NJ 07632, 1989.

[43] ISO/IEC JTC1/SC29/WG11. Motion Picture Expert Group. *Information technology - coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbits/s: video, ISO/IEC 11172-2*, 1992.

[44] ISO/IEC JTC1/SC29/WG11. Motion Picture Expert Group. *Test Model 5, Draft*, April 1993.

[45] ISO/IEC JTC1/SC29/WG11. Motion Picture Expert Group. *Information technology - generic coding of moving pictures and associated audio: video, ISO/IEC 13818-2*, 1995.

[46] ISO/IEC JTC1/SC29/WG11. Motion Picture Expert Group. *MPEG-4 video verification model version 6.0, MPEG96/N1582, Sevilla*, February 1996.

[47] A. K. Katsaggelos, L. P. Kondi, F. W. Meier, J. Ostermann, and G. M. Schuster. MPEG-4 and rate-distortion-based shape coding techniques. *Proceedings of The IEEE*, 86(6):1126–1154, June 1998.

[48] P. Kauff, B. Makai, S. Rauthenberg, U. Golz, J. L. P. D. Lameillieure, and T. Sikora. Functional coding of video using a shape-adaptive DCT algorithm and an object-based motion prediction toolbox. *IEEE Trans. on CSVT*, 7(1):181–196, February 1997.

[49] J.-B. Lee, J.-S. Cho, and A. Eleftheriadis. Optimal shape coding under buffer constraints. *IEEE ICIP'98*, 3(1):290–294, October 1998.

[50] J.-B. Lee and A. Eleftheriadis. Spatio-temporal model-assisted compatible coding for low and very low bitrate video telephony. *IEEE Proc. ICIP '96*, September 1996.

[51] J.-B. Lee and A. Eleftheriadis. Motion adaptive model-assisted compatible coding with spatio-temporal scalability. *Proc. VCIP '97*, February 1997.

[52] M.-C. Lee, W.-G. Chen, C.-L. B. Lin, C. Gu, T. Markoc, S. I. Zabinsky, and R. Szeliski. A layered video object coding system using sprite and affine motion model. *IEEE Trans. on CSVT*, 7(1):130–145, February 1997.

[53] D. LeGall. MPEG: a video compression standard for multimedia applications. *Communications of the ACM*, 34(4):46–58, April 1991.

[54] K. Lengwehasatit and A. Ortega. Distortion/decoding time tradeoffs in software DCT-based image coding. *Proceeding of ICASSP-97*, April 1997.

[55] K. Lengwehasatit, A. Ortega, A. Basso, and A. Reibman. A novel computationally scalable algorithm for motion estimation. *Proceeding of ICASSP-97*, April 1997.

[56] J. Mitchell, W. Pennebaker, C. Fogg, and D. LeGall. *MPEG video compression standard*. Chapman and Hall, New York, 1997.

[57] H. G. Musmann, M. Hotter, and J. Ostermann. Object-oriented analysis-synthesis coding of moving images. *Signal Processing: Image Communication*, 1(2):117–138, October 1989.

[58] A. Netravali and B. Haskell. *Digital Pictures: Repesentation, Compression, and Standards*. Plenum Press, New York, 1994.

[59] A. Ortega. Optimization techniques for adaptive quantization of image and video under delay constraints. *PhD Thesis, Columbia University in the City of New York*, 1994.

[60] A. Ortega, K. Ramchandran, and Martin Vetterli. Optimal Trellis-based buffered compression and fast approximations. *IEEE Iransactions on Image Processing*, 3(1):26–40, January 1994.

[61] J. Ostermann. Efficient encoding of binary shapes using MPEG-4. *IEEE ICIP'98*, 3(1):295–298, October 1998.

[62] W. A. Pearlman. A visual system model and a new distortion measure in the context of image processing. *J. Opt. Soc. Am.*, 68:374–386, 1978.

[63] D. E. Pearson. Developments in model-based video coding. *Proceedings of the IEEE*, 83(6), June 1995.

[64] F. Pereira. MPEG-4 video subjective test procedures and results. *IEEE Trans. on CSVT*, 7(1):32–51, February 1997.

[65] M. Pickering, J. Arnold, and M. Cavenor. VBR rate control with a human visual system based distortion measure. *In Australian Broadband Switching and Services Symposium*, July 1992.

[66] A. Puri and R. Aravind. On comparing motion-interpolation structures for video coding. *SPIE VCIP*, 1360:1560–1571, 1990.

[67] A. Puri and R. Aravind. Motion-compensated video coding with adaptive perceptual quantization. *IEEE Trans. on CSVT*, 1(4):351–361, December 1991.

[68] A. Puri and A. Eleftheriadis. MPEG-4: An object-based multimedia coding standard supporting mobile applications. *Mobile Networks and Applications*, 3:5–32, March 1998.

[69] W. Rabiner and A. Jacquin. Object tracking using motion-adaptive model of scene content. *Globecom '96*, pages 877–881, June 1996.

[70] B. Ramamurthi and A. Gersho. Classified vector quantization of images. *IEEE Trans. Communication*, COM-34:1105–1115, 1986.

[71] K. Ramchandran, A. Ortega, and M. Vetterli. Bit allocation for dependent quantization with applications to multiresolution and MPEG video. *IEEE Iransactions on Image Processing*, 3(5):533–545, September 1994.

[72] K. Ramchandran and M. Vetterli. Rate distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility. *IEEE Iransactions on Image Processing*, 3(5):700–704, September 1994.

[73] M. G. Ramos and S. S. Hemami. *Proceedings VCIP-96*, Orlando FL, March 1996.

[74] M. G. Ramos and S. S. Hemami. *Proceedings VCIP-98*, San Jose CA, March 1998.

[75] X. Ran and N. Farvardin. A perceptually motivated three-component image model-Part i: discription of the model. *IEEE Trans. on Image Processing*, 4(4):401–415, April 1995.

[76] E. Reusens, T. Ebrahimi, C. L. Buhan, R. Castagno, V. Vaerman, L. Piron, C. Fabregas, S. Bhattacharjee, F. Bossen, and M. Kunt. Dynamic approach to visual data compression. *IEEE Trans. on CSVT*, 7(1):197–211, February 1997.

[77] K. Rijkse. ITU standardisation of very low bitrate video coding algorithms. *Signal Processing: Image Communication*, pages 553–565, July 1995.

[78] J. Ronda, M. Eckert, S. Rieke, F. Jaureguizar, and A. Pacheco. Advanced rate control for MPEG-4 coders. *SPIE VCIP-97*, 3309:383–390, February 1997.

[79] P. Salembier, L. Torres, R. Meyer, and C. Gu. Region-based video coding using mathematical morphology. *Proceeding of the IEEE*, 83:843–857, June 1995.

[80] G. M. Schuster and A. K. Katsaggelos. An optimal ploygonal boundary encoding scheme in the rate distortion sense. *IEEE Trans. on Image Processing*, 7(1):13–26, January 1998.

[81] ITU-T SGXV. Expert's group for very low bitrate videophone, LBC-95. *Draft recommendation H.263*, July 1995.

[82] T. Sikora. The MPEG-4 video standard verification model. *IEEE Trans. on CSVT*, 7(3):449–458, February 1997.

[83] H. Sun, W. Kwok, M. Chien, and C. H. Ju. MPEG coding performance improvement by jointly optimzing coding mode decisions and rate control. *IEEE Iransactions on Circuits and Systems for Video Technology*, 7(3):449–458, June 1997.

[84] A. Vetro, H. Sun, and Y. Wang. MPEG-4 rate control for multiple video objects. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(1):186–199, February 1999.

[85] A. Vetro, H. Sun, and Y. Wang. MPEG-4 rate control for multiple video objects. *IEEE Trans. on CSVT*, 9(1):186–199, February 1999.

[86] L. Wang, A. Vincent, and P. Corriveau. Muliti-program video coding with joint rate control. *IEEE ICIP-96*, 3:1516–1520, September 1996.

[87] W. J. Welsh. Model-based coding of videophone images. *Electronics and Communication Engineering Journal*, pages 29–36, February 1991.

[88] W. J. Welsh, S. Searby, and J. B. Waite. Model-based coding. *British Technology Journal*, 8(3):94–106, July 1990.