MPEG-2 Streaming of Full Interactive Content

Kostas E. Psannis, Member, IEEE, Marios G. Hadjinicolaou, Member, IEEE, and Anargyros Krikelis, Senior Member, IEEE

Abstract—In this paper, we present an efficient method for supporting full interactive functions of MPEG-2 video streaming. The method is based on storing multiple differently encoded versions of the video stream at the server. A normal version is used for normal playback, while several other versions are used for fast/slow/forward (rewind)/rewind at variable speedups. Full interactive functions are produced by encoding every Nth frame of the original uncompressed movie as a sequence of I-P(M) frames. Mechanisms for controlling the interactive streams are also presented and their effectiveness is assessed through extensive simulations. Traditional interactive functions are supported with a minimum of additional resources on the server/network bandwidth and decoder complexity.

Index Terms—Digital video cassette recording (VCR), interactive services, MPEG-2 video, streaming video, video compression.

I. INTRODUCTION

DIGITAL video cassette recording (VCR) functionality enables quick and user-friendly browsing of multimedia content, thus making it a highly desirable feature in streaming video applications. Interactive access to video content is generally defined as a program or service controlled by the user and which can affect the content itself, the presentation manner of the content, or the presentation order of the content. Full range of interactive functions include play/resume, stop, pause, jump forward (JF)/jump backward (JB), fast forward (FF)/fast rewind FR), slow down (SD), and slow reverse (SR), rewind [1]. The difficulty of supporting interactivity varies from one interactive function to another. A stop or pause followed by resume is relatively easy to support since there is no requirement for more bandwidth than is already allocated for normal playback. However, fast scanning [(FF), (FR), (JF), (JB)] functions involve displaying frames at multiple times the normal rate. Transporting and decoding frames at such high-rate is prohibitively expensive and is not feasible using the hardware decoders available today or in the foreseeable future.

MPEG-2 also provides tools for “scalable” coding where useful video can be reconstructed from pieces of the total bitstream. Various scalable video coding techniques have been developed rapidly in the past decade. Spatial and temporal scalable video coding techniques that provide video at different resolutions and frames rate, were accepted in some main video coding standards such as MPEG-2, MPEG-4, and H.263 [2]–[4].

MPEG-2 structure allows a simple realization of forward (normal)-pay operation but imposes several additional constraints on the other interactive functions. In recent years several techniques for supporting interactive operations for MPEG-2 code video streaming applications have been devised [5]–[13]. In [5], [6] interactive functions are supported by dropping parts of the original MPEG-2 video stream. These approaches introduce visual discontinuities during the interactive mode, due to the missing video information. Alternatively interactive functions can also be supported using separate copies of the movie that are encoded at lower quality of the normal playback copy [7], [8]. In these cases, there is no significant degradation in the visual quality. However, the number of pre-stored copies of the movie limits the speed-up granularity. Other conventional schemes that support interactive functions require that frames are displayed at a rate much higher than the normal playback (for example, 90 fps [9]), or involves downloading parts of the video data in a player device located at the customer premises so that the customer can view without further intervention form the network [10]. In the latter case the downloading can be done prior to viewing. Other approaches [11], [12] address only the issue of reverse functions of MPEG-2 video streams; specifically, they describe methods of transforming an MPEG-2 I-B-P compressed bit stream into I-B and I-P(M) bit streams at client, respectively. However such approaches require much higher decoder complexity to perform the transformations and higher storage at the client. Some researchers suggested a model in which the server can partition each video stream into two substreams (a low resolution and a residual component stream) in order to support interactivity [13]. During the interactive mode, only the low-resolution stream is transmitted to the client. This reduces the amount of data that needs to be retrieved and sent to the client, although it still introduces extra seek and latency overhead on the load of the server. Note that none of the methods mentioned above fully address the problem of supporting interactive operations with minimum additional resources at the load of the server/network bandwidth and decoder complexity. In addition they support limited interactive functions.

In this paper, we propose a very efficient technique for supporting full-range of interactive operations in an MPEG-2 video streaming system based on encoding separate copies of the same movie. By encoding separate copies of the same movie the full range of interactive functions is supported with minimum additional resources. The corresponding interactive version is obtained by encoding every Nth (i.e., uncompressed) frame of the original movie as a sequence of I- P(M)-frames using different

Manuscript received December 27, 2004; revised July 15, 2005. This paper was supported in part by British Council (British Chevening/Sponsored by the British Government). This paper was recommended by Associate Editor E. Izquierdo.

K. E. Psannis and M. G. Hadjinicolaou are with the School of Engineering and Design, Electronic and Computer Engineering, Brunel University, Uxbridge UB8 3PH, U.K. (e-mail: mobility2net@gmail.com; Kostas.Psannis@brunel.ac.uk; Marios.Hadinicolaou@brunel.ac.uk). A. Krikelis is with Altera European Technology Centre, Buckinghamshire HP12 4XF, U.K. (e-mail: akrikeli@altera.com).

Digital Object Identifier 10.1109/TCSVT.2005.859933


\[
I_o \quad B_1 \quad B_2 \quad P_3 \quad B_4 \quad B_5 \quad I_6 \quad B_7 \quad B_8 \quad P_9
\]

(a)  
\[
P_3 \quad I_6 \quad P_9 \quad I_{12} \quad P_{15} \quad I_{18} \quad P_{21} \quad I_{24} \quad P_{27}
\]

(b)

Fig. 1. FF mode for Fs = 6. (a) Normal play. (b) Transmitted frames.

The group of pictures (GOP) pattern of the normal version. We analyze the effects of performing VCR functionality of the original bit stream by formulating a mathematical model in order to establish the constraints of interactivity. Mechanisms for controlling the interactive streams are also investigated to assess the overheads required in supporting interactive operations.

The paper is organized as follows. In Section II, the mathematical model is formulated accompanied by simulation results establishing the overheads on the server load, network bandwidth, and decoder complexity during the interactive mode. In Section III, the preprocessing steps required to support full interactive operations with minimum additional resources are detailed. The additional storage required to support full-range of interactive functions is described in Section IV. Finally, conclusions are discussed in Section V.

II. OPERATION DURING FAST FORWARD MODE

Assume that \(I_{i} \) is the starting point of the fast playback mode and frame-skipped (Fs) is the number of skipped frames. Consider the case in which the current frame is \(P_3 \) and frame \(P_9 \) is the following frame to be displayed during the fast playback for Fs factor equals to 6. Actually there is no need to send any of the B-frames because for the given Fs factor the displayed P-frames can be decoded only from the preceding I- or P-frames. Fig. 1(b) depicts the number of required frames to be transmitted in order to display correctly the P-frames.

It is valuable at this stage to develop a mathematical model in order to show the effect of the FF mode on the server load/network bandwidth and decoder memory. It is logical to assume that the start point of the FF mode is an I-frame. This is due to the fact that the I-frames will not cause an unpleasant effect in viewing because they are decoded independently. For simplicity but without loss of any generality it can be assumed that \(Fs \leq N \), where \(N \) is the distance between two successive I-frames, defining a GOP. \(N \) can be defined as follows:

\[
N = \begin{cases} 
\alpha \times M, & M > 0, \alpha > 0, & \text{I-P frames} \\
\alpha, & M = 1, \alpha > 0, & \text{I-P frames}
\end{cases}
\]

where \(M \) is the distance between two successive P-frames and \(\alpha \) is nonnegative constant. Note that, after \(Fs/\lambda \) GOPs the frame to be displayed will again be an I-frame, where \(\lambda = \text{greatest_common_divisor}(Fs/N) \). The total number of I-P-B-frames in the transmission/decoding pattern (the pattern is repeated every \(Fs/\lambda \) GOP) can be defined as follows.

\[
TF = \frac{Fs}{\lambda} \times N
\]

The number of I-frames in the transmitted/decoded pattern can be defined as follows:

\[
I = \frac{TF}{N} \times \frac{Fs}{\lambda}
\]

Case I: \(Fs = \text{equal to} N \)

In this case all the P- and B-frames are dropped and only the I-frames are transmitted/decoded and displayed. Hence every \(r \)th frame is an I-frame.

Case II: \(Fs = \text{a multiple of} \ M \ (Fs = \beta \times M, \beta \geq 1) \)

In this case all the B-frames need not to be transmitted/decoded and displayed. Note that \(\lambda = \text{greatest_common_divisor}(Fs/M,N/M)=\text{greatest_common_divisor}(\beta,\alpha) \). This is because \(M \) is the distance between a P-frame and the immediately preceding I- or P-frame. The numbers of transmitted I- and P-frames, every \(\beta/\lambda(M) \) GOP are: \(N = \beta/\lambda(M) \) and

\[
P = \beta/\lambda(M) \times (\alpha - 1) - P_{\text{skipped}}
\]

where \(P_{\text{skipped}} \) defines the number of nontransmitted P-frames. A P-frame needs not to be sent unless the successive P-frames are not displayed at the decoder. The number of the P-frames which need not be sent in the \(m \)th GOP, where \(1 \leq m \leq \beta/\lambda(M) \), is \((\mu \times \alpha)mod\beta - 1 \), where mod stands for the modular operation. Hence, the total number of nontransmitted P-frames in the \(\beta/\lambda(M) \) GOP can be computed from the following equation:

\[
I_{\text{displayed}} = \frac{Fs}{\lambda} \times \frac{\alpha}{\lambda(M)} - 1
\]

Finally, the number of the discarded I-, P-frames can be computed as follows:

\[
P_{\text{discard}} = \frac{\beta}{\lambda(M)} - 1
\]

\[
P_{\text{discard}} = \frac{\beta}{\lambda(M)} \times (\alpha - 1) - \left(\frac{\alpha}{\lambda(M)} - 1\right)
\]

Case III: \(Fs = \text{is not multiple of} \ M \ (Fs = (\rho \times M) + 1, \rho \geq 1) \)

The numbers of transmitted I-, P-, B-frames, every \(Fs/\lambda \) GOPs are

\[
I = \frac{Fs}{\lambda}, \quad P = \frac{Fs}{\lambda} \times (\alpha - 1) - P_{\text{skipped}}
\]

and

\[
B = \frac{Fs}{\lambda} \times (M - 1) - B_{\text{skipped}}.
\]
In this case the number of nontransmitted P-frames can be computed as follows: \( P_{\text{skipped}} = P_1 + P_2 - P_{\text{required-}\text{I-B}} \), where \( P_{\text{required-}\text{I-B}} \) is the number of required P-frames in order to decode correctly a B-frame. Computer simulations can be used to determine the numbers of non-transmitted B-frames \( (B_{\text{skipped}}) \) and the nontransmitted P-frames \( (P_{\text{skipped}}) \) for different combinations of \( N, M, \) and \( Fs \) factor. The numbers of displayed frames, every \( Fs/\lambda \) GOP are

\[
I_{\text{displayed}} = 1 \text{(one intra-frame)}
\]

\[
B_{\text{displayed}} = B_{\text{transmitted}}
\]

and

\[
P_{\text{displayed}} = \frac{N}{\lambda} - I_{\text{displayed}} - B_{\text{displayed}}
\]

A GOP is used as the retrieval block; normally stored contiguously on disk. Assuming that every I/O cycle, 1 GOP (15 frames) is retrieved, the disk executes one rotation, seek and transfer. If four different frames from three GOPs (12 frames) and three frames form the next GOP are retrieved during each cycle, the disk must perform four rotations, seeks and retrievals for every 15 frames which increases the overhead (load) on the server to process FF retrievals for every 15 frames which increases the network bandwidth required for sending the video stream with respect to different \( Fs \) factors. The server switches between the various versions depending on the requested interactive function. Assume that I-frame is always the start point of interactive mode. Since I-frames are decoded independently, switching from normal play to interactive mode and vice versa can be done very efficiently. Note that only one version is transmitted at a given instant time.

The corresponding interactive version is obtained by encoding every \( N \)th (i.e., uncompressed) frame of the original movie as a sequence of I-frames \( (N_{\text{interactive}} = \text{variable}, M_{\text{interactive}} = 1) \). Effectively this results in repeating the previous I-frame in the decoder, enhancing the visual quality during the interactive mode. This is because it freezes the content of the I-type frames, reducing the visual discontinuities. Moreover P(Marionette) frames are produced between successive I-frames in order to maintain the same frame of normal play and achieve full interactive operations at variable speeds.

The full interactive functions can be supported as follows.

- FF/FR is an operation in which the client browses the presentation in the forward/backward direction with normal sequence of pictures. This function is supported by abstracting all the I-type frames of the original (uncompressed) movie in the forward/backward direction and encoding each frame as a sequence of I-P(M)-frames.
- JF/JB is an operation in which the client jumps to a target time of the presentation in the forward/backward direction without normal sequence of pictures. JF/JB oper-
ation is supported by skipping forward/backward some
I-type frames of the original (uncompressed) movie and
encoding each of the remaining frames as a sequence of
I-P(M) frames.
• Rewind operations can be supported by abstracting all
the I-type frames of the original (uncompressed) movie in
reverse order and generate P(M) frames as many as P- and
B in a GoPs of normal playback (N = N_interactive).
• SD/SR is an operation in which the video sequence is
presented forward/backward with a lower playback rate.
This function can be supported by abstracting all the I-type
frames of the original uncompressed movie in the for-
ward/backward order and generate P(M) frames as many as
P- and B-frames in a recording ratio (RR) of normal
playback (N_interactive = RR).

It is useful to derive a closed-form formula to show the
number of the supported speedups of the proposed method.
The speedups can be computed as follows:

\[
\text{Speedups} = \begin{cases} 
  \frac{N}{N_{\text{interactive}}}, & S_I = 0 \\
  S_I \left( \frac{N}{N_{\text{interactive}}} - \frac{1}{N} \right) + \frac{N}{N_{\text{interactive}}}, & 1 \leq S_I < \frac{T_F}{N_{\text{interactive}}}, \text{ (IF)} 
\end{cases}
\]

where \( S_I \) is the number of skipped sequential I-type frames and
\( \Omega = RR_{\text{NormalPlay}}/N. \) Fig. 5 depicts the number of sup-
ported speedups (SpUsps) as a function of \( N_{\text{interactive}} \) for var-
ious numbers of skipped I-type frames. The graph in Fig. 5
shows that variable speedups can be achieved during the FF
mode by increasing/decreasing the number of P (M) frames
(\( N_{\text{interactive}} \)). On the other hand, variable speedups in the IF
mode depends on the number of P(M) frames between succes-
sive I-frames and the number of skipped I-type frames (\( S_I \))
of the normal stream.

B. Rate Control Schemes

A common approach to control the size of an MPEG-2 frame
is to vary the quantization factor on a per-frame basis.
The amount of quantization may be varied. This is the mechanism
that provides constant quality rate control. The quantized
coefficients \( QF[u, v] \) are computed from the DCT coefficients
\( F[u, v] \), the quantization_scale, \( MQUANT \), and a quantiza-
tion_matrix, \( W[u, v] \), according to the following equation:

\[
QF[u, v] = \frac{16 \times F[u, v]}{MQUANT \times W[u, v]} .
\]

The quantization step makes many of the values in the co-
efficient matrix zero, and it makes the rest smaller. The result
is a significant reduction in the number of coded bits with no
visually apparent difference between the decoded output and
the original source data [18]. The quantization factor may be
modified by varying the quantization scale (MQUANT) or by
varying the quantization matrix (\( W[u, v] \)). To bound the size of
I-frames of the interactive mode, the encoder uses two prede-
fined (upper-lower) thresholds.

\[
\text{Threshold}^{\text{upper}} = I_{\text{average}} \times S_{\text{upper}}, \quad 0 < S_{\text{upper}} \leq 1 \\
\text{Threshold}^{\text{lower}} = I_{\text{average}} \times S_{\text{lower}}, \quad 0 < S_{\text{lower}} \leq 1
\]

An I-frame is re-encoded such that its size is between

\[
\text{Threshold}^{\text{lower}} \leq I_{\text{bytes \_Size}} \leq \text{Threshold}^{\text{upper}}. \quad (7)
\]

Computer simulation can be used in order to define the values
of the two thresholds. Note that the selected values depend
on the type of motion of the original video movie. In order to
achieve minimum additional storage/network bandwidth and
acceptable visual quality during the interactive mode the fol-
lowing values of the predefined thresholds have been selected
for the mobile video sequence (complex motion)

\[
\text{Threshold}^{\text{upper}} = I_{\text{average}} \\
\text{Threshold}^{\text{lower}} = (0.9) \times I_{\text{average}}.
\]

It should be emphasized that other sequences were also used
and the values of the thresholds were similar when the amount
of motion was comparable to that of the mobile sequence. Two dif-
ferent schemes can be used to initialize the quantization factor
when an I-frame is to be re-encoded. In the first scheme, when
an I-frame is to be re-encoded for the first time the encoder
starts with the last quantization value that was used in the en-
coding of the previous P-frame. The main problem with this
method is that the quantization value might be kept unneces-
sarily high resulting in low quality during the interactive op-
érations. Moreover the encoder in this scheme uses one prede-
fined threshold (\( I_{\text{bytes \_Size}} \leq \text{Threshold}^{\text{upper}} \)). In the second
scheme, the encoding algorithm tries to track the nominal quan-
tization value, which was used in encoding the same type of
frame in the normal version. In the first encoding attempt, the
encoder starts with the nominal quantization value that was used
in the same I-frame of the normal version. After the first
encoding attempt, if the resulting frame size is between the
two pre-defined thresholds (7), the encoder proceeds to the next
frame. Otherwise, the quantization factor (quantization_matrix,
\( W[u, v] \)) varies and the same frame is re-encoded. The quantiza-
tion matrix can be modified by maintaining the same value at
the near-dc coefficients but with different slope toward the higher
frequency coefficients. This procedure is repeated until the size
of the compressed frame is between the two predefined thresh-
olds (7). The advantage of this scheme is that it tries to produce
interactive operations with the same constant quality of normal
playback, but when it is not possible it minimizes the fluctua-
tion in video quality during the interactive mode. Fig. 6 de-
picts the variations in the quantization values for I-frames when
\( S_{\text{upper}} = 1, S_{\text{lower}} = 0.9, \) and \( N_{\text{interactive}} = 5. \) In this experi-
ment, the nominal quantization values for I- and P-frames are
8 and 10, respectively. The quantization factors for I-frames are
plotted versus the index of every frame in the interactive ver-
sion. Fig. 7 shows two matrices both with the same value at the
Fig. 6. Quantization values during encoding of an interactive version.

Fig. 7. Two normalized quantization matrices $W[u, v]$ both with MQUANT = 8. (a) $W[u, v]$ with low slope. (b) $W[u, v]$ with high slope.

near-dc coefficients but with different slope toward the higher frequency coefficients. In other words, the quantization scale is fixed MQUANT and the quantization matrix $W[u, v]$ varies.

C. Visual Quality

The two approaches can be constrained with respect to video quality using the peak signal-to-noise ratio (PSNR). We use the PSNR of the Y-component of a decoded frame. The PSNR is obtained by comparing the original raw frame with its decoded version with encoding being done using the proposed algorithms. Fig. 8 depicts the PSNR values for mobile trace. Both schemes achieve acceptable visual quality because the PSNR is sufficiently large. The average PSNR value for the 60 frames is 37.02 dB for the first scheme and 37.88 dB for the second scheme. Clearly, the second algorithm achieves better visual quality than the first one, but at expense of more re-encoding attempts.

D. Network Bandwidth Allocation

Since the network bandwidth usually is the highest concern, the video during the normal version is coded with all I-P-B frames in order to achieve high compression ratios for the transport over the network with minimum bandwidth resources. In addition it is desirable to support interactive operations with minimum requirement in the network bandwidth. The impact of the two-rate control algorithms in the average frame size during the interactive mode is illustrated in Fig. 9 for the mobile sequence. The bandwidth during the interactive mode for the 1st algorithm is 30 fps × Average(IP)Size × 8 bits/bits = 0.53 Mbps and for the second algorithm is 30 fps × Constant(IP) × Interactive Size × 8 bits/bits = 1.23 Mbps.

IV. STORAGE OVERHEAD

Generate separate copies for interactive operations come at the expense of extra storage at the server. The storage require-
ments of the interactive mode and the extra storages required by [7] and [8] are given by

\[
W_{IP(M)} = TF_{IP(M)} \times \text{Average } \{\text{IP(M)}\} \times \text{Size}
\]

\[
W_{PB(s)} = TF(s) \times \left\{ \frac{P_{\text{scan average}}(s)}{N} + \frac{P_{\text{scan average}}(s)}{M} \right\}
\]

\[
W_{TB(s)} = TF(s) \times \left\{ \frac{P_{\text{scan average}}(s)}{N} + \frac{P_{\text{scan average}}(s)}{M} \right\}
\]

where \(P_{\text{scan average}}, P_{\text{scan average}}\) are the average sizes in a scan version, \(s\) is the number of skipped uncompressed frames of the original movie and \(TF(s)\) is the total number of frames in the scan mode. Fig. 10 depicts the storages overheads of the interactive bit streams as a function of skipped uncompressed frames of the original movie. For \(s > 8\) the storage overhead of our proposed technique is 1/2 times less than the extra storages required by [7] and [8] for the two rate control schemes.

V. CONCLUSION

In this paper, we investigated the constraints of implementing an MPEG-2 coded video streaming system, which supports full interactive services. In order to overcome these additional resources we proposed the use of multiple differently encoded version of each video sequence. Each one of the differently coded sequences is obtained by encoding every \(N\)th frame of the original (uncompressed) sequence. This allows for support of fully interactive operations at variable speedups. All versions of the original sequence are stored in the system’s server. The server responds to an interactive request by switching from the currently transmitted version to another version. By proper encoding versions of the original video sequence, interactive functionality can be supported with considerably reduced additional storage, network bandwidth and decoder complexity. The only noticeable drawback of the approach is some variability in the quality of motion picture during fast-interactive access.

REFERENCES