MPEG-4 Content Editing System for Real-Time IP Environment

Yasuyuki Miura and Michiaki Katsumoto
Communications Research Laboratory (CRL), 4-2-1, Nukui-Kitamachi, Koganei, Tokyo, 184-8795, Japan
Tel: +81-42-327-6818, Fax: +81-42-327-6884
E-mail: {miu, katsumoto}@crl.go.jp

Abstract—We propose a real-time content editing system that allows the input of multiple audio-visual streams. Our system is designed so that 1) the system can provide a content based on live video from multiple locations; 2) streaming and display modules can join or depart the system while it is operating, and 3) the system can provide proper encoding, communication, and synchronization in real-time environments. Content can also be encoded adaptively in various environments to distribute high quality live video movies to be presented in other different environments. To do this the scene description of the content must be changed when a streaming module joins the system or departs from it because the number of streaming objects is increased or decreased. In this paper, we present an overview of the system design, research achievements, and future work.

Index Terms—Multimedia systems, Real-time content editing system, MPEG-4

I. INTRODUCTION

THE spread of broadband networks has enabled the streaming of increasingly large volumes of video content[1]. These advances have also led to the creation of a number of transmitting and receiving services and tools for broadband networks[2-6]. Although these can be used to transmit movies, they can only transmit one movie at a time.

We are developing a Real-time Content Editing System that can input and receive multiple audio-visual streams and simultaneously encode, edit, and deliver them[7-8]. This system is designed with an editing module, some streaming modules, and display modules. The system makes it possible to compose, encode, and decode MPEG-4 [9] audio-visual objects, individual visual MPEG-4 objects, over a TCP/IP distributed environment. The system has the following features.

1. Users who are providing audio-visual objects can be involved in the editing of a scene.
2. The system can provide appropriate real-time environments for limited resources.

To enable users who are providing audio-visual objects to be involved in the editing of a scene, the system lets each transmitter and receiver freely join or depart from the system while it is operating and the system lets users rewrite the scene. To meet those requirements, we propose communication protocols for the join/depart modules.

Because MPEG-4 requires a lot of CPU power when inter-frame encoding algorithms are used, encoding live video data in real-time requires PCs with high processing speeds. However, while many users will have high-performance hardware, others will not. Thus, we propose an algorithm to select a suitable encoding policy.

In addition, an audio stream has to be encoded and delivered with higher accuracy than does a video stream because audio data becomes inaudible when it is delivered inaccurately [10]. Thus, a method of allocating CPU power that appropriately handles the encoding and synchronization of both video and audio data is needed. We address some issues of audio encoding and transmitting in our system.

II. REAL-TIME CONTENT EDITING SYSTEM

A. System Structure

Figure 1 is a schematic of our system. It consists of streaming, display, and editing modules. The editing module handles requests to send multicast audio-visual data to the streaming module; it also generates and sends scene descriptions. The streaming modules handle the encoding of DV (Digital Video) images from video cameras into the MPEG-4 format, and multicasting them to multiple display modules via an IP network. The display modules handle the display of edited movies that are based on MPEG-4 standards.

B. Module Structure

The modules are structured as follows.

1) Editing module

Figure 2 shows the editing module structure. An editing module starts sessions and transmits the BIFS and object descriptors. When a session starts, an editing module transmits a scene stream that includes a scene description and an Object Descriptor stream (“OD stream” in Figure 2) to display modules, and transmits an Acknowledgement to request transmission of AV (Audio-Visual) streams. BIFS control blocks change the scene based on request signals from a streaming module or display module, and send the OD ID to an object descriptor control block to transmit related ODs.

2) Streaming module

The structure of the streaming module is shown in Figure 3. Data encoded for MPEG-4 video in DV-MPEG transducers are divided into SL (sync layer) packets and delivered to display modules via UDP/IP. When a streaming module joins or departs from the system, streaming-module-join or
streaming-module-depart signals are generated in a request signal generation unit. Then the signal is sent to editing modules.

3) Display module

The display module structure is shown in Figure 4. Display modules include an MPEG-4 player and a request signal generation unit. The MPEG-4 player displays scenes based on object streams and BIFS. The request signal generation unit generates display-module-join or display-module-depart signals.

C. Stream Transmission

In our system, all streams are packetized as SL packets and transmitted via UDP/IP. Display modules of our system can join or depart the system while it is being operated. The outline of each event is as follows:

- Streaming module: Join (Fig. 5(a))
  1. A streaming module sends a Join-Streaming-module Request and information about the stream properties such as the width and height of images and the frame rate to an editing module.
  2. The editing module that receives this request returns a usable MG (Multicast Group) to the streaming module.
  3. The streaming module that receives MG returns an agreement signal if it is usable, or disagreement signal if it is not. If the streaming module returns a disagreement signal, then the editing module returns another MG. These processes are repeated until the streaming module returns an agreement signal.

- Departing module: Depart (Fig. 5(b))
  1. A streaming module sends a Depart-Streaming-module Request to an editing module. The editing module deletes the corresponding entry from the streaming module list.
2. The editing module processes the delete-a-scene procedure and sends a BFS command to display modules. Then it sends MG and ES_ID to delete to the display module.
3. Display modules that receive the MG and ES_ID delete the corresponding entry from the stream map table, then depart MG. Display modules that have finished leaving return a *finish departing signal*.
4. The editing module that receives a finish leaving signal sends *finish leaving signal* to streaming module when it receives signal from all display modules.
5. The streaming module that receives the signal stops distributing.

![Diagram](image)

(a) Streaming module: Join

(b) Streaming module: Depart

**III. DV-MPEG TRANSDUCER IN THE STREAMING MODULE**

**A. Requirements**

In our system, DV images can be distributed by using a portable PC, or an image from a meeting room can be distributed by using a high-performance computer. If our system must be used for the first case, a high-performance server cannot be used, so it is impossible to run complex encoding. In the latter case, the system can perform complex encoding, but if the server’s performance is insufficient, the system cannot fully encode the image. MPEG-4 encoding in a streaming module has a large computing cost, it is important to select the proper algorithm to provide the necessary quality for movies. Thus, the following problems are encountered for video and audio encoding.

1) **Video Encoding**

When a module transmits still images, such as a human face, transmitting at a lower frame rate may not be a serious problem. However, transmitting images containing quickly moving objects such as a sports scene, requires a certain minimum frame rate to prevent the movie quality from degrading. In the first case, reducing the frame rate is not such a serious problem, but in the second case, it is necessary to prevent a change in frame rate to simplify the encoding process itself.

![Diagram](image)

Fig. 6. Structure of DV-MPEG transducer

2) **Audio Encoding**

   In a streaming module, video and audio data are encoded at the same time. Because of a serious constraint in audio encoding, the following issues must be considered.

   - For interactive communications, latency must be reduced. To reduce latency, the priorities of video and audio encoding process are controlled.

   - Because the audio data have higher importance than the video data, they must be sent preferentially.

**B. Structure of the System**

1) **Video Encoder**

   The structure of the video-part of a DV-MPEG transducer for I-VOl and P-VOl is shown in Figure 6. The entire encoding process is divided into encoding segments, and all the encoding segments of Figure 6 are classified into basic and additional classes. A basic segment is defined as a minimum requirement, and the additional segments are those to be processed only when the transducer has surplus encoding time. Encoding segments framed in solid lines in Figure 6 are basic segments. They are the minimum segments needed for I-VOl encoding. AC/DC prediction, inverted quantization, IDCT, motion estimation, and motion vector encoding are additional segments. AC/DC prediction is used to reduce the number of I-VOl and P-VOl bits. The others are used to reduce the number of P-VOl bits.

   In our system, the transducer tracks the processing time for each segment and predicts future encoding times. It then decides whether P-VOl is to be used and how many P-VOl’s are used in the total VOP with a constraint of up to R frames per second. We have previously shown that our VOP decision algorithm preserves the frame rate of R in various conditions for various equipment.

2) **Audio Encoder**

   For encoding MPEG-4 audio, three methods have been used: Time Frequency mapping (T/F) as audio encoding for multi-purposes, Code Excited Liner Prediction (CELP) for voice data, and Parametric Encoding for low-bit encoding. T/F has two algorithms: Advanced Audio Coding (AAC), which has
been standardized for MPEG-2, and TwinVQ. CELP is the encoding method for voice data. The encoding time of this algorithm is shorter than that for T/F. Parametric encodings are the encoding algorithms for lower bitrates. Harmonic Vector eXcitation Coding (HVXC) for voice data and Harmonic Individual Line with Noise (HILN) for music data are used for parametric encoding. In real-time communications such as virtual meetings, human voices must be transmitted. Therefore, we use CELP and HVXC in our system, which provides an encoding algorithm for voice data, plus multipurpose encoding of T/F.

3) Experiment of Audio Encoding

We measured encoding times of TwinVQ. We used a VAIO notebook PC (PCG-GRX90P, SONY) that had a 1.7-GHz PentiumIV processor and a 4-pin IEEE1394 port (a high-performance notebook PC). Encoding time and the number of bytes of encoded data for 25 seconds of music data are shown in Table 1. In this experiment, the parameter “bit rate” was changed from 8000 bits/s to 64,000 bits/s. The results in Table 1 show that it is possible to drive an MPEG-4 audio decoder over a PC in a real-time environment.

Because audio data must be encoded with more severe real-time constraints than video data, audio must be given a higher priority. An algorithm for object allocation has been proposed for a cluster system [11]. Therefore, it should be possible to control the priority appropriately using the proposed algorithm. However, an original technique is required when using PCs because they do not have multiple clusters. In particular, one must consider calculating the encoding time for both audio data and video data, and then giving higher priority to audio encoding. Let Total be the average available encoding time of audio and video data for one frame and Taudio be the average encoding time of audio data that is synchronized with one frame of video. Then the average encoding time of 1 VOP of video data Tvideo is

\[ Tvideo \leq T_{total} - T_{audio} \]  

(1)

Simultaneous encoding of video and audio data is processed with the following approach.

1. Encode the audio input from a video camera and track the average encoding time for one frame.
2. Continuing the encoding of audio input and encode the video input by using a VOP decision algorithm[7] with the target encoding time Tvideo obtained by equation (1). At the same time, continuously encode and track the audio input time and correct the value of Tvideo.

### Table 1. Encoding Time of TwinVQ algorithm

<table>
<thead>
<tr>
<th>Bit Rate</th>
<th>Encoding Time</th>
<th>Number of Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>8000 bits/s</td>
<td>16.7s</td>
<td>27,030 bytes</td>
</tr>
<tr>
<td>16,000 bits/s</td>
<td>23.7s</td>
<td>53,258 bytes</td>
</tr>
<tr>
<td>32,000 bits/s</td>
<td>26.6s</td>
<td>106,556 bytes</td>
</tr>
<tr>
<td>64,000 bits/s</td>
<td>29.1s</td>
<td>212,306 bytes</td>
</tr>
</tbody>
</table>

### IV. NETWORK AND DISPLAY MODULE

A. Communicating with Modules

Because audio data have a severe constraint, giving them preference over video data in the network helps participants to communicate interactively. Thus, we chose an IPv6 protocol that allows real-time communication. The priority of packets can be assigned in IPv6. Therefore, it is possible to assign higher priority to audio objects.

B. Synchronizing Objects

In our system, various encoding times are used for VOP. Also, some objects are transmitted from multiple streaming modules at a number of points. Therefore, display modules must be appropriately synchronized. To synchronize objects, a timing model using the Sync Layer (SL) in an MPEG-4 system standard can be used. For future work, we will investigate a synchronizing method to reduce latency.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an overview of a proposed real-time content editing system, some problems we had designed the system, and the achievements and future issues of our research. Our system can input and receive multiple audio-visual streams and simultaneously encode, edit, and deliver them. The system is designed with an editing module, some streaming modules, and display modules. We decided the architecture of the system and proposed a VOP decision algorithm for real-time processing. In addition, we identified the following three points as future issues.

- Detailed specifications of a request handling script
- Specific video and audio compound encoding algorithm
- Synchronization of objects with SL.

Now, we are constructing a prototype system. Then, we will implement a VOP decision algorithm and resolve the three issues above.

OPL REFERENCES