3.4-2

Efficient Audio/Video Synchronization Method for Video Telephony System in Consumer Cellular Phones

Chanwoo Kim[1], Kwang-deok Seo[2], *Member, IEEE,* Wonyong Sung[3], *Member, IEEE,* and Soon-heung Jung[4]
[1]Carnegie Mellon Univ., USA, [2]Yonsei Univ., Korea, [3]Seoul National Univ., Korea, and [4]ETRI, Korea

Abstract- This paper presents an efficient audio/video synchronization method that is quite suitable for video telephony. In terms of computational complexity and simplicity of the software structure, the proposed algorithm shows improved results compared to the conventional synchronization algorithm. The derived audio/video synchronization algorithm is compactly expressed in a single decision rule. After simulations with real video telephony system based on Texas Instruments OMAP processor and Nucleus PLUS RTOS, we found the devised algorithm is quite suitable for consumer cellular phones with limited computational resources.

I. INTRODUCTION

Recently, many researchers in the field of hand-held consumer electronics have focused on incorporating various kinds of multimedia and supplementary services into cellular phones. Reflecting this trend, many of the contemporary cellular phones are implemented with applications like fingerprint recognition, speech recognition, video-telephony (VT) [2], and video-on-demand (VOD). Among them, with the recent advent of Wideband Code-Division Multiple Access (WCDMA) services in Korea, VT system has begun to draw significant attention. In order to transmit payload in real-time for VT systems, Real-time Transport Protocol (RTP) is usually employed. In pair with RTP, we usually employ RTP Control Protocol (RTCP) for the purpose of quality control. In order to synchronize between different media, we inspect every RTCP Sender Report (SR) to find out reference time corresponding to RTP timestamp conveyed on RTP packet.

In this paper, we propose an efficient audio/video (A/V) synchronization method. In this method, we do not need to process RTCP SR packet for synchronization. Additionally it does not require any floating-point operations or any divisions at all. Moreover, the decision criterion for synchronization can be compactly described just in a single equation. Through extensive simulations, the proposed algorithm shows noticeable advantages in terms of required computation load and simplicity of software structure.

II. AUDIO/VIDEO SYNCHRONIZATION ALGORITHM

A. Conventional Synchronization Algorithm

RTP has supplementary information like sequence number and RTP timestamp in its header to facilitate real-time transmission [3]. In the VT system, synchronization between audio and video data is a crucial issue, since audio and video data are transmitted in separate RTP streams. However, we cannot directly use RTP timestamp to synchronize data conveyed in different RTP sessions for the following two reasons. Firstly, RTP timestamp begins at a random number [3]. Thus, the initial RTP timestamps for audio and video sessions are different, even though they are actually sampled at the same time. Secondly, RTP timestamp increases in proportion to the sampling rate of media. Usually the sampling rates of audio and video data are quite different. Thus, the rates of increase in RTP timestamp for audio and video sessions are not the same. To circumvent these two problems, RTCP SR packets carrying both the RTP and the Network Time Protocol (NTP) timestamp are generally employed. NTP timestamp provides absolute time information specified by RFC1305 [4]. Fig. 1 illustrates the streams of RTP and RTCP packets for a certain media. Without loss of generality, let us assume that this media stream constitutes an audio session. For a specific RTP packet highlighted in Fig. 1, let us assume this RTP packet is located between i + 1 th and i + 2 th RTCP SR packets in time order. If we let T^A be the reference time in second corresponding to the RTP timestamp, M^A, of this specific RTP packet, we obtain the following relation:

$$\frac{T^A - T_{sr}^A(i+1)}{M^A - M_{sr}^A(i+1)} = \frac{T_{sr}^A(i+1) - T_{sr}^A(i)}{M_{sr}^A(i+1) - M_{sr}^A(i)}. \quad (1)$$

The superscript A in each term is added to show that they are related to audio session. In (1), $T_{sr}^A(i)$ and $T_{sr}^A(i+1)$ denote NTP timestamps conveyed by i + 1 th and i + 2 th RTCP SR packets, respectively. Even though NTP timestamp is a 64-bit integer [4], T^A, $T_{sr}^A(i)$, and $T_{sr}^A(i+1)$ in (1) are floating-point number in the unit of second. In the same manner, $M_{sr}^A(i)$ and $M_{sr}^A(i+1)$ are RTP timestamps carried with i + 1 th and i + 2 th RTCP SR packets, respectively. We can rearrange (1) to obtain the reference time, T^A as follows [7]:

$$T^A = T_{sr}^A(i+1) + \frac{T_{sr}^A(i+1) - T_{sr}^A(i)}{M_{sr}^A(i+1) - M_{sr}^A(i)}(M^A - M_{sr}^A(i+1)). \quad (2)$$

The same procedure can be applied to video session to obtain T^V, the reference time in second corresponding to RTP time stamp in video RTP packet. Comparing the obtained T^A and T^V values, we find out whether video data being decoded is relatively fast or slow compared to the audio data.

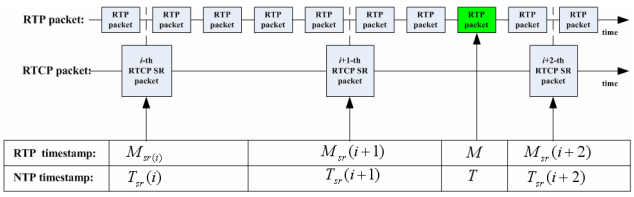

Fig. 1. RTP and RTCP packets.

0-7803-9459-3/06/$20.00 ©2006 IEEE

B. Proposed Synchronization Algorithm

If the sampling rate of audio data, R^A, is constant, we can simplify (1) into

$$T^A = T_{sr}^A(0) + \frac{M^A - M_{sr}^A(0)}{R^A}, \quad (3)$$

by noting that

$$R^A = \frac{M_{sr}^A(i+1) - M_{sr}^A(i)}{T_{sr}^A(i+1) - T_{sr}^A(i)}. \quad (4)$$

In the same manner, we can obtain reference time in second T^v, which corresponds to the RTP time stamp in a specific RTP packet in a video session by:

$$T^V = T_{sr}^V(0) + \frac{M^V - M_{sr}^V(0)}{R^V}. \quad (5)$$

By subtracting (5) from (3) and some arithmetic, we can obtain the following decision rule:

$$R^V M^A - R^A M^V \begin{array}{c} \text{Audio is too fast} \\ > \\ < \\ \text{Video is too fast} \end{array} \eta, \quad (6)$$

where the threshold, η, is given by:

$$\eta = R^A R^V (T_{sr}^V(0) - T_{sr}^A(0)) + R^V M_{sr}^A(0) - R^A M_{sr}^V(0). \quad (7)$$

We can easily compute the value of η after receiving the first SR packets in the audio and video sessions. Thus for every audio/video RTP packet, we only need to compute the left side of (6).

In (6), R^V and R^A can be obtained during the Session Description Protocol (SDP) [8] negotiation process, and as previously mentioned, they are integer values representing the sampling rates of each media. M^A and M^V are RTP timestamp contained in each RTP packet, which is a 32 bit integer value as specified in [3]. Since all of the R^V, R^A, M^A, and M^V values in (6) are fixed point numbers themselves, there is no need to utilize floating point operations at all. Additionally, (6) does not require any division operations unlike the case of (1), (2), and (4). Obviously, this is a clear advantage for embedded processors, which usually do not have floating point units. For ARM processors, avoiding division is also advantageous aspect. To implement (6), all we need to do is two fixed-point multiplications, one subtraction, and one comparison operation.

III. System Implementation and Experimental Results

The proposed method is incorporated in a prototype VT system. Fig. 2 shows the hardware structure of the developed system. We adopt H.263 video codec for video processing, and Qualcomm Code Excited Linear Prediction (QCELP) for audio processing. Video codec operates on top of TI Open Multimedia Application Platform (OMAP) 1510 processor [6], which incorporates an ARM925T core and a TMS320C5510 Digital Signal Processor (DSP). We adopt Qualcomm Mobile Station Modem (MSM) 5500 as a baseband modem. In this system, OMAP processor acts as the host processor, while MSM 5500 processor operates at the slave mode. For OMAP processor, we use Nucleus PLUS as an RTOS for the ARM core.

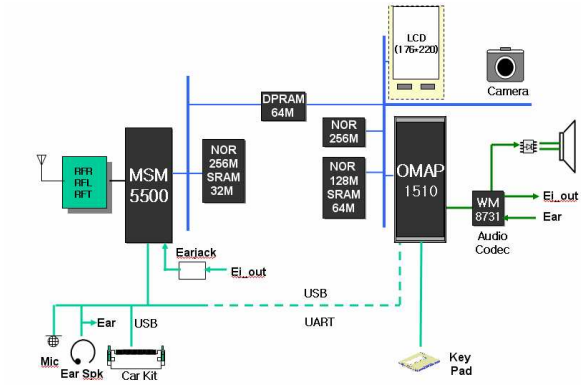

Fig. 2. Hardware block diagram for the developed VT system.

Table I shows comparison results of required computational amount in clock cycles for the case of ARM925T processor. As shown in the table, the proposed system can reduce the computational amount required for synchronization by more than one-tenths. The simplified method in this table is comprised by (3) and (5). Finally, the result for the proposed method is obtained using (6) and (7). In the case of conventional and simplified methods, we obtain these results after scaling into fixed-point arithmetic.

TABLE I
COMPARISON OF REQUIRED CLOCK CYCLES FOR DIFFERENT APPROACHES

Required Clock Cycles	Conventional Method	Simplified Method	Proposed Method
Decision rule	145 cycles	72 cycles	8 cycles

IV. Conclusions

In this paper, we describe an efficient A/V synchronization algorithm that is quite useful for video telephony applications for cellular phones. The proposed method requires far less computation compared to the conventional algorithm. As shown in (6), the obtained decision rule is very easy to implement. Part of this algorithm is now pending as a Korean patent [1].

References

[1] C. Kim and K.-d. Seo, "An apparatus for synchronizing audio/video for hand-held devices," *Korea Patent Application*, P04-046697.

[2] C. Kim, S. Park and K.-d. Seo, "An efficient audio/video synchronization method for video telephony," KISS Korea Computer Congress, July 2005.

[3] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "Real-time transport protocol," *RFC 3550*, IETF, July 2003.

[4] D. Mills, "Network time protocol specification, implementation and analysis," *RFC 1305*, IETF, Mar. 1992.

[5] S. Furber, *ARM System Architecture*, Harlow, UK: Addison-Wesley, 1996.

[6] *OMAP1510 Multimedia Processor (Technical Reference Manual),* Dallas, TX: Texas Instruments, June 2002.

[7] C. Perkins, RTP: *Audio and Video for the Internet*, Boston, MA: Addison-Wesley Professional, 2003.

[8] M. Handley, V. Jacobson, "Session description protocol," *RFC 2327,* IETF, Apr. 1998.