Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Resynchronization and remultiplexing for transcoding to H.264/AVC*

ZHOU Jin, XIONG Hong-kai, SONG Li, YU Song-yu‡

(*Institute of Image Communication and Information Processing, Shanghai Jiao Tong University, Shanghai 200030, China*)

E-mail: zhoujin@sjtu.edu.cn; xionghongkai@sjtu.edu.cn; songli@qansoft.com; syyu@cdtv.org.cn

Received Dec. 9, 2005; revision accepted Feb. 19, 2006

Abstract: H.264/MPEG-4 AVC standard appears highly competitive due to its high efficiency, flexibility and error resilience. In order to maintain universal multimedia access, statistical multiplexing, or adaptive video content delivery, etc., it induces an immense demand for converting a large volume of existing multimedia content from other formats into the H.264/AVC format and vice versa. In this work, we study the remultiplexing and resynchronization issue within system coding after transcoding, aiming to sustain the management and time information destroyed in transcoding and enable synchronized decoding of decoder buffers over a wide range of retrieval or receipt conditions. Given the common intention of multiplexing and synchronization mechanism in system coding of different standards, this paper takes the most widely used MPEG-2 transport stream (TS) as an example, and presents a software system and the key technologies to solve the time stamp mapping and relevant buffer management. The solution reuses previous information contained in the input streams to remultiplex and resynchronize the output information with the regulatory coding and composition structure. Experimental results showed that our solutions efficiently preserve the performance in multimedia presentation.

Key words: Transport stream (TS), Remultiplex, Time stamp, Transcoding, H.264/AVC
doi:10.1631/jzus.2006.AS0076 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

With its prominent coding efficiency, flexibility and error resilience, the H.264/MPEG-4 AVC standard, jointly developed by the ITU-T and the MPEG committees, appears highly competitive. Compared with various standards such as MPEG-1, MPEG-2, and MPEG-4, H.264/AVC offers an improvement of about 1/3 to 1/2 in coding efficiency with perceptually equivalent quality video (Thomas *et al*., 2003). These significant bandwidth savings open a great market to new products and services. As a result, the potential of H.264/AVC yields several possibilities for employing the transcoding and corresponding technologies (Hari, 2004). Especially, it induces immense demand for converting a large volume of existing multimedia content from other formats into the H.264/AVC format and vice versa.

The transcoding technologies also lead to the emergence of the corresponding technologies. For example, currently, MPEG-2 system coding is widely used in digital TV, DVD, and HDTV applications. It is specified in the program stream (PS) and the transport stream (TS) based on packet-oriented multiplexes (ISO/IEC JTC11/SC29/WG11, 1994). It is assumed that transport streams be the preferred format. It is of great interest that the video ESs in transport streams are transcoded to the H.264/AVC video ESs to save a great amount of bandwidth and resources. Transport streams provide coding syntax necessary and sufficient to synchronize the video and audio ESs, while ensuring that data buffers in the decoders do not overflow or underflow. Therefore,

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (No. 60502033), the Natural Science Foundation of Shanghai (No. 04ZR14084) and the Research Fund for the Doctoral Program of Higher Education (No. 20040248047), China

remultiplexing and resynchronization in system coding for the dedicated transcoding to H.264/AVC will become two key technologies.

In this work, we focus on the issues of remultiplexing and resynchronization in system coding after video transcoding. Unlike previously proposed remultiplexers which are usually implemented by hardware, this paper proposes a software solution. Our solution reuses the management and time information contained in the input streams instead of regenerating new information by hardware, to remultiplex and resynchronize the output information combining with the regulatory coding and composition structure. Consequently, it is cheap in the sense that it is simple and easy to implement, and could be adopted by desktop PCs or intermediate gateways servers with a minimum extra cost.

This paper is organized as follows. Section 2 describes the mechanisms in multiplexing. And Section 3 describes the architecture of our system. Section 4 analyzes the key technologies in the system. Experimental results are discussed in Section 5. We close the paper with concluding remarks in Section 6.

MULTIPLEXING MECHANISM

Although the semantics and syntax differ much in system coding of various coding standards, the main functions of the system part are similar: to multiplex different bitstreams and provide information that enable demultiplexing by decoders.

The transport stream (TS) combines one or more programs with one or more independent time bases into a single stream. Between the encoder and decoder, a common system time clock (STC) is maintained. Decoding time stamp (DTS) and presentation time stamp (PTS) specify when the data is to be decoded and presented at the decoder. A program is composed of one or more elementary streams, each labeled with a PID. Other formats such as program stream have similar synchronization mechanism (using time stamps) and also the management mechanism, which are sometimes even much simpler as there may be just one program in a program stream.

Therefore, we can apply these mechanisms for H.264/AVC in system layer. When multiplexing the H.264 bitstream into a transport stream, the H.264/AVC video bitstream can be regarded as an elementary stream and encapsulated in the transport stream with other elementary streams such as audio ES, Program Specific Information (PSI) data, etc., and the necessary time stamps are also inserted for synchronization. The entire system is illustrated in Fig.1 and can also be regarded as a remultiplexer consisting of three major parts: demultiplexer, transcoder and multiplexer.

ARCHITECTURE OF THE TRANSCODING SYSTEM

Demultiplexer

The main function of the demultiplexer is to extract the video bitstream and the necessary information from the input multiplexed stream. In our system, the demultiplexer first scans the transport stream to obtain the needed PSI. Second, the video ES is extracted from the transport stream according to the given PSI.

Meanwhile, the demultiplexer ought to store all the time stamps and their exact positions in the trans-

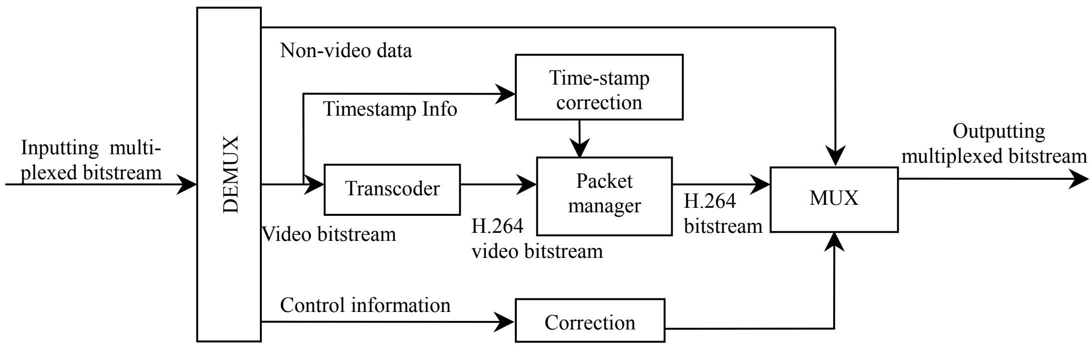

Fig.1 Architecture of the system

sport stream. To describe these positions, the demultiplexer can mark each frame in the video sequence with a unique number. For example, the first frame is marked 1, the second is 2, and so on.

Transcoder

The main function of the transcoder is to transcode the input video sequence to a H.264/AVC compliant video stream without any time information.

The video coding layer (VCL) of H.264/AVC is similar in concept to other standards such as MPEG-2 video. Moreover, the concept of generalized B-pictures in H.264/AVC allows hierarchical temporal decomposition. In fact, any picture can be marked as reference picture and used for motion-compensated prediction of following pictures independent of the coding types of the corresponding slices in H.264/AVC.

Multiplexer and the corresponding problems

At the input of the multiplexer are the newly-transcoded H.264/AVC video bitstream and the original multiplexed stream. The multiplexer will replace the encapsulated original video bitstream with the new generated H.264/AVC bitstream and output the transcoded multiplexed stream, which is, however, not as simple as it appears.

For example, in a transport stream, as an elementary stream is erased and another resized elementary stream enters, the PSI and the timing and spacing of PCR packets will be disturbed.

Furthermore, the input video sequence and the H.264/AVC sequence can possibly have different picture structures. Therefore, the time stamps of the former cannot be directly copied to the latter and time stamps correction step should be taken.

Finally, how to insert the data on video ES and other ESs into single transport stream needs consideration.

As discussed above, we will recommend our solutions for the four key technologies: DTS/PTS correction, PCR correction, PID mapping and data insertion (Wang *et al*., 2002).

OUR SOLUTIONS

PTS/DTS correction

Other coding standards may have their own video/audio synchronization mechanism, although many common standards use time stamps for synchronization. For example, the composition time stamp (CTS) and DTS in MPEG-4 are similar to the PTS and DTS in MPEG-2 respectively.

Assume TS_i^{d} and TS_i^{p} to be the DTS and the PTS of the *i*th picture, and $TS_i^{\mathrm{d}\prime}$ and $TS_i^{\mathrm{p}\prime}$ to be those of the *i*th H.264/AVC picture respectively.

The PTS/DTS correction between the *i*th picture and the *i*th H.264/AVC picture is done as follows:

Step 1: If PTS is present while DTS is not, the DTS value of the input picture is supposed to be equal to the PTS value.

$$TS_i^{\mathrm{d}} = TS_i^{\mathrm{p}}\text{, if } TS_i^{\mathrm{d}} \text{ is not provided.} \tag{1}$$

Step 2: As the decoding order is assumed to be the same as the encoding order, the DTS value of pictures with the same position should be equal.

$$TS_i^{\mathrm{d}\prime} = TS_i^{\mathrm{d}}. \tag{2}$$

Step 3: PTS correction.

The purpose of PTS correction is to compensate for the time delay due to the picture reordering. There are 3 conditions as the H.264/AVC picture may be an anchor picture (I- or P-picture), a B-picture, or a hierarchical B-picture.

(1) H.264/AVC I-picture or P-picture

There are still 2 conditions as the corresponding input video picture is an anchor picture or a B-picture:

(a) If the input picture is an I- or P-picture, $TS_i^{\mathrm{p}\prime}$ is calculated as follows:

$$TS_i^{\mathrm{p}\prime} = TS_i^{\mathrm{p}} - [(N_1 - N_2)/\,frame\_rate] \times frequency, \tag{3}$$

where N_1 denotes the number of B-pictures between the relevant input picture and the next MPEG-2 I- or P-picture and N_2 is the number of frames between the H.264/AVC I-picture and the next H.264/AVC I- or P-picture. *frame_rate* is the number of frames displayed per second. *frequency* denotes the frequency of encoder STC.

(b) If the input picture is a B-picture, $TS_i^{\mathrm{p}\prime}$ is calculated as follows:

$$TS_i^{p\prime} = TS_i^{p} + [(N_2 + 1) / frame\_rate] \times frequency. \tag{4}$$

(2) H.264/AVC B-picture

B-pictures are displayed immediately as soon as it is received. Consequently, $TS_i^{p\prime}$ is calculated as follows:

$$TS_i^{p\prime} = TS_i^{d\prime}. \tag{5}$$

Especially, DTS should not be inserted in B-picture as it is always equal to PTS.

(3) H.264/AVC hierarchical B-picture

If the pyramid coding is employed in the transcoder, hierarchical B-pictures will be present in the bitstream. $TS_i^{p\prime}$ is calculated as follows:

(a) If the input MPEG-2 picture is an I- or P-picture,

$$TS_i^{p\prime} = TS_i^{p} - \{[N_1 - (dis\_ord - enc\_ord)] / frame\_rate\} \times frequency. \tag{6}$$

(b) If it is a B-picture,

$$TS_i^{p\prime} = TS_i^{p} + \{[1 + (dis\_ord - enc\_ord)] / frame\_rate\} \times frequency, \tag{7}$$

where *dis_ord* and *enc_ord* are the display order and the encoding order of the H.264 hierarchical B-picture in a group of pictures (GOP). For example, if we code the sequence using the coding pattern I0-P8-RB4-RB2-RB6-B1-B3-B5-B7-P16-RB12-RB 10-RB14-B9-B11-B13-B15-P24…, where RB refers to a B slice only coded picture which can be used as a reference. See Fig.2. Then the *dis_ord* and *enc_ord* of

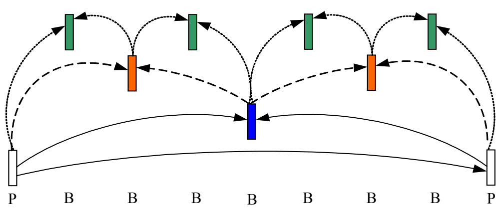

Fig.2 Level pyramid structure

RB4 are 4 and 2 while those of B1 are 1 and 5 respectively.

PID mapping

In a transport stream, PID is used to identify different elementary streams, as our multiplexer just exchanges the H.264/AVC video sequence for the video sequence. The PID of the input sequence can be assigned to the H.264/AVC video as its PID, and the PID of other elementary streams may stay the same. Therefore, the Program Association Table (PAT) and Program Map Table (PMT) are not necessary to make a change.

PCR correction and data insertion

As the video sequence is resized, the original PCR ought to be changed to accommodate the new bit rate. In (Bin and Klara, 2002), a software approach to reuse the PCRs was proposed. This approach scales the PCR packets' distance and achieves a fixed new bit rate. The remaining non-video stream packet can be simply mapped to the new position on the scaled output time line corresponding to the same time point in the original input time line. For example, if in the input stream, packet 3 is a frame header, packet 6 is an audio packet and packets 4, 5, 7 through 15 are video data from the same frame as packet 3. Since the video sequence is shrunk to its 1/2 size after transcoding, while the non-video data are not changed, the transport stream is approximately 2/3 of its bandwidth. After shrinking, packet 3 is positioned to slot 2, and packet 6 to slot 4. The empty slots such as slot 3, 5 through 10, are used to carry the other H.264/AVC video data. As the video data are about 1/2 of original data, the slots should be enough to carry the video frame. And the redundant slots can simply be padded with NULL packets to maintain a constant bit rate (CBR).

Resynchronization when the input TS is unsynchronized

An important problem is how to synchronize the video and audio stream if the input TS is not synchronized. We can solve this problem when the extractor marks the input video frame. For example, if the frame rate is 25 Hz and the video is 0.08 s (2 frames' time) later or earlier than the audio, we should mark the first frame −1 instead of 1 and so do the following frames. Then the third frame would be stamped by the first frame's time stamp and display at the time when the first frame used to. Consequently, the stream is synchronized again.

EXPERIMENTAL RESULTS

We applied our system to a transport stream whose scenario is a person's talk. The transport stream multiplexed an MPEG-2 video stream and an MP2 audio stream. We use libmpeg2 as the MPEG-2 decoder and JM 10.1 as the H.264/AVC encoder. In our experiments, the MPEG-2 video ES, which complies with a picture structure I P B B P B B…, is transcoded into an H.264/AVC stream with the structure I P B P B P B…. The MP2 audio ES remains the same. After transcoding, the H.264/AVC ES and MP2 ES and other data bitstreams are multiplexed again to produce the new TS. Remultiplexing and resynchronization are achieved by employing our solution in the multiplexer as discussed with the experimental results being satisfactory as expected.

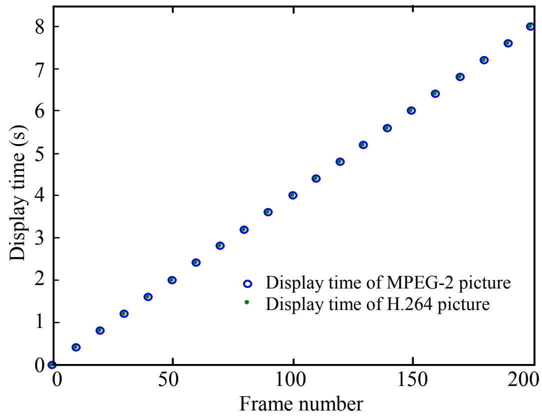

Fig.3 Comparison of display time

Fig.3 shows 20 pairs of display time of the frames sampled from the two bitstreams. It is clear that each pair of marked spots are approximate superposition, indicating that the transcoded video sequence is also temporal consistent with the audio just as the MPEG-2 video is. After analyzing the experimental results, we find that the maximum jitter among the 20 pairs of display time is less than 10 ms which assures the synchronization of video and audio after transcoding given the timestamp information provided by the original video bit stream are all correct and that the MPEG-2 sequence has already been synchronized.

Fig.4 shows the comparison between the two video ES by sampling them at the same points of time. As the MP2 audio ES does not change in the transcoding process, pictures sampled at the same time points of the two video ES are comparable.

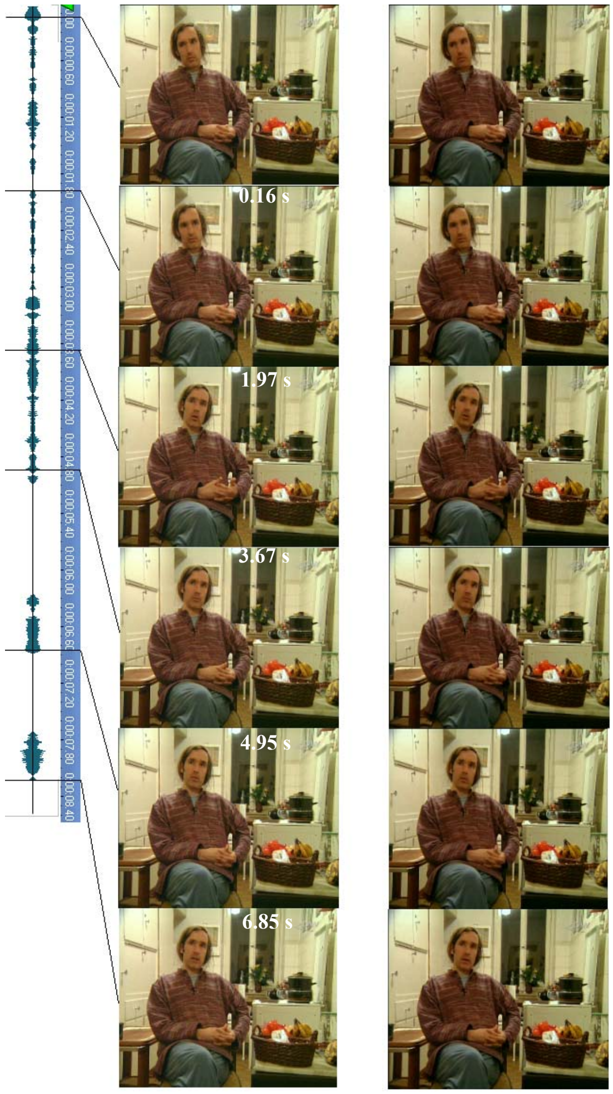

Fig.4 Comparisons of synchronization during the multimedia presentation

The leftmost part of Fig.4 is the time axis and the audio waveform of the MP2 mentioned above. Six pairs of pictures sampled from the MPEG-2 ES and H.264/AVC ES respectively are shown in the middle and the corresponding sampling time is listed in the right. Obviously, each two relevant pictures show great similarity which means that the resynchronization obtained properly after applying our method in the remultiplexing step.

CONCLUSION

In this work, we studied the problem of the bit rate being changed after transcoding to the H.264/AVC bitstream, control information and that time stamps in the multiplexed stream may be destroyed. We have proposed a software system and discussed the key technologies and have taken transport stream as an example. We studied the relationship between the DTS/PTS in the input and output TS to make DTS/PTS correction. DTS/PTS correction is also effective for transcoding with hierarchical B-pictures. The PID mapping method we used is simple and effective and a scaling method is employed to do the PCR correction and data insertion. Moreover, our system can easily synchronize the TS which was previously unsynchronized. Experimental results showed that with our solutions, the H.264/AVC bitstream is remultiplexed and well-synchronized with the audio bitstream.

References

Bin, Y., Klara, N., 2002. A Realtime Software Solution for Resynchronizing Filtered MPEG-2 Transport Stream. Proceedings of the IEEE Fourth International Symposium on Multimedia Software Engineering.

Hari, K., 2004. Issues in H.264/MPEG-2 Video Transcoding. Consumer Communications and Networking Conference (CCNC). The First IEEE, p.657-659.

ISO/IEC JTC11/SC29/WG11, 1994. Generic Coding of Moving Pictures and Associated Audio: Systems. ISO/IEC 13818-1.

Thomas, W., Gary, S., Gisle, B., Ajay, L., 2003. Overview of the H.264/AVC video coding standard. *IEEE Trans. on Circuits and Systems for Video Technology*, **7**:560-576.

Wang, X.D., Yu, S., Liang, L., 2002. Implementation of MPEG-2 transport stream remultiplexer for DTV broadcasting. *IEEE Trans. on Consumer Electronics*, **48**(2): 329-334. [doi:10.1109/TCE.2002.1010139]