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

Jae-Beom Lee, *Member, IEEE*, Jin-Soo Cho, *Student Member, IEEE*, and Alexandros Eleftheriadis, *Member, IEEE*

***Abstract*—We propose an optimal buffered compression algorithm for shape coding as defined in the forthcoming MPEG-4 international standard. The MPEG-4 shape coding scheme consists of two steps: first, distortion is introduced by down and up scaling; then, context-based arithmetic encoding is applied. Since arithmetic coding is "lossless," the down up scaling step is considered as a virtual quantizer. We first formulate the buffer-constrained adaptive quantization problem for shape coding, and then propose an algorithm for the optimal solution under buffer constraints. Recently the fact that a conversion ratio (CR) of $1/4$ makes coded image irritating to human observers for QCIF size was reported for MPEG-4 shape coding. Therefore, a careful consideration for small size images such as QCIF should be given to prevent coded images from being unacceptable. To this end, a low bit rate tuned algorithm is proposed in this paper as well. Experimental results are given using an MPEG-4 shape codec.**

I. Introduction

A project of the International Standardization Organization (ISO), MPEG-4 is an emerging coding standard that supports new ways for communication, access and manipulation of digital audio–visual data. MPEG-4 offers 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 [10].

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 [3].

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 [3].

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 paper, we concentrate on binary shape compression.

Lossy shape coding techniques were reported in several recent papers [4], [11]. 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 [4]. In this paper, 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 [6] 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

Manuscript received May 5, 1999; revised February 5, 2001. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Rashid Ansari.

J.-B. Lee was with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA. He is now with SolidStreaming, Inc., New York, NY 10005 USA.

J.-S. Cho was with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA. He is now with the Georgia Institute of Technology, Atlanta, GA 30332 USA.

A. Eleftheriadis is with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: eleft@ee.columbia.edu).

Publisher Item Identifier S 1057-7149(01)03271-7.

interactive multimedia systems suggests further investigation on the optimal buffered compression issue on shape information as well.

This paper 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 suboptimal, solutions. In addition, a low bit rate tuned algorithm is proposed for very low bitrate applications such as wireless and/or Internet video.

The structure of the paper is as follows. In Section II, the optimal buffered compression problem is described. Section III explains the background for MPEG-4 shape coding, and provides a quantitive 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 IV and V, and simulation results follow in Section VI. A discussion and concluding remarks are given in Section VII.

II. Optimal Buffered Compression with Mode Selection

A. 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% (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 [13]. 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 [13]. 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 [4], [11]. 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 [4]. More recently, a shape coding

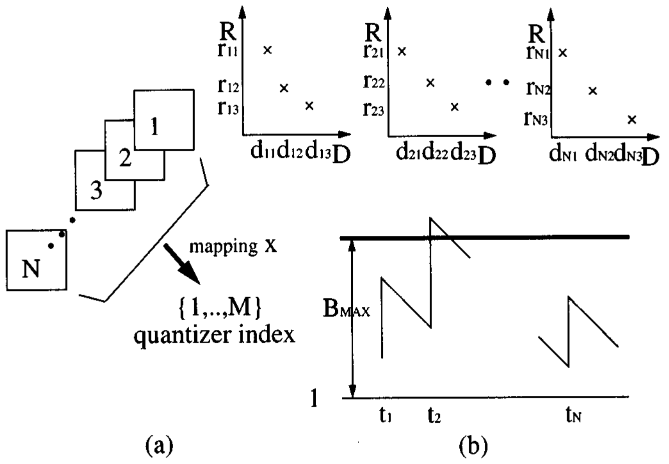


Fig. 1. (a) Each block in the sequence has a different R-D characteristic (for a given choice of quantizers for the blocks in the sequence, we can obtain R-D points to form the composite characteristic) and (b) R at t_2 is not a feasible solution with the chosen buffer size (buffer is limited).

control algorithm was devised to reduce the amount of bits used for shape data under low bit-rate conditions [13] 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 paper, a framework of optimality in operational rate-distortioin 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 $= 1/4$ makes coded image irritating to human observers for QCIF size, as was reported in [7] through the context of MPEG-4 CAE. Thus, a low bit rate tuned algorithm is introduced as well.

B. Problem Definition

Recently, optimal trellis-based buffered compression was proposed in [6] to provide optimal buffer control strategies for video sequences in a finite buffer environment. Fig. 1 depicts the traditional optimal buffered compression approach. Originally, optimal buffered compression was defined to select only the optimal quantizer. In this paper, 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 $\{(2, \text{intra\_mode}), (3, \text{inter\_mode}), (3, \text{intra\_mode}), (1, \text{intra\_mode}), (2, \text{inter\_mode}), (1, \text{inter\_mode})\}$. 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 $\{1, 2, \ldots, N\}$ to $\{1, 2, \ldots, M\}$, $x = \{x(1), x(2), \ldots, x(N)\}$, where each $x(i)$ is the quantizer index for block i. Therefore, $(r_{1x(1)}, \ldots, r_{Nx(N)})$ and $(d_{1x(1)}, \ldots, d_{Nx(N)})$ 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

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

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.

General Formulation: (Integer Programming): The problem is to find the mapping x that solves

$$\min_{x(i),\, i=1,\ldots,N} \left(\sum_{i=1}^{N} d_{ix(i)} \right) \tag{2}$$

subject to

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

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.

III. Opiimal Rate Control for MPEG-4 Shape Coding

A. 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.

Current rate control and rate reduction in MPEG-4 is realized through size conversion of the binary alpha information as

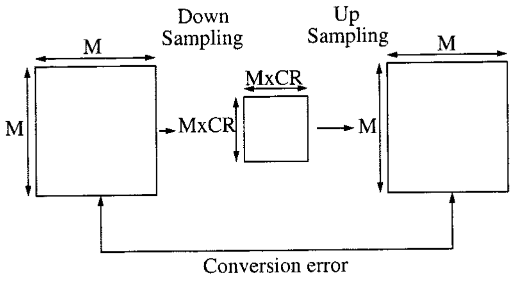


Fig. 2. Size conversion.

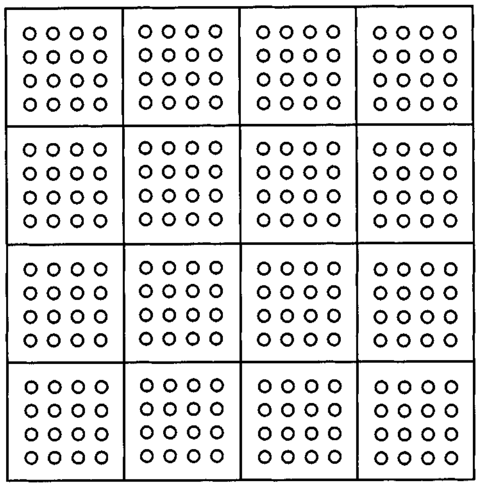


Fig. 3. A BAB consists of 16 PBs.

shown in Fig. 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 Fig. 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:

$$ACQ(\text{BAB}') \overset{def}{=} \text{MIN}(acq_1, acq_2, \ldots, acq_{16}) \tag{3}$$

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 $\{0, 16, 32, 64, \ldots, 256\}$. 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-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 eight-bit filter context, C_f, is calculated as follows:

$$C_f = \sum_k c_k \cdot 2^k. \tag{4}$$

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 Fig. 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 $ACQ(\text{BAB}')$. 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 follows:

1) `MVDs` $==$ `0 && No Update`
2) `MVDs!` $=$ `0 && No Update`
3) `all_0`
4) `all_255`
5) `intraCAE`
6) `MVDs` $==$ `0 && interCAE`
7) `MVDs!` $=$ `0 && interCAE`.

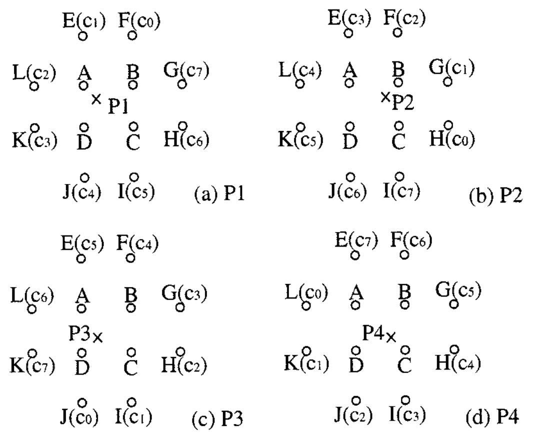


Fig. 4. Interpolation filter and interpolation construction.

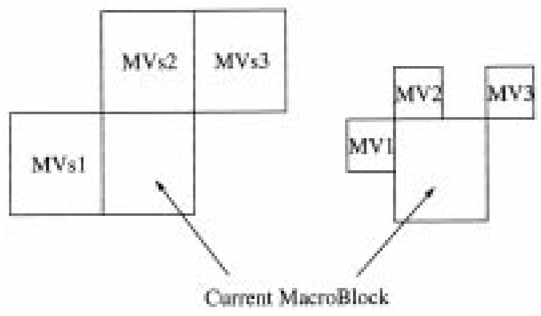


Fig. 5. Candidates for MVPs.

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 Fig. 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.

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 subblocks, 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.

B. 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 paper, 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 VM, 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 the 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 $(\text{BAB}(i).\text{CR}, \text{coding\_mode})$, thus eliminating $alpha\_th$. In the above expressions, BAB(i) means ith BAB. Therefore, the problem formulation can be given as follows.

Shape Coding Formulation: (Integer Programming): Let x be $\{x(1), x(2), \ldots, x(N)\}$ where $x(i) = (\text{BAB}(i).\text{CR}, \text{coding\_mode})$.

The problem is to find the mapping x that solves

$$\min_{x(i),\, i=1,\ldots,N} \left(\sum_{i=1}^{N} d_{ix(i)} \right) \tag{5}$$

subject to

$$B_i \le B_{\max}, \qquad \forall\, i = 1, \cdots, N$$

where $B_{\max}$ is the maximum buffer size.

Fig. 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 BAB.CR value involved. The problem of optimal shape coding is to decide a certain path which gives minimum overall distortion under $B_{\max}$ as defined in Equation (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

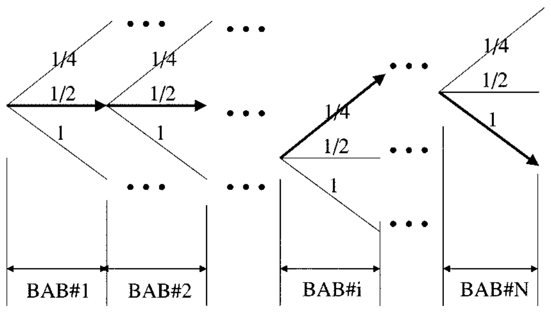


Fig. 6. One path determines a quality and rate of output shape data (branch value means BAB.CR).

considered), then we need only to keep the minimum cost path [5] and [6].

IV. 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.

A. 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 [6], [12]. The greedy method was reported to achieve a nearly optimal performance with much relaxed burden of computational complexity in [12]. The proposed greedy method in [12] gives much less computational complexity than the Lagrange multiplier method in [6].

We take greedy method approach in this paper based on following two reasons: firstly, the greedy method shows to achieve a very close performance to optimal algorithm as shown in [5], [6], [8], [9], and [12]; 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 $\text{CR} = 1$ 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 [6], was computational intensive since the number of quantizers is usually not small (i.e., the QP value ranges from one to 31). However, the shape part in this paper is quite feasible because the number of quantizers is just three (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 seven BAB coding modes (`MVDs` $==$ `0 && No Update all_0, all_255, intraCAE, MVDs` $==$ `0 && interCAE`, `MVDs!` $=$ `0 && interCAE`).

Let us 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 $=$ $1/2$ or CR $= 1/4$ also produces no distortion. Since the size is smaller at CR $= 1/2$ or CR $= 1/4$, the generated bits by the CAE are definitely smaller than that of CR $= 1$.

In CR $= 1/2$ case, distortion is usually introduced. Therefore, the encoding mode which makes the lowest distortion should be a candidate among BAB coding modes at CR $= 1/2$ 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 $= 1/4$ produces a smaller distortion than that of CR $= 1/2$.

The same consideration should be also applied to CR $= 1/4$ case. In CR $= 1/4$ 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 $= 1/4$ 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 {
  $mode_1 = MODE_{\min}(D_{intraCAE}(\mathrm{CR} = 1), D_{interCAE}(\mathrm{CR} = 1), D^T_{intraCAE}(\mathrm{CR} = 1), D^T_{interCAE}(\mathrm{CR} = 1))$;
  $mode_2 = MODE_{\min}(D_{intraCAE}(\mathrm{CR} = 1/2), D_{interCAE}(\mathrm{CR} = 1/2), D^T_{intraCAE}(\mathrm{CR} = 1/2), D^T_{interCAE}(\mathrm{CR} = 1/2))$;
  $mode_3 = MODE_{\min}(D_{intraCAE}(\mathrm{CR} = 1/4), D_{interCAE}(\mathrm{CR} = 1/4), D^T_{intraCAE}(\mathrm{CR} = 1/4), D^T_{interCAE}(\mathrm{CR} = 1/4))$;
  if ($D_{mode_3}(\mathrm{CR} = 1/4)$ is the smallest distortion) {
    Use $mode_3$;
  }
  else if (($D_{mode_2}(\mathrm{CR} = 1/2)$ is the smallest distortion) {
    Use $mode_2$;
  }
  else if ($D_{mode_1}(\mathrm{CR} = 1)$ 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. $B_{intraCAE}(\mathrm{CR} = 1)$ and $B_{interCAE}(\mathrm{CR} = 1)$ mean the current buffer sizes just after `intraCAE`$(\mathrm{CR} = 1)$ and `interCAE`$(\mathrm{CR} = 1)$ are applied respectively. Similar meanings can be defined for the expression of $B_{intraCAE}(\mathrm{CR} = 1/2)$, $B_{interCAE}(\mathrm{CR} = 1/2)$, $B_{intraCAE}(\mathrm{CR} = 1/4)$, and $B_{interCAE}(\mathrm{CR} = 1/4)$. And $B^T_{intraCAE}(\mathrm{CR} = 1)$, $B^T_{interCAE}(\mathrm{CR} = 1)$, $B^T_{intraCAE}(\mathrm{CR} = 1/2)$, $B^T_{interCAE}(\mathrm{CR} = 1/2)$, $B^T_{intraCAE}(\mathrm{CR} = 1/4)$, and $B^T_{interCAE}(\mathrm{CR} = 1/4)$ 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, $D_{(encode)}(\mathrm{CR} = \mathrm{cr})$ 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. $D_{intraCAE}(\mathrm{CR} = 1)$, $D_{interCAE}(\mathrm{CR} = 1)$, $D_{intraCAE}(\mathrm{CR} = 1/2)$, $D_{interCAE}(\mathrm{CR} = 1/2)$, $D_{intraCAE}(\mathrm{CR} = 1/4)$, and $D_{interCAE}(\mathrm{CR} = 1/4)$ are defined as distortion just after the BAB are CAE-ed as such. And $D^T_{intraCAE}(\mathrm{CR} = 1)$, $D^T_{interCAE}(\mathrm{CR} = 1)$, $D^T_{intraCAE}(\mathrm{CR} = 1/2)$, $D^T_{interCAE}(\mathrm{CR} = 1/2)$, $D^T_{intraCAE}(\mathrm{CR} = 1/4)$, and $D^T_{interCAE}(\mathrm{CR} = 1/4)$ 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 VM document since the notion of optimal buffered compression does not allow us to use $alpha\_th$ (i.e., motion vectors in interCAE mode in the current MPEG-4 VM 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.

B. 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 $\mathrm{CR} = 1/2$ has smaller distortion than that from $\mathrm{CR} = 1/4$. We also assume that this is true between the original image and the reconstructed image from $\mathrm{CR} = 1/2$. 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 nonoverflow buffer condition, we do not 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.

C. 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 ($\mathrm{Min}\{B_{intraCAE}(\mathrm{CR} = 1), B_{interCAE}(\mathrm{CR} = 1),$
$B^T_{intraCAE}(\mathrm{CR} = 1), B^T_{interCAE}(\mathrm{CR} = 1)\} < B_{\max})$ {
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR} = 1), B_{interCAE}(\mathrm{CR} = 1),$
  $B^T_{intraCAE}(\mathrm{CR} = 1), B^T_{interCAE}(\mathrm{CR} = 1))$;
}
else if ($\mathrm{Min}\{B_{intraCAE}(\mathrm{CR} = 1/2), B_{interCAE}(\mathrm{CR} = 1/2),$
$B^T_{intraCAE}(\mathrm{CR} = 1/2), B^T_{interCAE}(\mathrm{CR} = 1/2)\} < B_{\max})$
  {
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR} = 1/2),$
  $B_{interCAE}(\mathrm{CR} = 1/2), B^T_{intraCAE}(\mathrm{CR} = 1/2),$
  $B^T_{interCAE}(\mathrm{CR} = 1/2))$;
}
  else if
  ($\mathrm{Min}\{B_{intraCAE}(\mathrm{CR} = 1/4), B_{interCAE}(\mathrm{CR} = 1/4),$
$B^T_{intraCAE}(\mathrm{CR} = 1/4), B^T_{interCAE}(\mathrm{CR} = 1/4)\} < B_{\max})$
  {
```

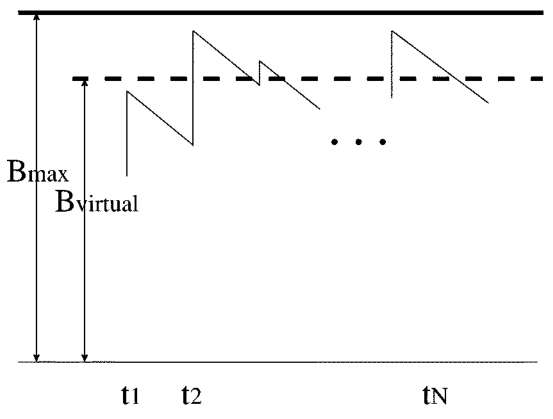


Fig. 7. Marginal buffer reservation.

```
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR}=1/4),$
  $B_{interCAE}(\mathrm{CR}=1/4),$
  $B^T_{intraCAE}(\mathrm{CR}=1/4), B^T_{interCAE}(\mathrm{CR}=1/4))$;
}
else if (“the number of backward” <= a
  pre-fixed number) {
    Backward mode;
}
else {
    Choose $\mathrm{CR}=1/4$ 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. 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.

D. Marginal Buffer Reservation Algorithm

We observe that the maximum buffer size does not 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 $B_{\max}$ as shown in Fig. 7. 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 do not need to go backward. Since the algorithm always assigns CR $= 1/4$ 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;
}
else if $(\mathrm{Min}\{B_{intraCAE}(\mathrm{CR}=1),$
  $B_{interCAE}(\mathrm{CR}=1), B^T_{intraCAE}(\mathrm{CR}=1),$
  $B^T_{interCAE}(\mathrm{CR}=1)\} < B_{virtual})$ {
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR}=1), B_{interCAE}(\mathrm{CR}=1),$
  $B^T_{intraCAE}(\mathrm{CR}=1), B^T_{interCAE}(\mathrm{CR}=1))$;
}
else if $(\mathrm{Min}\{B_{intraCAE}(\mathrm{CR}=1/2),$
  $B_{interCAE}(\mathrm{CR}=1/2), B^T_{intraCAE}(\mathrm{CR}=1/2),$
  $B^T_{interCAE}(\mathrm{CR}=1/2)\} < B_{virtual})$ {
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR}=1/2),$
  $B_{interCAE}(\mathrm{CR}=1/2),$
  $B^T_{intraCAE}(\mathrm{CR}=1/2), B^T_{interCAE}(\mathrm{CR}=1/2))$;
}
else if $(\mathrm{Min}\{B_{intraCAE}(\mathrm{CR}=1/4), B_{interCAE}(\mathrm{CR}=$
  $1/4), B^T_{intraCAE}(\mathrm{CR}=1/4), B^T_{interCAE}(\mathrm{CR}=$
  $1/4)\} < B_{virtual})$ {
{
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR}=1/4),$
  $B_{interCAE}(\mathrm{CR}=1/4), B^T_{intraCAE}(\mathrm{CR}=1/4),$
  $B^T_{interCAE}(\mathrm{CR}=1/4))$;
}
else {
    Choose $\mathrm{CR}=1/4$ and perform CAE;
}.
```

Note that this algorithm is a modified version for rate distortion optimization of a practical method shown in typical MPEG rate control methods. More specifically, if buffer ocupancy is over 90%, then the worst quantizer (i.e., 31) is used in typical methods to avoid buffer overflow. The main difference is that the worst quantization is chosen based only on buffer fullness (e.g., eventhough better quantizers are allowed in terms of buffer space) in traditional methods, while the worst quantizer is chosen if any of quantizer is not allowed under the virtual buffer maximum in the proposed method. Once again, the marginal buffer reservation method was devised to eliminate backward step-backs to relax computational burden, while the traditional method was devised to avoid buffer overflow. Note that such a case happens more frequently in MPEG-4 shape coding than MPEG texture coding because only three quantizers are allowed.

V. A Low-Bit-Rate-Tuned Algorithm

To prevent coded images from being unacceptable at a low bit rate, in [7] it is recommended to maintain CR value as 1 or 1/2.

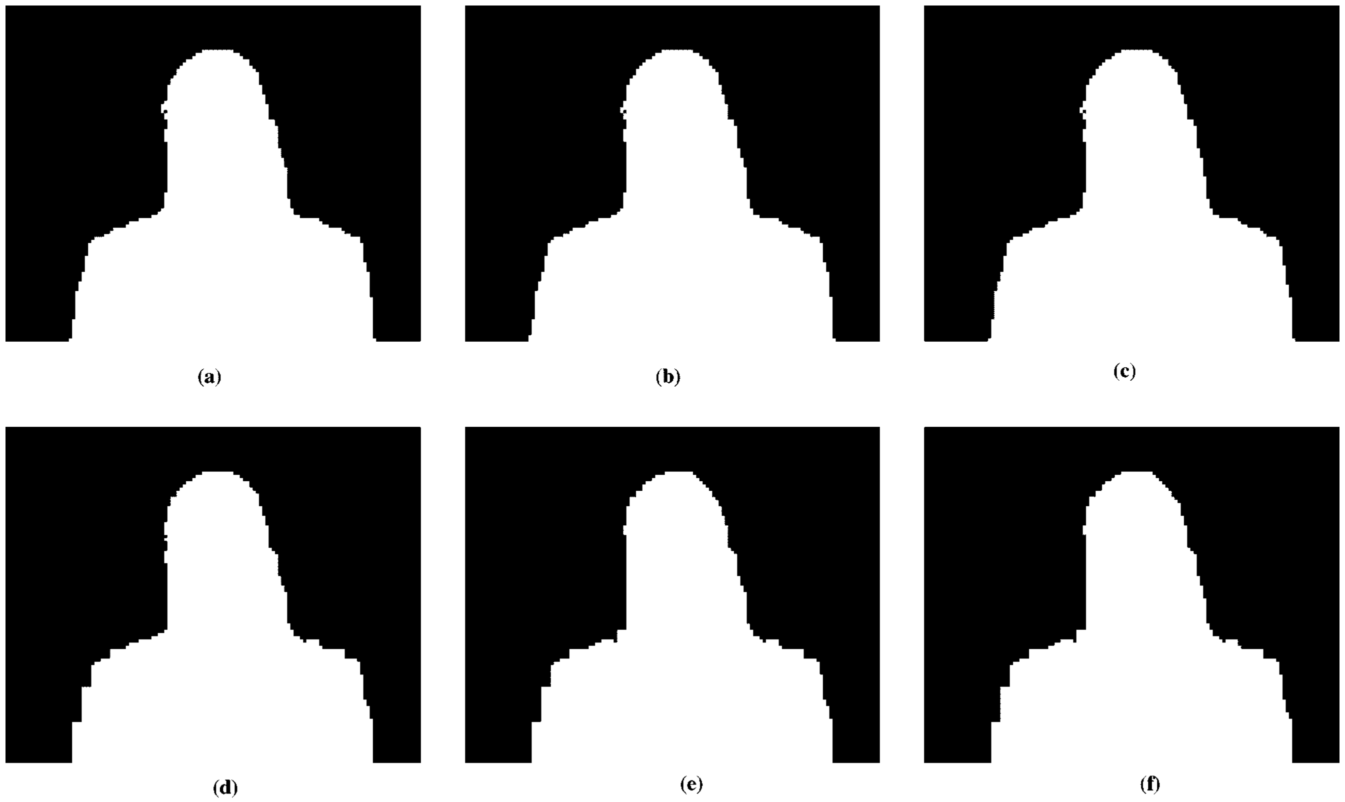


Fig. 8. Decoded outputs (MPEG-4 shape decoder $B_{\max} = 1000, r = 6$): (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

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 do not 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 $= 1/2$ 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 $(\mathrm{Min}\{B_{intraCAE}(\mathrm{CR} = 1), B_{interCAE}(\mathrm{CR} = 1), B^T_{intraCAE}(\mathrm{CR} = 1), B^T_{interCAE}(\mathrm{CR} = 1)\} < B_{virtual})$ {
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR} = 1), B_{interCAE}(\mathrm{CR} = 1), B^T_{intraCAE}(\mathrm{CR} = 1), B^T_{interCAE}(\mathrm{CR} = 1))$;
}
else if $(\mathrm{Min}\{B_{intraCAE}(\mathrm{CR} = 1/2), B_{interCAE}(\mathrm{CR} = 1/2), B^T_{intraCAE}(\mathrm{CR} = 1/2), B^T_{interCAE}(\mathrm{CR} = 1/2)\} < B_{virtual})$ {
  Use $MODE_{\min}(B_{intraCAE}(\mathrm{CR} = 1/2), B_{interCAE}(\mathrm{CR} = 1/2), B^T_{intraCAE}(\mathrm{CR} = 1/2), B^T_{interCAE}(\mathrm{CR} = 1/2))$;
}
else {
   Choose $\mathrm{CR} = 1/2$ and perform CAE;
}.
```

VI. 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 $B_{\max}$. Figs. 8 and 9 show that the steady-state quality, say sixth 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., $r = 6$)

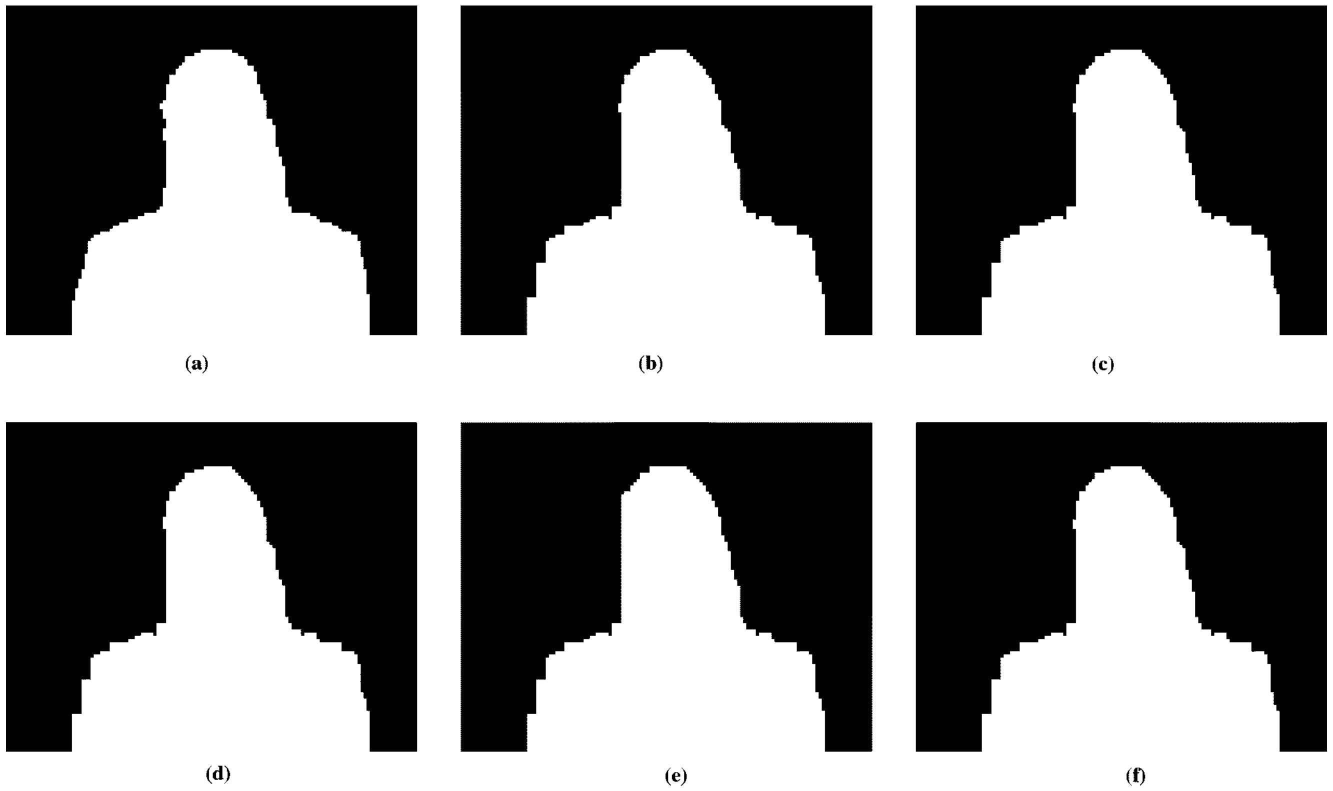

(a) (b) (c) (d) (e) (f)

Fig. 9. Decoded outputs (MPEG-4 shape decoder $B_{\max} = 500,\ r = 6$: (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

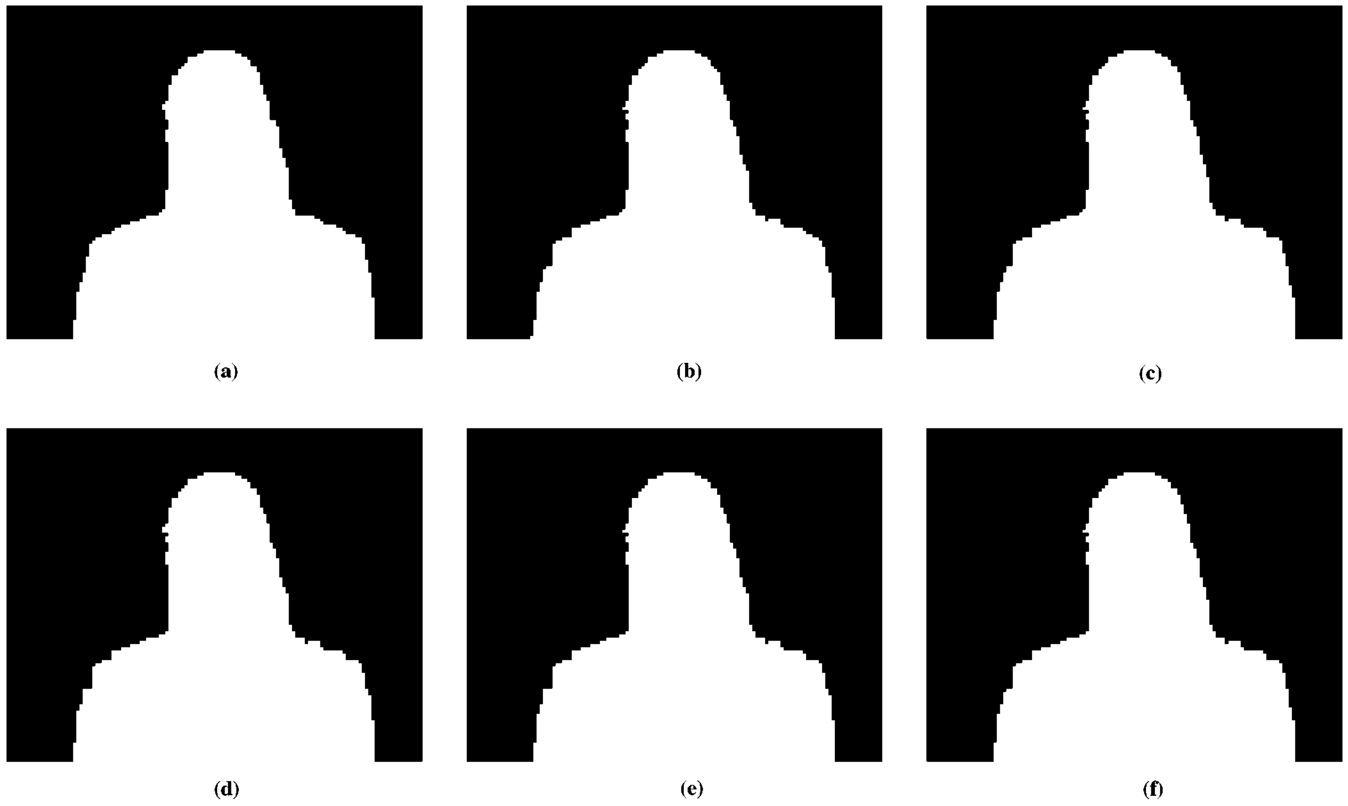

(a) (b) (c) (d) (e) (f)

Fig. 10. Decoded outputs (MPEG-4 shape decoder $B_{\max} = 500,\ r = 7$): (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

in Figs. 8 and 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,

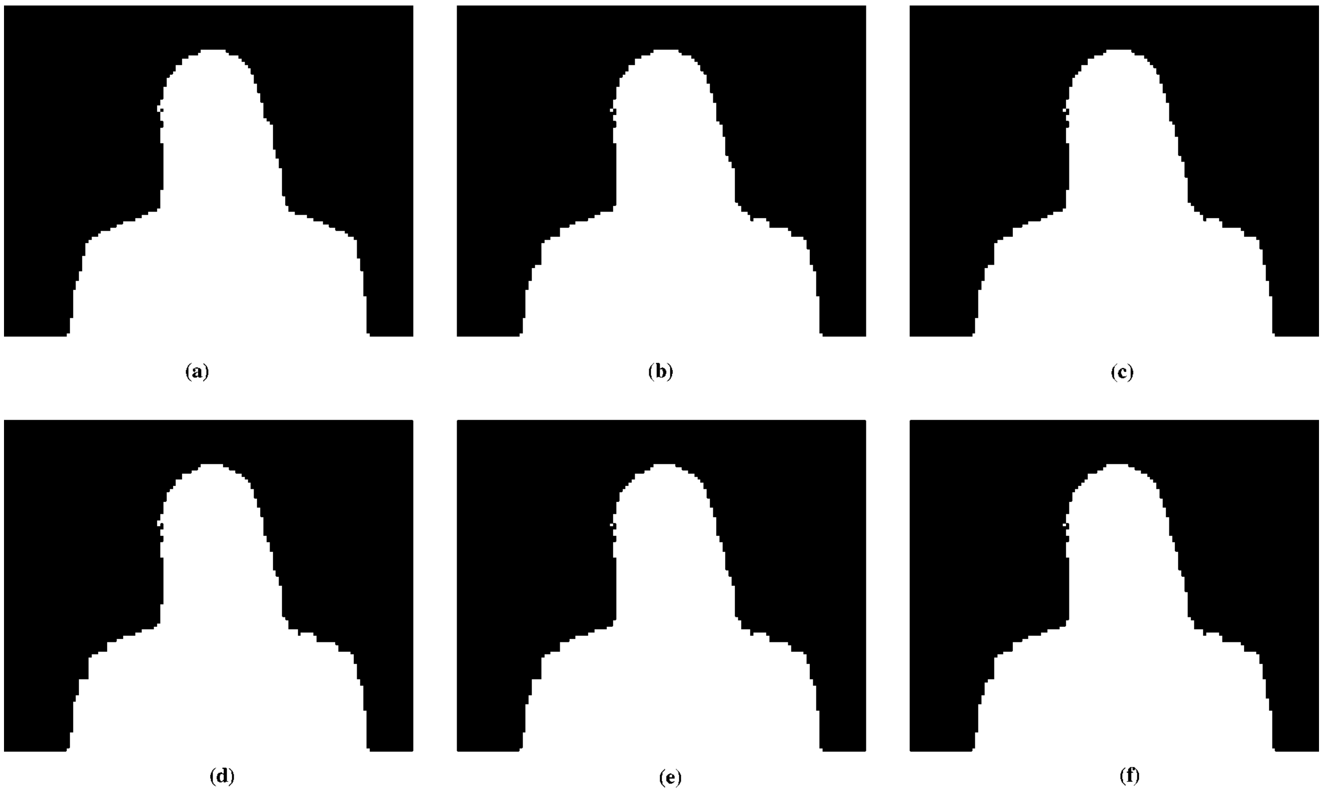


Fig. 11. Decoded outputs (MPEG-4 shape decoder $B_{\max} = 480$, $r = 7$): (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

two backward steps were allowed. Generally, the shape coding does not 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. Figs. 10 and 11 show the decoded outputs at $r = 7$.

This means that each BAB can be encoded by seven 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 ten 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 five bits as the channel rate, we will have distorted images. With the similar reason for each BAB the CR value of "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 us call that range of the channel rate "break down" region. In Akiyo sequence, meaningful distortion usually happens at channel rate r from six based on our experiments, while almost noiseless coding is adopted/applied from channel rate nine.

Binary data for a BAB is of 16×16 bits (i.e., 256). If we take the channel rate six for Akiyo sequence, the compression ratio is 42.66 to one in steady-state. If we take the channel rate eight, the compression ratio is 32 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,

TABLE I
Dn FROM THE FINITE BACKWARD MODIFICATION

(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 II
Dn FROM THE MARGINAL BUFFER RESERVATION ALGORITHM WITH $B_{\max} = 1000$ AND $B_{\max} = 500$

$(B_{virtual},r)$	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

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 finite backward modification and the marginal buffer algorithm. The general characteristics are the same as both two cases.

Table I shows the Dn obtained from the finite backward modification. If we take a look at the (1000, $r = 6$) and (500, $r = 6$) cases, the steady-state Dns seem to be similar around 0.07. This explains, once again, that the maximum buffer size $B_{\max}$ does

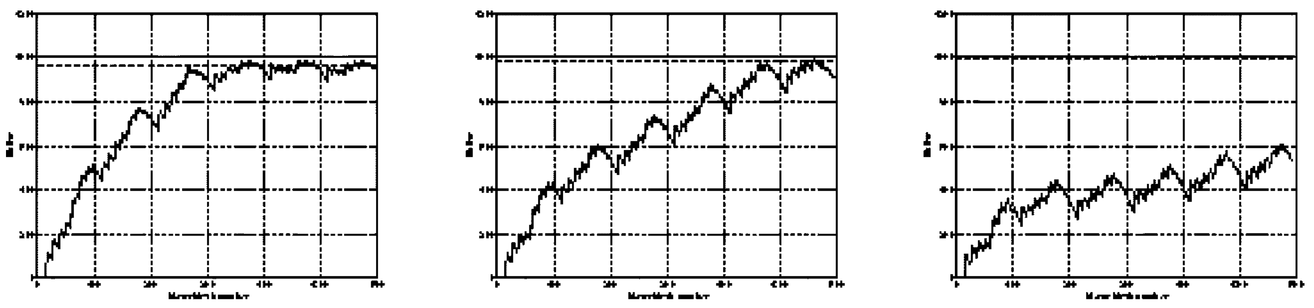


Fig. 12. Buffer occupancy (MPEG-4 shape encoder $B_{\max} = 1000$): (a) $B_{virtual} = 960$ and $r = 6$, (b) $B_{virtual} = 980$ and $r = 7$, and (c) $B_{virtual} = 990$ and $r = 8$.

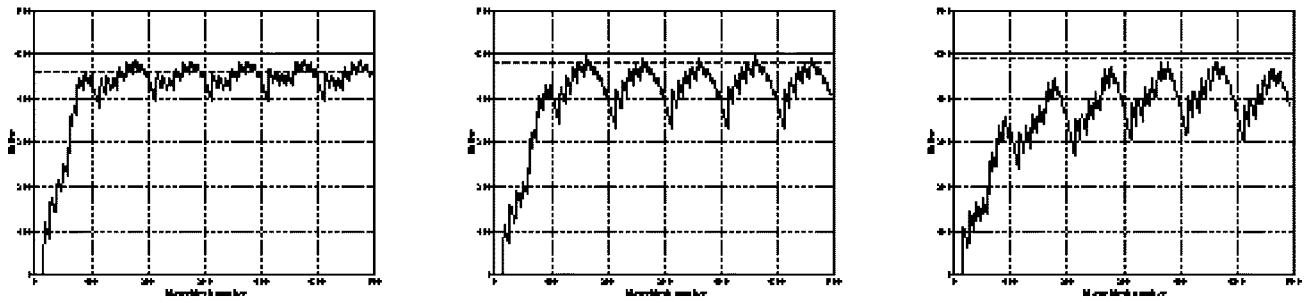


Fig. 13. Buffer occupancy (MPEG-4 shape encoder $B_{\max} = 500$): (a) $B_{virtual} = 460$ and $r = 6$, (b) $B_{virtual} = 480$ and $r = 7$, and (c) $B_{virtual} = 490$ and $r = 8$.

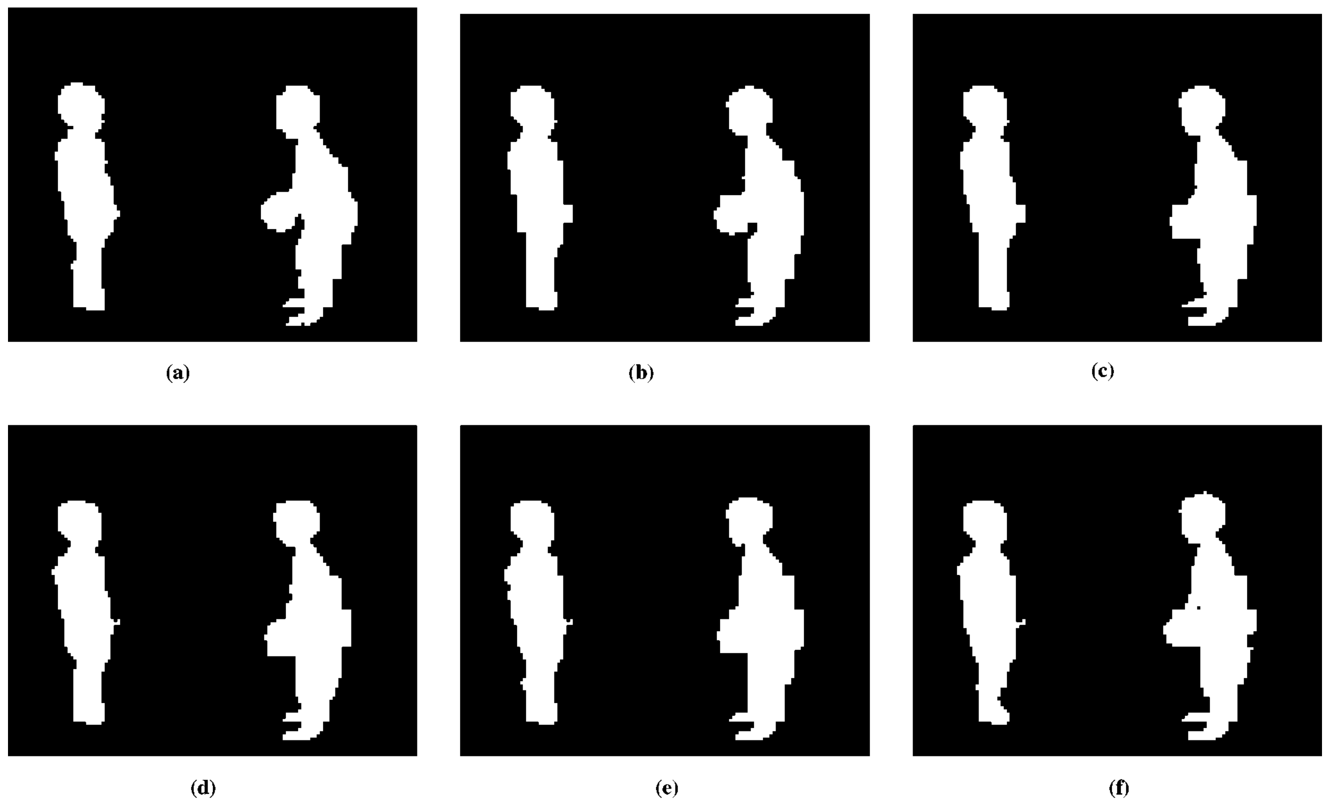


Fig. 14. Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500,\ r = 12$): (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

not affect the steady-state quality. Some of the values are "infinite," which means that the current buffer occupancy does not approach $B_{\max}$. Table II also shows the Dn obtained from the marginal buffer reservation algorithm. Note that the Dn 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 Dns for frames later than second frame are stabilized around 0.07.

The characteristics of finite backward modification is that the occupancy of the encoding buffer cannot undergo a certain max-

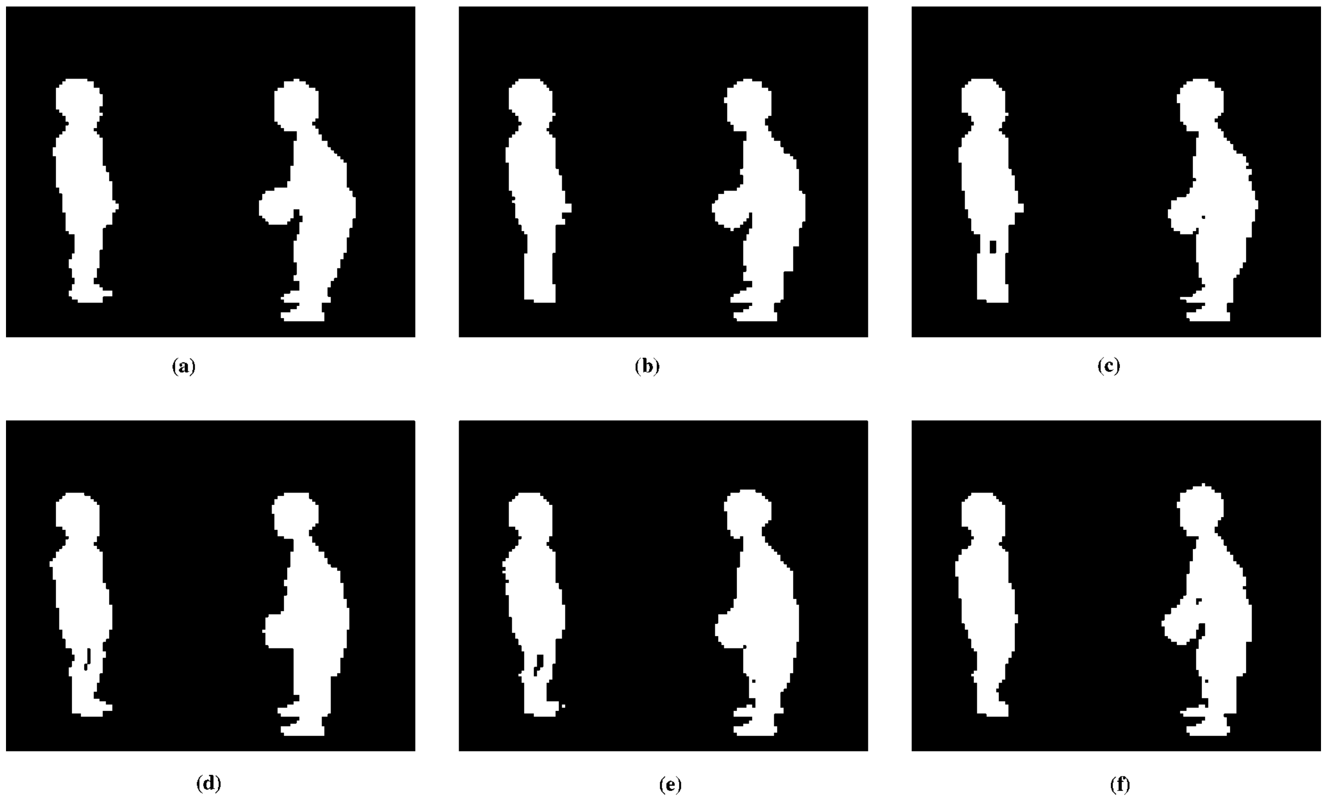


Fig. 15. Decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500,\ r = 15$): (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

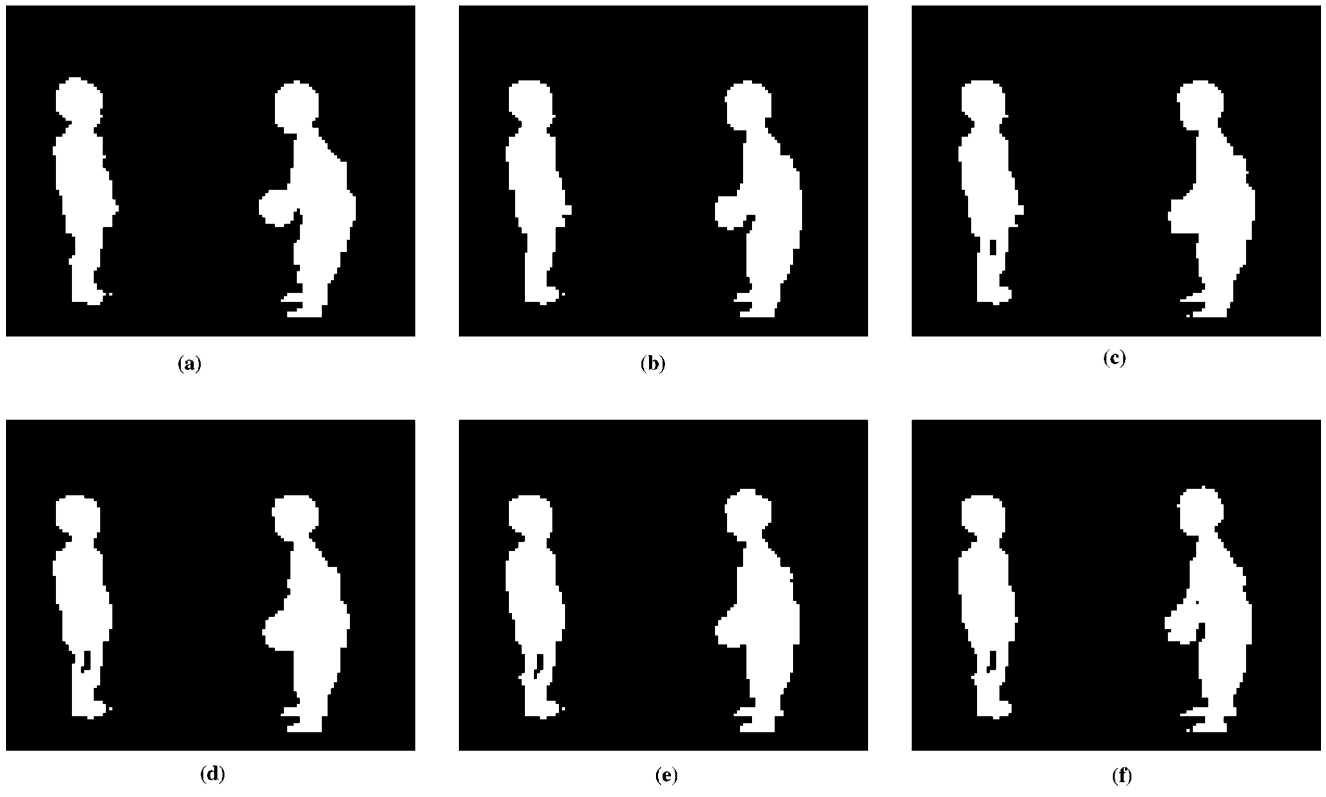


Fig. 16. Low-bit-rate-tuned decoded outputs (MPEG-4 shape decoder $B_{virtual} = 500,\ r = 12$): (a) first frame, (b) second frame, (c) third frame, (d) fourth frame, (e) fifth frame, and (f) sixth frame.

imum buffer. 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 $B_{virtual}$. Figs. 12 and 13 explain these facts.

Complicated shape data/pattern such as the ones found in Children sequence usually have a higer "break down" threshold. Therefore, an encoder requires a higher range of the channel rate to work as shown in Figs. 14 and 15. Since the Children sequence is more complicated than Akiyo, a higher channel rate should be given.

Fig. 16 depicts the results of the low bit rate tuned algorithm. If we consider $r = 12$ case (a relatively low bit rate), we see some quality difference between Figs. 14 and 16. The contour of the trousers can not be seen in the left child's legs in Fig. 14, while it is obvious in Fig. 16. The contour of the trousers looks blocky in the right child's legs in Fig. 14, the contour of the trousers looks blocky, while it is quite smooth in Fig. 16. An important aspect is that in a very low channel rate the encoding buffer fluctuates (sometimes it will break 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.

VII. Concluding Remarks

In this paper, 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 substeps. 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 using only CR-enable. Rate control by CR-only usually reduces the efficiency of encoders [7]. Therefore, 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.

References

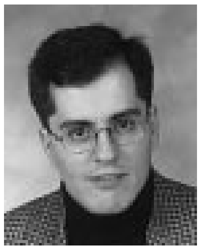
[1] *MPEG-4 Video Verification Model Version 8.0,* ISO/IEC JTC1/SC29/WG11 Std., July 1997.
[2] *Final Committee Draft Coding of Audio-Visual Object: Visual,* ISO/IEC JTC1/SC29/WG11 Std., Sept. 1998.
[3] T. Ebrahimi, "MPEG-4 video verification model: A video encoding/decoding algorithm based on content representation," *Signal Process.*, vol. 3, no. 1, pp. 26–40, June 1997.
[4] A. K. Katsaggelos, L. P. Kondi, F. W. Meier, J. Ostermann, and G. M. Schuster, "MPEG-4 and rate-distortion-based shape coding techniques," *Proc. IEEE*, vol. 86, pp. 1126–1154, June 1998.
[5] A. Ortega, "Optimization techniques for adaptive quantization of image and video under delay constraints," Ph.D. dissertation, Columbia Univ., New York, 1994.
[6] A. Ortega, K. Ramchandran, and M. Vetterli, "Optimal trellis-based buffered compression and fast approximations," *IEEE Trans. Image Processing*, vol. 3, pp. 26–40, Jan. 1994.
[7] J. Ostermann, "Efficient encoding of binary shapes using MPEG-4," in *IEEE Int. Conf. Image Processing '98*, vol. 3, Oct. 1998, pp. 295–298.
[8] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video," *IEEE Trans. Image Processing*, vol. 3, pp. 533–545, Sept. 1994.
[9] K. Ramchandran and M. Vetterli, "Rate distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility," *IEEE Trans. Image Processing*, vol. 3, pp. 700–704, Sept. 1994.
[10] K. Rijkse, "ITU standardization of very low bitrate video coding algorithms," *Signal Process.*, pp. 553–565, July 1995.
[11] G. M. Schuster and A. K. Katsaggelos, "An optimal ploygonal boundary encoding scheme in the rate distortion sense," *IEEE Trans. Image Processing*, vol. 7, pp. 13–26, Jan. 1998.
[12] H. Sun, W. Kwok, M. Chien, and C. H. Ju, "MPEG coding performance improvement by jointly optimzing coding mode decisions and rate control," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 449–458, June 1997.
[13] A. Vetro, H. Sun, and Y. Wang, "MPEG-4 rate control for multiple video objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 186–199, Feb. 1999.

Jae-Beom Lee (S'96–M'99) was born in Pusan, Korea, in 1965. He received Ph.D. degree in electrical engineering from Columbia University, New York, in 2000.

He was a Graduate Research Assistant with ADVENT group, an industrial affiliates program at Columbia University where his main interest was various aspects of MPEG-4 video coding. He currently leads a codec group at SolidStreaming, New York, where he is charge of algorithm/firmware development for low- and high-end wireless media streaming devices. He was with C-Cube Microsystems (New York Design Center) from 1999 to 2001, where he focused on design/implementation of MPEG transcoder engine for heterogeneous networks. His main contribution was the implementation of firmware/microcode algorithms for the high-performance two-chip MPEG-2 transcoder with a dual stream preprocessor. In 1998, he was a Summer Technical Employee with the Video Compression Systems Lab, Sarnoff Corporation, Princeton, NJ, where he was involved in MPEG-2 compressed domain logo insertion. From 1995 to 1999, he was a Graduate Research Assistant with the Center for Telecommunications Research (ADVENT project), Columbia University. From 1990 to 1991, he was a Member of Technical Staff with the Advanced Television Group, Samsung Electronics R&D Center, Suwan, Korea, working on subband-based HDTV proposals. His current research interests include a broad aspect of MPEG-2/MPEG-4 video coding applications. In particular, he is interested in very-low-bit-rate video coding technology and its systems issues such as in the Internet and/or wireless networks. He has contributed to compressed-domain video manipulation with a special emphasis on compressed-domain filtering techniques.

Jin-Soo Cho (S'01) was born in Seoul, Korea, in 1969. He received the B.S. degree in electronics engineering from Inha University, Inchon, Korea, in 1994, and the M.S. degree in electrical engineering from Columbia University, New York, in 1998. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, where he is a Graduate Research Assistant with the Bioengineering Group.

His current research interests include video signal processing and compression, and biomedical image processing with emphasis on motion analysis.



Alexandros Eleftheriadis (S'88–M'95) was born in Athens, Greece, in 1967. He received the Diploma degree in electrical engineering and computer science from the National Technical University of Athens, Athens, Greece, in 1990, and the M.S., M.Phil., and Ph.D. degrees in electrical engineering from Columbia University, New York, in 1992, 1994, and 1995, respectively.

Since 1995, he has been on the faculty of Department of Electrical Engineering, Columbia University, currently as an Associate Professor, where he is leading a research team working on media representation, with emphasis on multimedia software, video signal processing and compression, video communication systems (including video-on-demand and Internet video), and mathematical fundamentals of compression. He is also Co-Principal Investigator of the ADVENT Project, an industrial affiliates program at Columbia University that is performing research on all aspects of digital video representation, communication, and description. He is a member of the ANSI NCITS L3.1 Committee and the ISO-IEC JTS-1/SC29/WG-11 (MPEG) Group, which develops national and international standards for audio and video coding. He has served as the Editor of the MPEG-4 Systems (ISO/IEC14449-1) specification, has authored more than 60 publications in international journals and conferences, and holds ten patents (with nine more pending). He is on the editorial board of the *Multimedia Tools and Applications Journal*, and has served as a Guest Editor, Committee Member, and Organizer for several international journals and conferences.

Dr. Eleftheriadis is a member of ACM, AAAS, and the Technical Chamber of Greece, and is the recipient of a National Science Foundation CAREER Award.