LETTER *Special Section on Human Communication II*

Efficient Media Synchronization Method for Video Telephony System

Chanwoo KIM[†], ***Nonmember***, **Kwang-Deok SEO**[††a)], ***and*** **Wonyong SUNG**[†††], ***Members***

SUMMARY In this letter, we derive an efficient audio/video synchronization method for video telephony. For synchronization, this method does not require any further RTCP packet processing except for the first one. The derived decision rule is far more compact than the conventional method. This decision rule is incorporated in an actual video telephony system adopting Texas Instruments (TI) OMAP 1510 processor and Qualcomm MSM 5500. The computational requirement was compared with the conventional method and through simulations the superiority of the proposed method is proved.

key words: *media synchronization, video telephony, multimedia communications*

1. Introduction

Recently, the conspicuous trends of cellular phones lie in the fact that many multimedia services become to be incorporated in a single handset. Reflecting this trend, many of the contemporary cell phones are implemented with applications like finger-print recognition, speech recognition, video-telephony (VT), and video-on-demand (VOD). Among them, with the recent advent of Wideband Code-Division Multiple Access (WCDMA) services in Korea and Japan, VT system has begun to draw significant attention.

To transmit data over Internet Protocol (IP) in real-time, Real-Time Protocol (RTP) and RTP Control Protocol (RTCP) are usually employed. RTP carries the payload with some additional information like sequence number and RTP timestamp. RTCP serves for controlling quality of the transmitted data. RTP timestamp begins at a random number and its rate of increment is proportional to its sampling rate [3]. Thus, they do not directly give information on absolute time reference. To synchronize audio and video data, we need to utilize RTCP Sender Report (SR) packet to find out the absolute time information corresponding to each RTP timestamp carried by each RTP packet [3].

In this letter, we propose an efficient audio/video (A/V) synchronization method. In this method, we do not need to process every RTCP SR packet for synchronization. Moreover, it does not require any floating-point operations or any divisions at all. Obviously, this is a clear advantage for embedded processors used for VT handsets. Through extensive simulations, the proposed method shows noticeable advantages compared to previous ones like Bertoglio's method [1].

The organization of this paper is as follows: In Sect. 2, we explain the conventional audio/video synchronization algorithm. In Sect. 3, we describe a comprehensive explanation on the proposed method for audio/video synchronization in detail. Section 4 describes the entire structure of the implemented VT system incorporating the proposed synchronization method and shows the simulation results. Finally, we draw conclusions in Sect. 5.

2. Conventional Audio/Video Synchronization Algorithm

To transmit multimedia data in real-time over the Internet Protocol (IP), we usually adopt RTP and RTCP as previously mentioned. Figure 1 (a) shows the header of an RTP packet. To facilitate the real-time transmission, RTP header carries sequence number and RTP timestamp as shown in this figure.

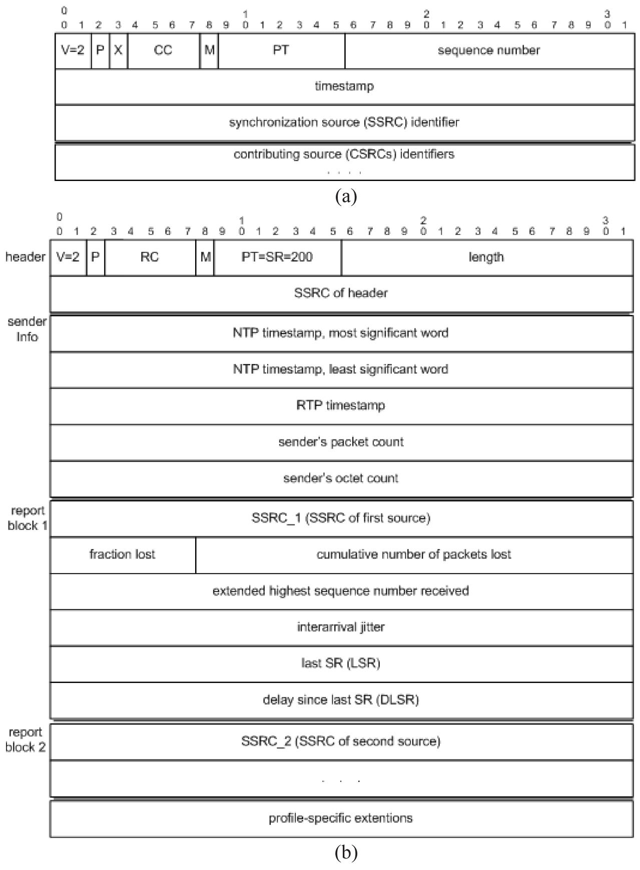
Lip synchronization is a crucial human perception issue for VT and VOD systems. Previous research shows the following experimental results on lip synchronization [1], [7]. In most cases, people do not detect the synchronization error if the time difference between audio and video is less than 80 ms. However, if the time difference is larger than 160 ms, every observer detects this error and feels uncomfortable with the video service. If the error is larger than 80 ms but smaller than 160 ms, the detection of the error depends on the communication environment. One interesting result is that people feel more comfortable with the "video ahead of audio" case than the other one.

According to RFC 3550, it is stated that separate audio and video streams should not be carried in a single RTP session and demultiplexed based on the payload type or SSRC fields [3]. However, we cannot directly use RTP timestamp to synchronize data carried by different RTP sessions for the following two reasons. Firstly, the initial value of the RTP timestamp should be random as specified in RFC 3550 [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

Manuscript received September 28, 2005.
Manuscript revised January 7, 2006.

[†]The author is with the School of Computer Science, Carnegie Mellon Univ., Pittsburgh, PA, 15213–3891, USA.
[††]The author is with the Computer and Telecommunications Engineering Division, Yonsei Univ., Wonju, Gangwon, 220–710, Korea.
[†††]The author is with the School of Electrical Engineering and Computer Science, Seoul National Univ., Seoul, 151–744, Korea.
a) E-mail: kdseo@yonsei.ac.kr

DOI: 10.1093/ietisy/e89–d.6.1901



V=2
P
X
CC
M
PT
sequence number
timestamp
synchronization source (SSRC) identifier
contributing source (CSRCs) identifiers
(a)
header
V=2
P
RC
M
PT=SR=200
length
SSRC of header
sender Info
NTP timestamp, most significant word
NTP timestamp, least significant word
RTP timestamp
sender's packet count
sender's octet count
report block 1
SSRC_1 (SSRC of first source)
fraction lost
cumulative number of packets lost
extended highest sequence number received
interarrival jitter
last SR (LSR)
delay since last SR (DLSR)
report block 2
SSRC_2 (SSRC of second source)
profile-specific extentions
(b)

Fig. 1 (a) Structure of RTP header and (b) Structure of RTCP SR header and reception report block.

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]. Figure 1 (b) illustrates the structure of RTCP SR packet. By inspecting the relation between RTP and NTP timestamps in RTCP SR packet, we can find out the reference time corresponding to the RTP timestamp.

Figure 2 shows RTP and RTCP streams for audio and video sessions. As previously mentioned, audio and video payloads are transmitted by different RTP streams. And separate RTCP streams are used for audio and video sessions. In what follows, we will use the superscript A and V to denote audio and video sessions, respectively.

For the derivation of the relationship between the RTP timestamp of a specific RTP packet and the absolute time reference, let us consider the shaded RTP packet in the audio session shown in this figure. For this RTP packet, $T^A_{I_A}(j_A)$ is the absolute time reference for RTP timestamp of the j_A-th RTP packet after the I_A-th RTCP packet has been received. Network Time Protocol (NTP) tells us how to set the absolute time information [4]. As a special case, when $j_A = 0$, $T^A_{I_A}(0)$ is the NTP timestamp contained in the I_A-th RTCP packet. Similarly, $M^A_{I_A}(j_A)$ is the RTP timestamp contained in the j_A-th RTP packet after the I_A-th RTCP packet and $M^A_{I_A}(0)$ is the RTP timestamp for the I_A-th RTCP packet. Note that the absolute time reference, $T^A_{I_A}(j_A)$ is not contained in the RTP header, but we should obtain from the RTP timestamp, $M^A_{I_A}(j_A)$, which is contained in the header.

Let us assume that the shaded RTP packet in the video session is sampled at the same time with the shaded RTP packet in the audio session. If the absolute time reference of this video RTP packet is represented by $T^V_{I_V}(j_V)$, it is required that $T^A_{I_A}(j_A) = T^V_{I_V}(j_V)$ for perfect synchronization. Here, we need to note that the transmission rates of RTP packets are normally not the same for different sessions. Additionally, RTCP packets for each session may be transmitted at different time. Thus, even if $T^A_{I_A}(j_A) = T^V_{I_V}(j_V)$, I_A and j_A of the audio session may not be equal to I_V and j_V of the video session, respectively.

Now we will touch briefly on the previously reported synchronization method based on the abovementioned technique using the RTP timestamp and the RTCP SR packet. Using this technique, we can compute $T^A_{I_A}(j_A)$ of a RTP timestamp using $M^A_{I_A}(j_A)$ carried by this RTP packet. $M^A_{I_A}(0)$ and $T^A_{I_A}(0)$ values are also used in the computation, which can be obtained by the I_A-th RTCP packet.

In the method proposed by Bertoglio [1], the absolute time reference $T^A_{I_A}(j_A)$ is obtained by

$$T^A_{I_A}(j_A) = T^A_{I_A}(1) + \sum_{k=2}^{j_A} \frac{\Delta M^A_{I_A}(k)}{R^A}, \tag{1}$$

where R^A is the sampling rate of audio data.
$T^A_{I_A}(1)$ is obtained by

$$T^A_{I_A}(1) = T^A_{I_A}(0) + \frac{M^A_{I_A}(1) - M^A_{I_A}(0)}{R^A}. \tag{2}$$

In (1), $\Delta M^A_{I_A}(k)$ is the difference between the RTP timestamps of two adjacent RTP packets and is given by

$$\Delta M^A_{I_A}(k) = M^A_{I_A}(k) - M^A_{I_A}(k-1). \tag{3}$$

Computation of (1) and (3) continues until a new control packet (RTCP) is received. After receiving the $(I_A + 1)$-th RTCP packet, (1) and (2) are computed again using the value $T^A_{I_A+1}(0)$ and $M^A_{I_A+1}(0)$ carried by this RTCP packet. When computing $T^A_{I_A}(j_A)$ by (1) in Bertoglio's method for the j_A-th RTP packet, they do not compute the term $\sum_{k=2}^{j_A} \frac{\Delta M^A_{I_A}(k)}{R^A}$ directly. Instead, since they already know the value of $T^A_{I_A}(j_A - 1)$, they compute the value by

$$T^A_{I_A}(j_A) = T^A_{I_A}(j_A - 1) + \frac{\Delta M^A_{I_A}(j_A)}{R^A}. \tag{4}$$

The same procedure (1)–(4) can be applied to the video session to obtain $T^V_{I_V}(j_V)$.

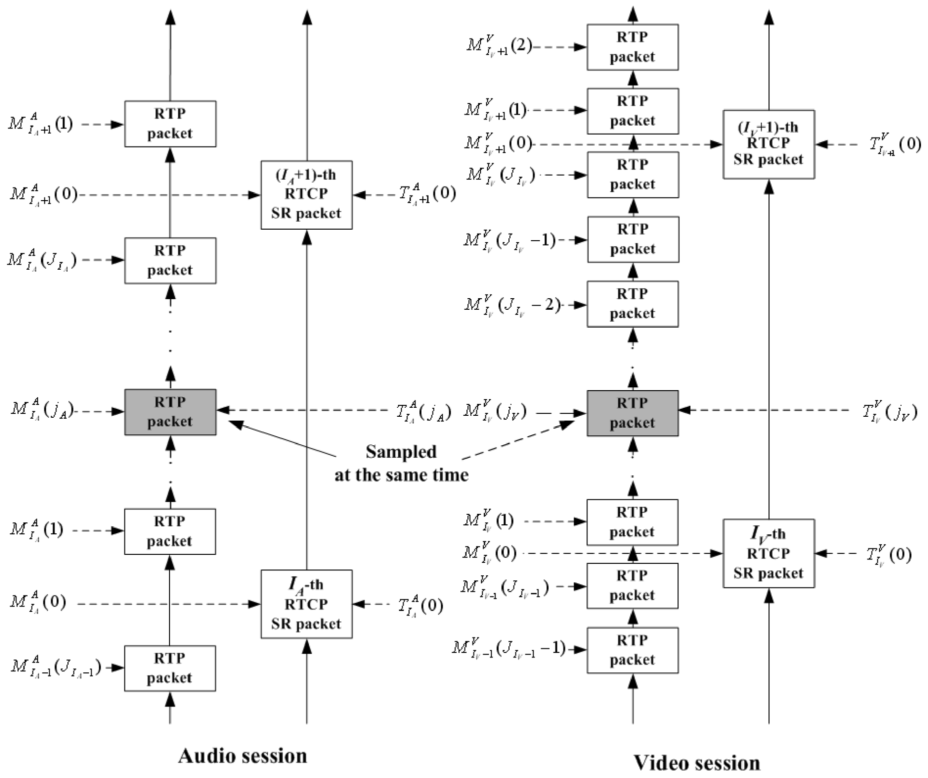


Fig. 2 RTP and RTCP streams for audio and video sessions.

When processing j_A-th RTP packet for the audio session and j_V-th RTP packet for the video session for synchronization, Bertoglio's decision rule based on $T^A_{I_A}(j_A)$ and $T^V_{I_V}(j_V)$ for synchronization is as follows:

$$\begin{array}{ll} T^V_{I_V}(j_V) - T^A_{I_A}(j_A) > \eta_+ & \text{: Video is ahead of audio,} \\ \eta_+ \geq T^V_{I_V}(j_V) - T^A_{I_A}(j_A) \geq -\eta_- & \text{: Audio and video are in synchronization,} \\ T^V_{I_V}(j_V) - T^A_{I_A}(j_A) < -\eta_- & \text{: Audio is ahead of video,} \end{array} \tag{5}$$

where η_+ and η_- are thresholds used for boundaries of the in-sync region [1]. To apply this decision rule, it is evident that we need to inspect every RTCP packet for the computation of (1) through (5).

3. Proposed Synchronization Algorithm

In this section, we will derive the proposed scheme from the Bertoglio's method described by (1)–(5). In this study, we exploit the fact that after a call has been setup, the codec type and the sampling rate are sustained during this call [2].

By canceling out each term in the computation of $\sum_{k=2}^{j_A} \frac{\Delta M^A_{I_A}(k)}{R^A}$ in (1) and by using (2), we can simplify (1) into the following form

$$T^A_{I_A}(j_A) = T^A_{I_A}(1) + \frac{M^A_{I_A}(j_A) - M^A_{I_A}(1)}{R^A}$$

$$= T^A_{I_A}(0) + \frac{M^A_{I_A}(j_A) - M^A_{I_A}(0)}{R^A}. \tag{6}$$

Assuming that R^A is kept as a constant, we can obtain (7) from the NTP and RTP timestamps carried by the 0-th and the I_A-th RTCP packet as follows

$$R^A = \frac{M^A_{I_A}(0) - M^A_0(0)}{T^A_{I_A}(0) - T^A_0(0)}. \tag{7}$$

Rearranging (7) for $T^A_{I_A}(0)$ and substituting it into (6) yields

$$T^A_{I_A}(j_A) = T^A_0(0) + \frac{M^A_{I_A}(j_A) - M^A_0(0)}{R^A}. \tag{8}$$

Similarly, we can apply this procedure to obtain the relation (9) for the RTP stream of video session.

$$T^V_{I_V}(j_V) = T^V_0(0) + \frac{M^V_{I_V}(j_V) - M^V_0(0)}{R^V}. \tag{9}$$

By subtracting (9) from (8) and applying this result to (5), we can derive the following compact decision rule (10) after some arithmetic,

$$\begin{array}{ll} d > \eta_0 + R^A R^V \eta_+ & \text{: Video is ahead of audio,} \\ \eta_0 + R^A R^V \eta_+ \geq d \geq \eta_0 - R^A R^V \eta_- & \text{: Audio and video are in synchronization,} \\ d < \eta_0 - R^A R^V \eta_- & \text{: Audio is ahead of video,} \end{array} \tag{10}$$

where the variable d and the threshold constant η_0 are defined by

$$d = R^A M^V_{I_V}(j_V) - R^V M^A_{I_A}(j_A), \tag{11}$$

$$\eta_0 = R^A R^V (T^V_0(0) - T^A_0(0)) + R^V M^A_0(0) - R^A M^V_0(0). \tag{12}$$

In (10), η_+ and η_- are the same thresholds as used in (5).

Note that we only need to compute d by (11) when examining the synchronization of each pair of RTP packet by (10). The reason is that $\eta_0 + R^A R^V \eta_+$ and $\eta_0 - R^A R^V \eta_-$ in (10) are needed to be computed just once after receiving the first RTCP packet. Since all of the R^V, R^A, $M^V_{I_V}(j_V)$, and $M^A_{I_A}(j_A)$ values in (11) are fixed-point numbers themselves, there is no need to utilize floating-point operations at all. Obviously, this is a clear advantage for embedded processors, which usually do not have floating point units. Additionally, (11) does not require any division operations like the case of (1)–(4). For the ARM (Advanced RISC Machine) processors, avoiding division is also an advantageous aspect, since they do not have any hardware divider [5]. In [1], they reported that truncation round-off errors may accumulate, since the Bertoglio's method in (1)–(5) is computed by repeated divisions and summations. They argue that repeated processing of RTCP SR packet is mandatory to prevent the accumulation of round-off errors from becoming too large. However, it is obvious that there are no possibilities of error accumulation in the computation of (11). Thus, the proposed method is also advantageous in this respect. Note that only two fixed-point multiplications and one subtraction are needed for the computation of d in (11).

Now, we will describe an efficient RTP time-stamping for the proposed synchronization. The RTP timestamp reflects the sampling instant of the first octet in the RTP data packet. In some conventional real-time communication systems, the calculation of a specific RTP timestamp can be done based on its previous RTP timestamp value. In this case, there may be possible accumulation errors after repeated calculation. However, in our system, we do not compute the RTP timestamp for a specific packet based on the previous RTP timestamp value. So, the accumulation error is not present at all. As specified in [3], the initial value of the RTP timestamp should be random, as for the sequence number. Let us assume that this initial random value for a specific RTP session is RTP_0 corresponding to an internal clock t_0 of the system. If we assume that the first octet of any "arbitrary" RTP packet is sampled at t, we can easily compute the corresponding RTP timestamp RTP_t by the following equation:

$$RTP_t = RTP_0 + R_{media} \cdot (t - t_0), \tag{13}$$

where R_{media} is the sampling rate of the media. We can use this equation, since for VT system in Korea, the codec type does not change during a single call and subsequently there is no change in the media sampling rate. Thus, we can compute the RTP timestamp without any accumulation of previous RTP timestamp values.

On the other hand, the NTP timestamp in the system can be directly obtained using the internal clock of the system and the internal clock can be inaccurate. However, its accuracy is not important at all for the proposed synchronization process. In the proposed method, we are only interested in the timing precedence between the media as shown in (5). If there is an error in the original value of t_0, it is applied to both of the audio and video data. Thus, the timing precedence between audio and video does not change at all. In sum, we would like to insist that our system is free of any possible accumulation of error and quite suitable for A/V synchronization.

4. System Implementation and Experimental Results

Figure 3 shows the hardware block diagram of the developed system, which is a prototype VT system. The proposed method described in Sect. 3 is tested on top of this system, which is based on Mobile Station Modem (MSM) 5500 CDMA modem and TI OMAP 1510 multimedia processor [6]. Figure 4 illustrates the software structure of this VT system. We adopt H.263 video codec for video processing, and Qualcomm Code Excited Linear Prediction (QCELP) for audio processing. Video codec operates on the TMS320C5510 DSP incorporated in the OMAP 1510 processor, while audio is processed at the Qualcomm Digital Signal Processor (QDSP) within MSM 5500. This system is basically based on CDAM 1x EVDO (Evolution Data Only) technique. A commercial RTOS (Real-time Operating System) Nucleus is ported on the ARM 925T proces-

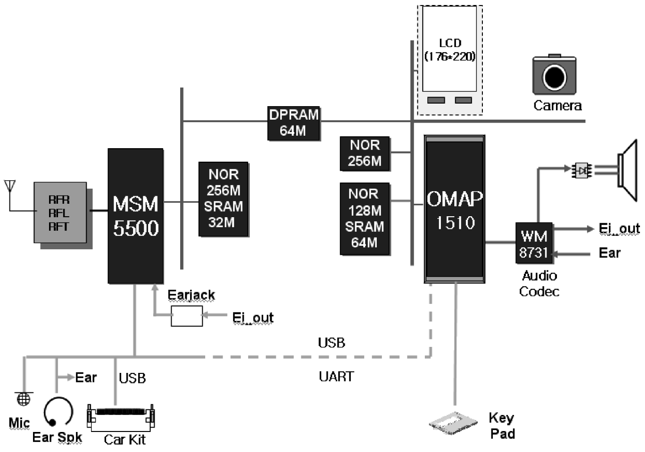


Fig. 3 Hardware block diagram of the developed VT system.

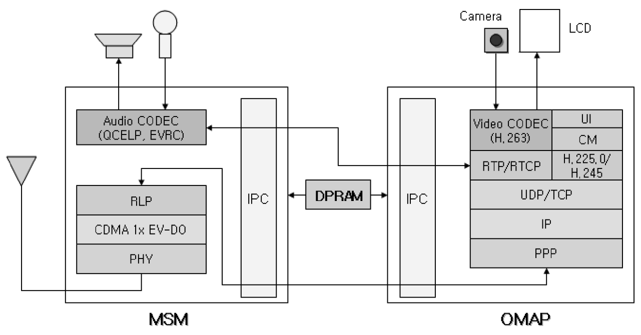


Fig. 4 Software structure of the developed VT system.

Table 1 Comparison of required clock cycles for processing a pair of RTP packets using the different decision rules.

Method	Bertoglio's Method	Proposed Method
Required Clock Cycles	73 cycles	9 cycles

sor of the OMAP 1510. While H.263 codec operates on the TMS320C5510 DSP side of the OMAP processor, protocols like RTP and RTCP are operated in the ARM side.

When audio and video data are received, they are first stored in the input side of queues, which are based on the linked list data structure. In the mean time, audio and video decoder fetches the stored data in the other side of queue. In the implemented system, RTP and RTCP are processed in a single Nucleus task. The proposed decision rule is employed whether the audio and video data being decoded are in synchronization. If a synchronization error is found using (10), then we temporarily blocks the operation of the decoder which goes too faster until the synchronization condition in (10) is met. After repeated experiments, we found that it is reasonable to choose the thresholds of $\eta_+ = \eta_- = 50\,\text{ms}$ in (10). In choosing these threshold values, we considered the fact that the difference in the net processing delay between the audio and video data is very small in the current VT system through extensive simulations. The time for processing H.263 video data is usually larger than that for processing QCELP data. However, in the current implementation, the QCELP is located on the MSM chip and there is an additional delay for transmitting the decoded audio data to the OMAP chip through the DPRAM (Dual Port RAM) interconnection between MSM and OMAP. Therefore, the net delays for processing the audio and video data are very similar in the current VT system. Using those threshold values, we found that the audio and video data are fully in-sync in various simulations. Note that the suitable values of η_+ and η_- for other multimedia communication systems may be slightly different from the current values, since they depend on the hardware configuration and/or the codec types.

Table 1 compares the required clock cycles for each decision rule when applied to a pair of audio and video RTP packets in the ARM 925T processor. As shown in this table, the proposed method requires far less computation compared to the Bertoglio's method. From the previous discussion, we can easily find that in the case of the proposed method, we only need to perform one subtraction, two multiplications, and two comparisons. However, we need to perform three subtractions, two divisions, two additions, and two comparison operations in the case of Bertoglio's method. In this table, we averaged the number of clock cycles for the case of Bertoglio's method. The reason is that the number of clock-cycles required for a division operation is varied, since it is implemented as a numerical software routine in the case of ARM processors. As previously mentioned, our proposed method is also advantageous in that it does not require further processing of the RTCP SR packets except the first one.

5. Conclusions

In this letter, we propose an efficient audio/video synchronization method. We can summarize the advantageous aspects of the proposed method in the following three ways. First, the decision rule is far simpler than the previous Bertoglio's method, so it requires far less computation. Second, it does not require RTCP SR packet processing for synchronization except first RTCP packet. Thus, the software structure becomes much simpler and the computational requirement can be reduced even further. Finally, the proposed method does not suffer from the accumulation of round-off errors that are inherent in the previous ones like Bertoglio's method. Simulations with a real video-telephony system proved those improvements. We also expect that the developed algorithm can be efficiently used for synchronization in applications like VOD systems or multi-party multimedia conferencing systems.

References

[1] L. Bertoglio, R. Leonardi, and P. Migliorati, "Intermedia synchronization for video conference over IP," Signal Process., Image Commun., vol.15, no.1, pp.149–164, 1999.
[2] C. Kim and K.-D. Seo, "Method for synchronizing video/audio data of mobile communication terminal," Japan Patent Application, 2005-162052.
[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 (version 3) specification, implementation and analysis," RFC 1305, IETF, March 1992.
[5] S. Furber, ARM System Architecture, Addison-Wesley, Harlow, UK, 1996.
[6] OMAP1510 Multimedia Processor (Technical Reference Manual), Dallas, Texas Instruments, June 2002.
[7] R. Steinmetz, "Human perception of jitter and media synchronization," IEEE J. Sel. Areas Commun., vol.14, no.1, pp.61–72, Jan. 1996.