Perceptual Codec and Interaction Aware Playout Algorithms and Quality Measurement for VoIP Systems

Kuo-Kun Tseng, Yuan-Cheng Lai and Ying-Dar Lin

Abstract —*To reduce the effect of network jitter, the playout algorithm for voice streams should correctly adjust the playout delay. Conventional playout algorithms were based on network delay only; they did not consider the perceptual quality, and were not aware the codec and interactive mode. Therefore, we present two novel approaches: codec aware adaptive playout (CAAP) algorithm and interactive aware adaptive playout (IAAP) algorithm, which intent to optimize the voice quality based on codec and interactive mode respectively. The performance of CAAP and IAAP are superior to the prior algorithms in our substantial evaluation. Since no objective mechanisms for measuring the speech quality of two-way communication exist, we also propose a new quality measurement for it.*

Index Terms —VoIP, Network Jitter, Real Time Streams, Speech Quality Measurement, Perceptual Quality.

I. INTRODUCTION

The popularity of the Internet and the rapid development of network technologies have made real-time multimedia networked applications possible. These applications, such as Online Conferencing and Internet Phone, have become more and more popular. These devices over the Internet provide low cost communication as compared with public telephones. Thus voice over IP (VoIP) is an attractive alternative for the long distance communication.

One of the major obstacles in VoIP communication is the packets transmitting over the Internet will encounter different delays and thus causes the unpredictable and uncontrollable jitter. Without properly processing, the jitter could cause the packet loss and the degradation of voice qualities. The jitter is typically alleviated by using a delaying mechanism, which queues the irregular arrival packets in the playout buffer and then plays the buffer packet after a proper playout delay. How to determine an appropriate playout delay is very important and also is a challenge. The playout delay can not be neither too long nor too short since the long delay will cause unnecessary waiting and short delay will cause the unnecessary packet loss, herein that means the packet will be dropped when the jitter is longer than playout delay.

The conventional adaptive playout algorithms were jitter-oriented, that is, they estimated the playout delay from the jitter only. They intended to minimize the packet loss by setting the playout delay according to the jitter, but the resulting speech quality may not be good because of the long delay.

A fundamental and representative algorithm for estimating playout delay is the mean delay and variance (MDV) algorithm described in [1] [2]. MDV estimates playout delay from the difference of previous network delays in conjunction with a smoothing factor, which is adjustable according to the network conditions. Another method is described in the real-time transport protocol (RTP) standard [3]. The RTP method is essentially the MDV method applied with a fixed smoothing factor. Other prior algorithms for estimating playout delay include a spike detection algorithm (SDA) [1], and a related gap-based algorithm (GBA) [4]. Both SDA and GBA offer little significant improvement over the MDV method at the expense of added complexity. We also surveyed other related playout algorithms [5] [6] [7] [8] [9] [10], but no similar idea to our proposed algorithms was found.

In the paper, in additional to network jitter, codec type and interactive mode are used to determine the playout delay. Because each codec uses its specific compression algorithm, thus it has its own compression ratio and loss tolerance. For example, G.711 can tolerate more packet losses than G.721 without affecting speech quality a lot. Thus it will produce excellent speech quality that a playout delay is decided by extra considering codec type. For the interactive mode, it has interactive and non-interactive communication, the interactive communication means that two people are talking to each other, and the non-interactive communication means that only one person is talking. The emergencies of real-time are different between the interactive mode, such as interactive talking, and the non-interactive mode, such as audio broadcasting. The latter usually allows a longer playout delay than the former.

Thus, we propose the codec aware adaptive playout (CAAP) algorithm and interactive aware adaptive playout (IAAP) algorithm, which intent to optimize the voice quality based on codec and interactive mode respectively. Our playout algorithms determine the playout delay according to the feedback of perceptual quality. The perceptual quality is computed by our proposed a new quality measurement - $LMOS - DMOS$ measurement, which is able to map the packet delay and loss to the perceptual quality of speech.

This paper is organized as follows. In section 2, we survey some related works of playout algorithms. In section 3, we present the $LMOS - DMOS$ measurement, the architectures, and algorithms of CAAP and IAAP. In section 4, we describe the emulation architecture and setting configuration to evaluate

K. K. Tseng is with Department of Computer and Information Science at National Chiao Tung University, Taiwan (e-mail: kktseng@cis.nctu.edu.tw).

Y. C. Lai is with the Department of Information Management at National Taiwan University of Science and Technology, Taiwan (e-mail: laiyc@cs.ntust.edu.tw).

Y. D. Lin is with the Department of Computer and Information Science at National Chiao Tung University, Taiwan (e-mail: ydlin@cis.nctu.edu.tw).

Contributed Paper
Manuscript received November 21, 2003

CAAP and IAAP. Section 5 exhibits the simulation results, emulation results, and their analyses. Finally in section 6 we state our conclusions.

II. Related Works

Fig. 1 is a conventional playout mechanism for the Internet phone. For receiving path, the architecture of Internet phone is composed of data processing units: receiver, playout buffer and media output, and control processing units: jitter estimator, playout controller and parameters. Incoming voice packets are received from the receiver, and then queued in the playout buffer, which is used to absorb the jitter and is controlled by the playout controller. When the playout delay is expired, the playout buffer sends the packets to the media output. Conventional algorithm resides in playout controller and uses jitter estimator and related parameters to estimate playout delay. For transmitting path, the voice packets from media input device are directly transmitted to network.

A fundamental and representative algorithm for estimating playout delay is the mean delay and variance (MDV) algorithm. To determine the jitter, MDV calculates average delay firstly. The average delay $\widetilde{d}_i$ is the average round-trip time (RTT) of n previous packets from the i^{th} packet, and is ideally calculated as

$$\widetilde{d}_i = \sum_{k=i}^{i-n+1} \frac{RTT_k}{n}. \tag{1}$$

Also MDV simplifies the calculation of $\widetilde{d}_i$ to a moving average equation as $\widetilde{d}_i = \alpha \bullet \widetilde{d}_{i-1} + (1-\alpha) \bullet d_i$, where $\widetilde{d}_i$ is a weighted sum over network estimated delay d_i with a smoothing factor α. The estimated delay d_i is as $d_i = |(r_i - r_{i-1}) - (s_i - s_{i-1})|$, where s_i and r_i are sender timestamp and receiver timestamp of the i^{th} packet, respectively. MDV also uses the moving average to determine the network jitter $\widetilde{v}_i = \beta \bullet \widetilde{v}_{i-1} + (1-\beta) \bullet v_i$, where the variance of network delay is $v_i = \left|\widetilde{d}_i - d_i\right|$, and β is a smoothing factor. In summary, the MDV algorithm is calculated as

$$\begin{aligned} d_i &= |(r_i - r_{i-1}) - (s_i - s_{i-1})|, \\ \widetilde{d}_i &= \alpha \bullet \widetilde{d}_{i-1} + (1-\alpha) \bullet d_i, \\ v_i &= \left|\widetilde{d}_i - d_i\right|, \\ \widetilde{v}_i &= \beta \bullet \widetilde{v}_{i-1} + (1-\beta) \bullet v_i. \end{aligned} \tag{2}$$

In MDV algorithm, after $\widetilde{v}_i$ is determined, the playout delay of the i^{th} packet, p_i^{MDV} is directly set to $\widetilde{v}_i$, that is $p_i^{MDV} = \widetilde{v}_i$.

Another method, RFC, is essentially similar to the MDV method with a fixed smoothing factor. The playout delay of the i^{th} packet, p_i^{RFC} is a moving average over the difference of $\widetilde{d}_i - p_{i-1}^{RFC}$ with a fixed weighted factor $\frac{1}{16}$, as

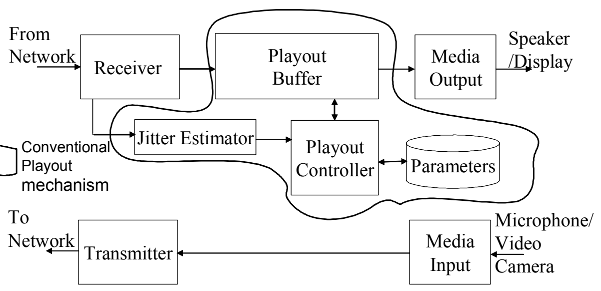


Fig. 1. Architecture of Internet phone

$$p_i^{RFC} = p_{i-1}^{RFC} + \frac{(\widetilde{d}_i - p_{i-1}^{RFC})}{16}. \tag{3}$$

While it is simpler than the MDV method, the RFC method offers a less accurate estimate of playout delay.

The MDV and RFC algorithms are used to competitive playout algorithms in our evaluation. The SDA and GBA algorithms are not considered because they are outperformed by MDV and RFC in our preliminary trial.

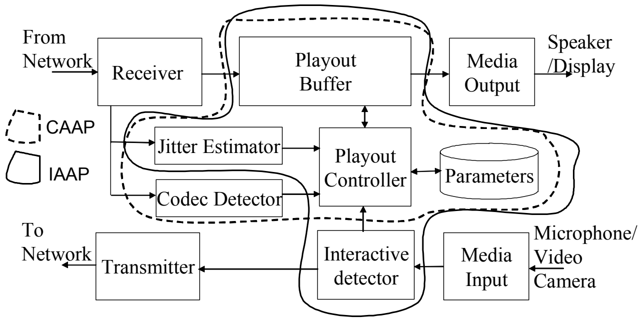


Fig. 2. Applying our algorithm to an Internet phone, CAAP is within the dash line area. IAAP is within the solid line area.

III. Architecture and Algorithm

This section first describes the architecture of our algorithm that applying to an Internet phone, and then describes our CAAP and IAAP algorithms.

A. Architecture

Fig. 2 is a practical embodiment for our algorithm. CAAP is within the dash line area and additionally consists of codec detector. Within the solid line area is the IAAP architecture. Unlike the CAAP, the IAAP processes both the receiving path and the transmitting path. The interactive detector sends a interactive mode indication to the playout controller, which calculates a playout delay according to the jitter estimator and the indication of interactive mode.

As shows in fig. 3, in addition to network jitter, the playout algorithms in the playout controller uses the codec type, communication interactive mode, overflow loss and previous playout delay to determine new playout delay. To determine a new playout delay in the playout controller; there are three main estimators, that are jitter, CAAP and IAAP estimators. CAAP estimator has two sub-stages – simple CAAP (SCAAP) and extended CAAP (ECAAP). The dissimilarity from the conventional algorithms is that our algorithms consider more valuable parameters, and try to optimize voice quality by the perceptual algorithm, instead of reducing packet loss only.

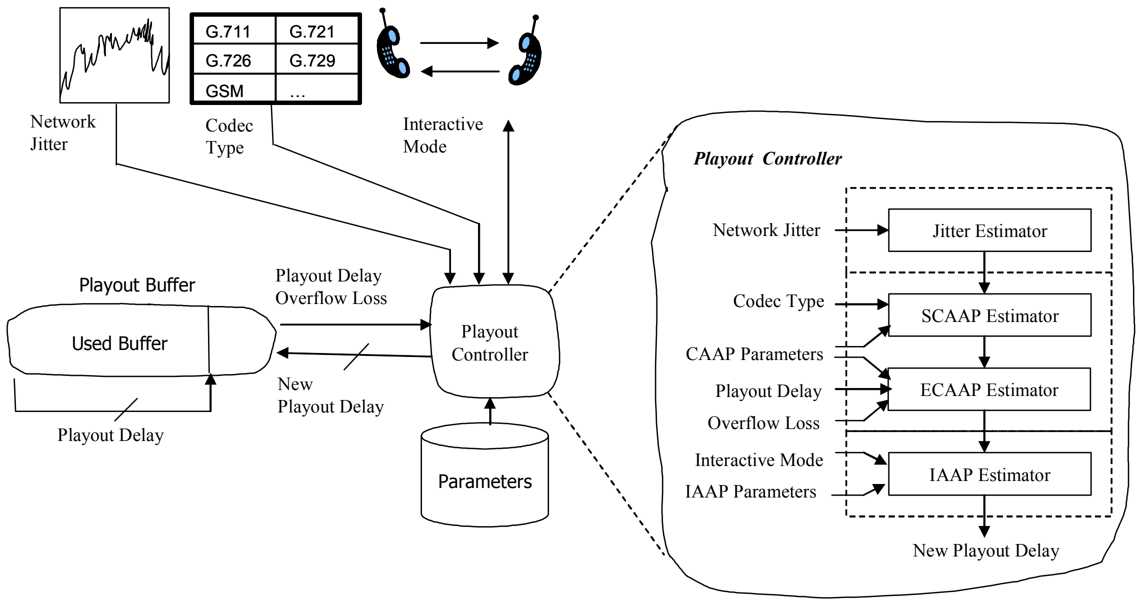


Fig. 3. Codec and Interaction aware adaptive playout algorithms, and playout controller controls the playout buffer and consist of three computing stages.

B. LMOS and DMOS Measurements

Loss mean opinion score ($LMOS$) and delay mean opinion score ($DMOS$) measurements are used in our playout algorithm to provide the feedback quality for the playout control. Moreover, $LMOS$-$DMOS$ measurement can evaluate performance of the playout algorithms. The previous methods of speech quality measurement like Emodel[11][12], PSQM [13], NMB [14] and PESQ [15] etc., are not suitable for measuring two-way communication that two parties are transmitting the stream at the interactive mode. Conventional measurements cannot measure the quality degradation of two way communication that causes by delay and packet loss. Therefore, we proposed a $LMOS$-$DMOS$ measurement which uses an objective mean opinion score (MOS) alike mechanism for the playout algorithms. In this measurement, at first, the $LMOS_i$ and $DMOS_i$ mapped onto normalized quality ranges (excellent, good, medium, poor and worst) according to the loss rate and queueing delay measured for the i^{th} packet, respectively. Then mean MOS ($MMOS_i$) is a final quality value which is an weighted average of $LMOS_i$ and $DMOS_i$. We can obtain $LMOS_i$, $DMOS_i$ and $MMOS_i$ by

$$LMOS_i = \begin{cases} LMOS_{excellent} & | \; R_i \geq \gamma_{excellent} \\ LMOS_{good} & | \; \gamma_{excellent} > R_i \geq \gamma_{good} \\ LMOS_{medium} & | \; \gamma_{good} > R_i \geq \gamma_{medium} \\ LMOS_{poor} & | \; \gamma_{medium} > R_i \geq \gamma_{poor} \\ LMOS_{worst} & | \; Otherwise, \end{cases} \tag{4}$$

$$DMOS_i = \begin{cases} DMOS_{excellent} & | \; D_i \geq \lambda_{excellent} \\ DMOS_{good} & | \; \lambda_{excellent} > D_i \geq \lambda_{good} \\ DMOS_{medium} & | \; \lambda_{good} > D_i \geq \lambda_{medium} \\ DMOS_{poor} & | \; \lambda_{medium} > D_i \geq \lambda_{poor} \\ DMOS_{worst} & | \; Otherwise, \end{cases} \tag{5}$$

$$MMOS_i = \phi \bullet LMOS_i + (1-\phi) \bullet DMOS_i . \tag{6}$$

This measurement characterizes loss quality $LMOS_i$ and delay quality $DMOS_i$ using many discrete ranges, where the R_i and D_i are current packet loss rate and packet delay measured for the i^{th} packet, respectively. γ and λ are the predefined packet loss rates and network delays to partition the corresponding quality range. Note that they should be different for each codec. The effects of loss and delay can be estimated correctly if γ and λ are properly chosen. The detailed setting of $LMOS_i$ and $DMOS_i$ parameters can be referred in section 4.

Once the $LMOS_i$ and $DMOS_i$ have been determined for the i^{th} packet, they are averaged with a control factor ϕ according to (6) to obtain the overall quality $MMOS_i$. The control factor ϕ is used to adjust the weight between $LMOS_i$ and $DMOS_i$. The large ϕ value is used for loss sensitive codec and small ϕ is used for delay sensitive codec.

C. CAAP Algorithm

CAAP estimates playout delays for a current packet based on $LMOS$, $DMOS$, and $MMOS$ with reference to the codec. There are two sub-stages of CAAP – simple CAAP (SCAAP) and extended CAAP (ECAAP). SCAAP uses single estimated playout delay, while ECAAP chooses the best playout delay among multiple estimated playout delays. Besides, depend on the requirements of system complication and performance, the SCAAP and ECAAP can be used either independently or together.

1) Simple CAAP (SCAAP)

As stated before, the major impact to the voice quality is delay and loss. Fig. 4 explains the concept of SCAAP in two cases. In the case 1, for the loss insensitive codecs, we can decrease the playout delay to improve $MMOS$ because $DMOS$ has more significant effect than $LMOS$. Thus the improving the amount of $DMOS$ is larger than the decreasing the amount of $LMOS$, then we can improve the overall voice quality, i.e., a larger $MMOS$. In the case 2, vice versa, we can increase the playout delay for the loss sensitive codecs to obtain the better voice quality. Each codec has its specific tolerance to packet loss, thus the playout delay can be adjusted by varying the control factor.

To correctly estimate the playout delay of the i^{th} packet, SCAAP requires calculating its packet delay and packet loss rate first. The prior playout algorithms only use the network delay as the packet delay, but SCAAP selects three major delay elements to accumulate end-to-end delay, as

$$D_{i-1} = d^c_{i-1} + d_{i-1} + d^p_{i-1}. \tag{7}$$

The codec delay d^c_{i-1} represents time required for the codec to decompress the (i-1)th packet, d_{i-1} represents the current measured network delay of the (i-1)th packet, as calculated in (2), and the estimated playout delay of the (i-1)th packet, d^p_{i-1}, which is equal to p^{SCAAP}_{i-1}, as calculated in (11).

The playout controller also measures the packet loss rate R_{i-1}. Firstly we need to determine packet loss count L_{i-1} in

$$L_{i-1} = \begin{cases} 1 \ | \ p^{SCAAP}_{i-1} \le v_{i-1} \\ 0 \ | \ p^{SCAAP}_{i-1} > v_{i-1}\,. \end{cases} \tag{8}$$

A packet can be discarded if its jitter in network is longer than playout delay. Becuase, if p^{SCAAP}_{i-1} is less than v_{i-1}, then this packet is out of date to be played and L_{i-1} is counted as the lost one. The v_{i-1} is delay variance determined from the previous MDV algorithm in (2).

(9) is ideally to determine packet loss rate, which average the packet loss count over a period of time. The average loss rate R^n_{i-1}, observed at the (i-1)th packet from the previous n packet can be inefficiently calculated as

$$R_{i-1} = \sum_{k=i-1}^{i-n} \frac{L_k}{n}. \tag{9}$$

In reality, our implementation uses (10) instead of (9) to determine R_{i-1} using a moving average method with a loss smoothing factor ρ as

$$R_{i-1} = \rho \bullet R_{i-1} + (1\text{-}\rho) \bullet L_{i-1}. \tag{10}$$

After the estimated delay D_{i-1} and loss rate R_{i-1} is determined, then we use the LMOS and DMOS functions to mapping them into the value of $LMOS_{i-1}$ and $DMOS_{i-1}$, and then obtain $MMOS_{i-1}$ with a parameter ϕ as (6). Thus, the new playout delay of the i^{th} packet is obtained from that $\tilde{v}_i$ multiplied by a scaling factor δ and a ratio of previous $LMOS_{i-1}$ and $MMOS_{i-1}$, as

$$p^{SCAAP}_i = \tilde{v}_i \bullet \frac{\delta \bullet LMOS_{i-1}}{MMOS_{i-1}}. \tag{11}$$

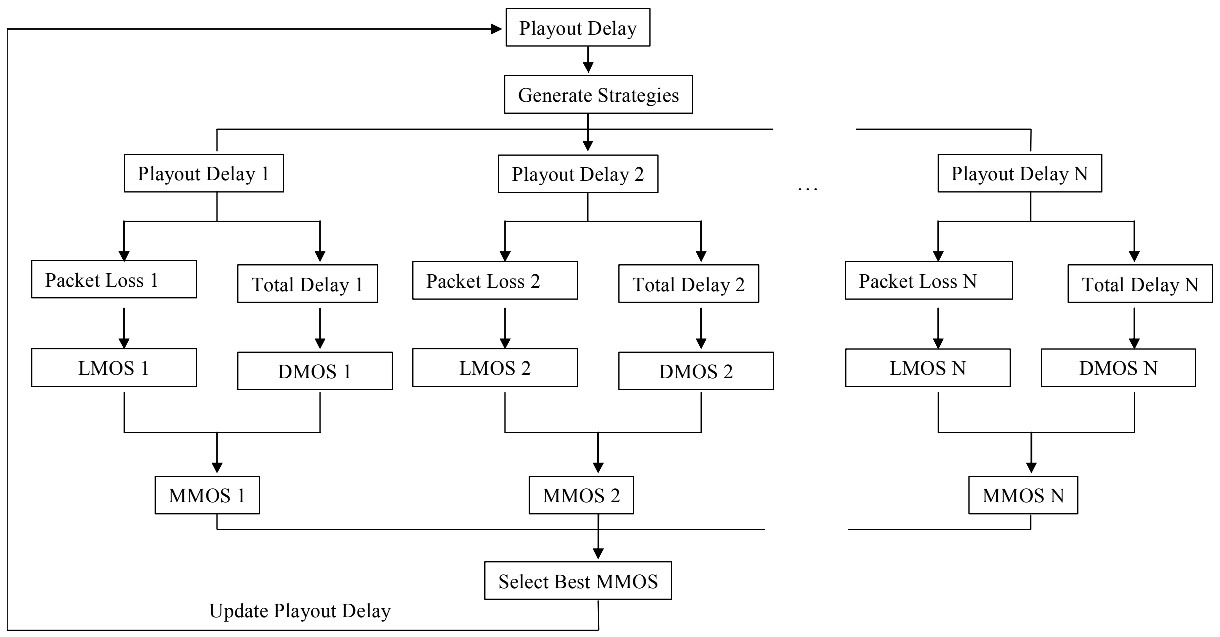


Fig. 5. Extended CAAP.

Prior algorithms directly set the network jitter to playout delay and they did not consider the perceptual quality and the differences between various codecs. However, the SCAAP playout delay p_i^{SCAAP} is adjusted by the previous voice quality, which is determined by packet delay and loss rate.

2) Extended CAAP (ECAAP)

ECAAP employs the multiple estimated playout strategies to produce many estimated playout delays, and then selects the best playout delay among them, as shown in fig. 5.

In the practical implementation, we choose four strategies playout delay p_i^{str} in the ECAAP implementation, where the subscript $\text{str} \in \{\text{PRE}, \text{INC}, \text{DEC}, \text{SCAAP}\}$, PRE is the strategy using the previous playout delay, INC is the strategy increasing the previous playout delay, DEC is the strategy decreasing the previous playout delay, and SCAAP is the strategy using SCAAP algorithm.

The ECAAP algorithm adopts the following four steps:

Step 1. Measure basic network statistics

ECAAP calculates d_i, $\tilde{d}_i$, v_i and $\tilde{v}_i$ in (2) as the MDV algorithm.

Step 2. Decide four estimated playout delays

Four estimated previous playout delays p_i^{PRE}, p_i^{INC}, p_i^{DEC}, and p_i^{SCAAP} are used, as

$$\begin{aligned} p_i^{PRE} &= p_{i-1}^{ECAAP}, \\ p_i^{INC} &= p_{i-1}^{ECAAP} + \Delta, \\ p_i^{DEC} &= p_{i-1}^{ECAAP} - \Delta, \\ p_i^{SCAAP} &= v_i \bullet \frac{LMOS_{i-1}}{MMOS_{i-1}}. \end{aligned} \tag{12}$$

Where Δ is a constant step size.

Step 3. Calculate end-to-end delay D_i^{str} and packet loss rate R_i^{str}

Because the playout delay can dynamically affect the end-to-end delay and packet loss rate, we require recalculating the packet D_i^{str} with (7) and R_i^{str} for each strategy with (8)-(10).

Step 4. Determine new playout delay p_i^{DCAAP}

Given D_i^{str} and R_i^{str} parameters, we can obtain $DMOS_i^{str}$ and $LMOS_i^{str}$ according to the LMOS-DMOS formula in the section III.A, then we can get $MMOS_i^{str}$. Finally, ECAAP chooses the highest $MMOS_i^{str}$ among multiple playout strategies, as

$$p_i^{ECAAP} = p_i^{str} \; with \max MMOS_i^{str} \,|\, \text{str} \in \{\text{PRE}, \text{INC}, \text{DEC}, \text{SCAAP} \tag{13}$$

D. IAAP Algorithm

IAAP first detects the communication mode, which is either interactive or non-interactive, and then calculates the playout delay for a current packet based on the detected communication mode. For a non-interactive mode, the user can allow a long waiting time in listening a speech because user is not aware of communiation delay. Thus, the playout delay in the non-interactive mode can be set longer than that in the interactive mode.

IAAP algorithm follows the below two steps:

Step 1. Calculate network jitter $\tilde{v}_i$.

The IAAP's $\tilde{v}_i$ calculation is the same in (2) as the MDV.

Step 2. Calculate estimated interactive playout delay e_i^{IAAP} and new playout delay p_i^{IAAP}.

The estimated interactive playout delay e_i^{IAAP} is obtained by adjusting $\tilde{v}_i$ with the different scaling factors, $\delta_{interactive}$ and $\delta_{non-interactive}$, for non-interactive and interactive modes, respectively, by

$$e_i^{IAAP} = \begin{cases} \tilde{v}_i \bullet \delta_{interactive} \mid Interactive \\ \tilde{v}_i \bullet \delta_{non-interactive} \mid Non\text{-}Interactive \end{cases}. \tag{14}$$

That $\delta_{non-interactive}$ is larger than $\delta_{interactive}$ means that the longer playout delay is more allowable in the non-interactive mode.

Because IAAP switches dynamically between interactive and non-interactive modes, the different scaling factors $\delta_{non-interactive}$ and $\delta_{interactive}$ might cause significant fluctuation in playing the media. Thus we require applying a moving average on the IAAP algorithm.

The IAAP's playout delay p_i^{IAAP} is a moving average from e_i^{IAAP} with the different smoothing factors $\alpha_{non-interactive}$ and $\alpha_{interactive}$ for the non-interactive and interactive mode, respectively, by

$$p_i^{IAAP} = \begin{cases} \alpha_{interactive} \bullet p_{i-1}^{IAAP} + (1\text{-}\alpha_{interactive}) \bullet e_i^{IAAP} \mid Interactive \\ \alpha_{non-interactive} \bullet p_{i-1}^{IAAP} + (1\text{-}\alpha_{non-interactive}) \bullet e_i^{IAAP} \mid Non-Interactive \end{cases}. \tag{15}$$

$\alpha_{interactive}$ is less than $\alpha_{non-interactive}$ because the non-interactive mode is less sensitive to the interactive listener.

IV. Evaluation Methodology

In this section, we describe our emulation architecture and related configuration for the performance evaluations.

A. Emulation Architecture

Emulation architecture comprises three main blocks - sender, network node and receiver, as shown in fig. 6. Both sender and receiver are implemented in the Speakfreely

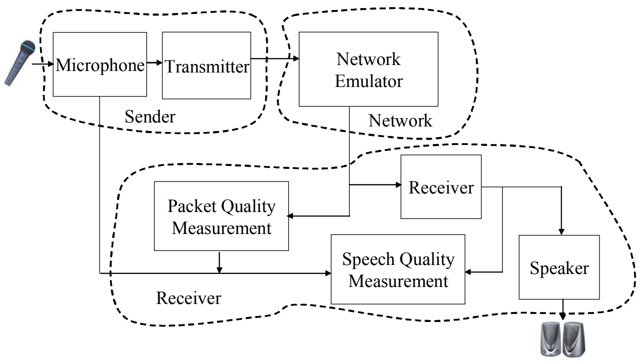


Fig. 6. Emulation architecture that comprises the three dash line areas, the sender, network node and receiver.

TABLE 1. G.711 MOS FOR CAAP

Quality	Loss	Delay	LMOS	DMOS
Excellent	0.05	60.00	4.50	4.50
Good	0.10	120.00	3.50	4.00
Medium	0.15	240.00	3.00	3.50
Poor	0.25	480.00	2.50	2.50
Worst	0.35	720.00	1.00	2.00

TABLE 2. G.721 MOS FOR CAAP

Quality	Loss	Delay	LMOS	DMOS
Excellent	0.05	60.00	4.50	4.50
Good	0.10	120.00	4.00	4.00
Medium	0.15	240.00	3.50	3.50
Poor	0.25	480.00	3.00	2.50
Worst	0.35	720.00	2.50	2.00

TABLE 3. GSM-FR MOS FOR CAAP

Quality	Loss	Delay	LMOS	DMOS
Excellent	0.05	60.00	3.50	3.50
Good	0.10	120.00	3.00	3.20
Medium	0.15	240.00	2.50	2.80
Poor	0.25	480.00	1.80	2.00
Worst	0.35	720.00	1.20	1.50

TABLE 4. G.711 MOS FOR IAAP

Quality	Loss	Delay	ILMOS	IDMOS	NLMOS	NDMOS
Excellent	0.05	60.00	4.50	4.50	4.50	4.50
Good	0.10	120.00	3.00	3.50	3.00	4.00
Medium	0.15	480.00	2.50	2.50	2.50	3.80
Poor	0.25	720.00	2.00	1.00	2.00	3.50
Worst	0.35	1000.00	1.00	0.00	1.00	3.00

software. The sender is a speaker equipped the RTP transmitter function. It generates traffic with an 8K sample rate, 16 bits per sample, and 30 second of speech patterns in each test. The network node is emulated by using a NIST network emulator to generate the Internet traffic. It manipulates the traffic with various jitter ranges, and the delay variations are the white random number among each jitter range. The receiver includes the playout algorithm and the mechanism which gathers network statistics and measures speech quality.

B. Configuration

The simulation and emulation use codecs, G.711, G.721 and GSM-FR, to compare the adaptive playout algorithms - CAAP, IAAP, MDV and RFC. The simulation and emulation are also tested over various jitters ranging among 60, 120, 240, 480 and 720 ms. A specific set of parameters is used for each codec. In each set, LMOS and DMOS are defined with referencing many documents. The speech quality ranges of the loss rate and delay are obtained from ETSI's quality ranges [16] and SLAC's real world measurement [17]. The LMOS and DMOS values are obtained from the ITU G.107 [18], G.113 [19], G.114 [20], ETSI TR41.4-01-02-005 [21] and other related documents [22][23][24][25]. The table 1, 2 and 3 show the parameter setting of G.711, G.721 and GSM-FR used in the simulation and emulation, respectively.

In the IAAP simulation and emulation, the probability of interactive mode is generated by a random number. The parameters setting for IAAP in table 4 is similar to that for CAAP, but it has two LMOS and DMOS columns; namely interactive LMOS (ILMOS), interactive DMOS (IDMOS), non-interactive LMOS (NLMOS) and non-interactive DMOS (NDMOS).

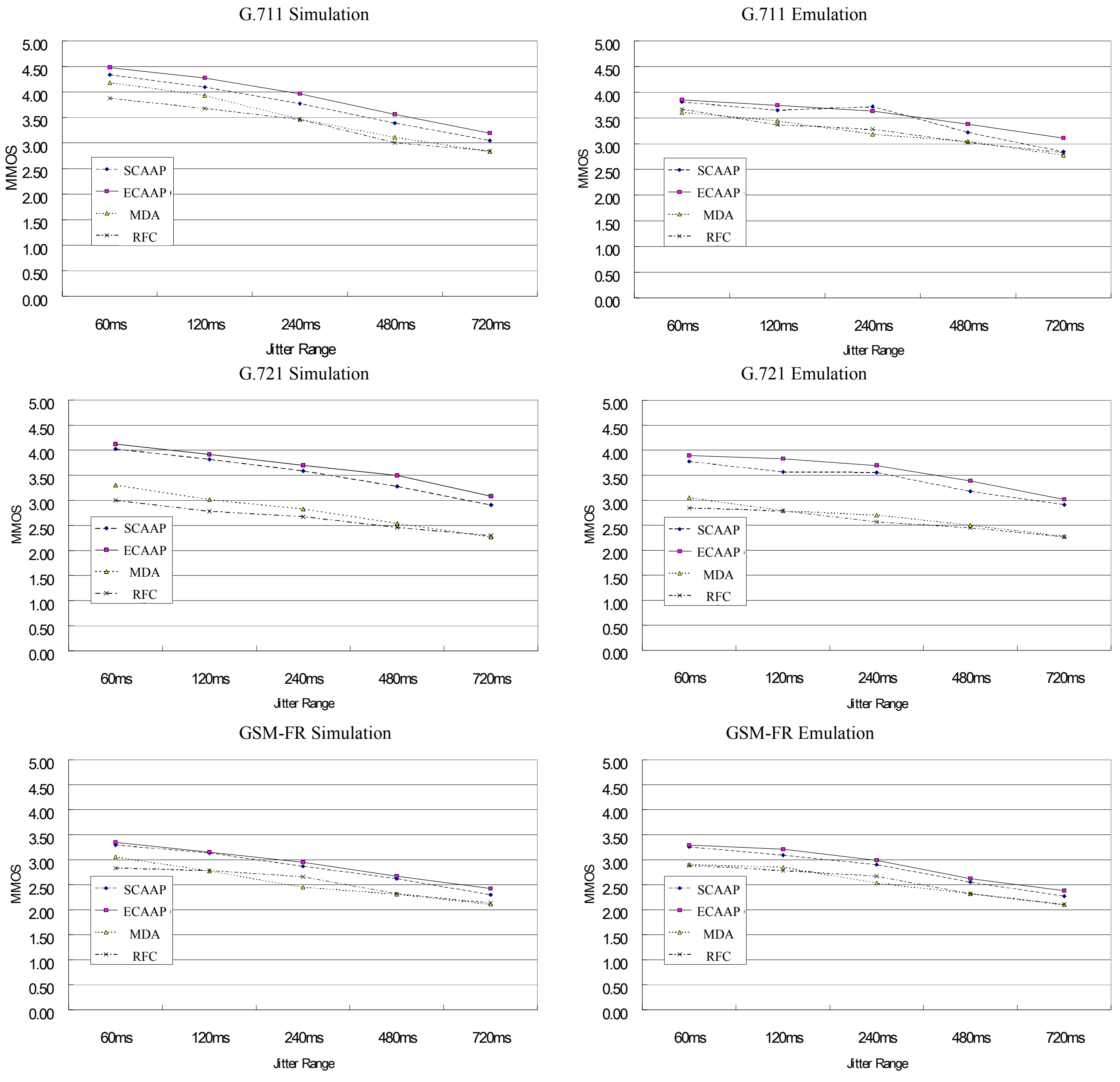
As for the setting related factors in the previous algorithms, In the MDV algorithm, the smoothing factor α for network estimated delay d_i is set to 0.8, the smoothing factor β for the network jitter $\tilde{v}_i$ is set to 0.8. And the d_i and $\tilde{v}_i$ value is the setting among 0.6 to 0.9 range. In the $LMOS$ - $DMOS$ Measurements, depending on the codec type, the control factor ϕ is set in a range from 0.3 to 0.7. In the CAAP algorithm, the loss smoothing factor ρ, scaling factor δ and step size Δ are set to 0.75, 0.8 to 1.2 and 4 ms respectively. In IAAP algorithm, the interactive and non-interactive scaling factors $\delta_{non-interactive}$ and $\delta_{interactive}$ are set to 3 and 1 respectively, and interactive and non-interactive smoothing factors $\alpha_{non-interactive}$ and $\alpha_{interactive}$ are set to 0.7 and 0.9, respectively. The above settings are configured to the best performance for each algorithm in the experiments.

V. RESULTS AND ANALYSIS

For comparison, we conduct some simulations and emulations for CAAP and IAAP. CAAP has evaluated with three codecs, G.711, G.721 and GSM-FR. IAAP is evaluated with two different ratios of interactive to non-interactive mode in G.711 codec only.

A. CAAP Results

In this simulation, the different codec has different baseline qualities with respect to their compression schemes. The baseline qualities of G.711, G.721 and GSM-FR MMOS qualities are around 4.5, 4.0 and 3.5 respectively. Their qualities are all degraded by increasing the jitter amplitude, but the degrade amount are different for the different codecs. Fig. 7 (a) illustrates the speech quality in MMOS value for each algorithm under various jitter ranges. MMOS is decreased as increasing jitter from low range to high range. Obviously, SCAAP and ECAAP have better performance than MDV and RFC in all codec types and all jitter ranges. Thus, the SCAAP and ECAAP improvements seem to be independent of jitter range. Also, the performance metric MMOS is codec dependent. G.711 has a larger decline than GSM-FR when the jitter range is increased, that is, GSM-FR has a stronger jitter resistance than G.711. We also see that ECAAP outperforms the SCAAP, and SCAAP outperforms the MDV and RFC. The statistics for this simulation exhibit that the improvement of ECAAP to MDV is from 9% to 23% and the improvement of SCAAP to MDV is from 5% to 19% on average. Surely the ECAAP and SCAAP have more improvement than RFC, because the MDV is superior to RFC in the normal case. For the further observation, using the lower bit rate codec under



(a) Simulation (b) Emulation

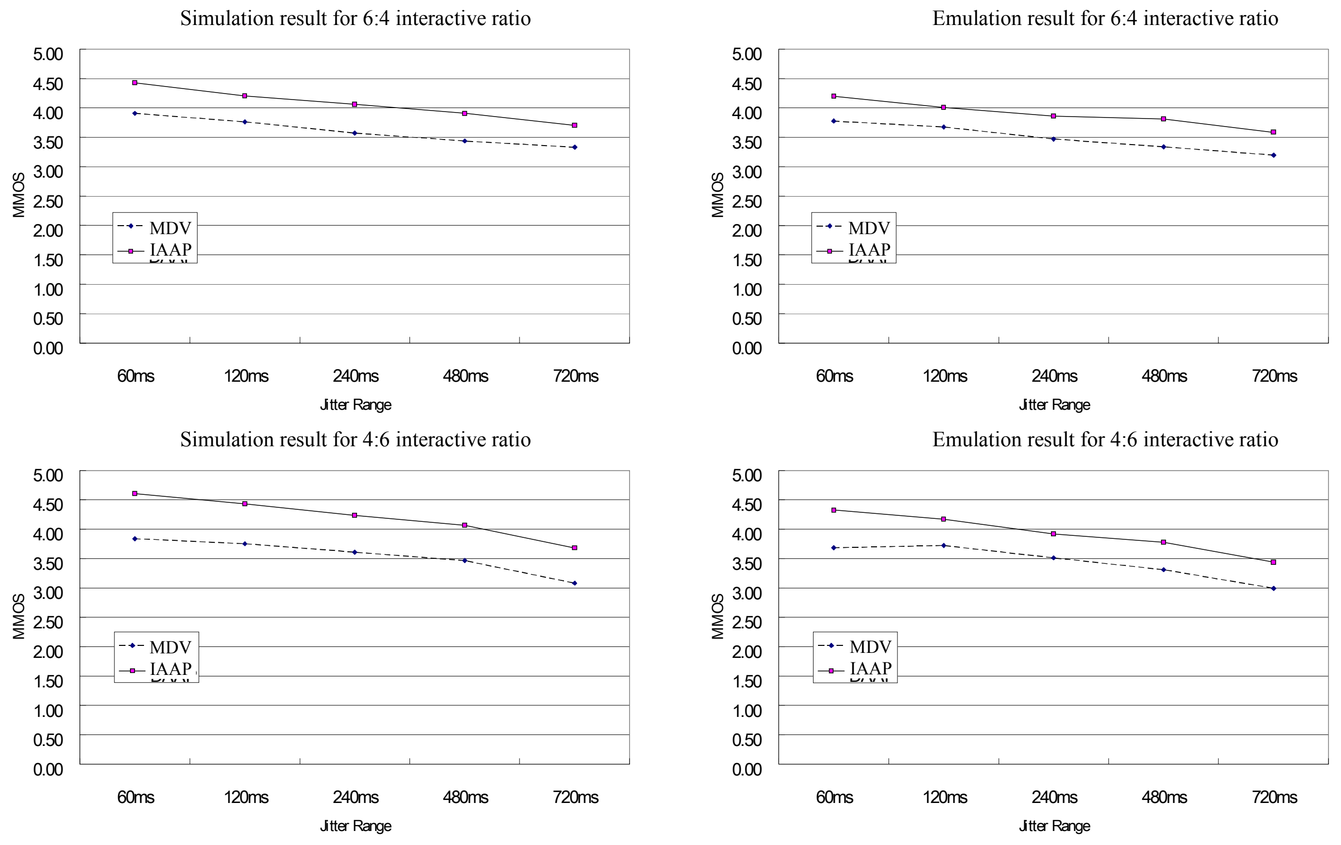
Fig. 7. Performance result of SCAAP, ECAAP MDA and RFC playout algorithms (a) Simulation results for G.711, G.721 and GSM-FR codec. (b) Emulation results for G.711, G.721 and GSM-FR codec.

the large jitter has more improvement, perhaps because a larger jitter gives more opportunities for optimization, and a lower bit rate codec is more sensitive to packet loss, that requires more accurate playout delay control.

As shown in Fig. 7 (b), the emulation results are similar to the simulation results as well. However, the dissimilarity between them is caused by more uncontrollable factors existing in the emulation. For example, the network emulator may cause the non-randomized delay. In this emulation, the improvement of ECAAP to MDV is from 7% to 20% and the improvement of SCAAP to MDV is from 5% to 16%. And the ECAAP and SCAAP to RFC improvement is slightly better than ECAAP and SCAAP to MDV.

B. *IAAP Results*

Theoretically, IAAP can be applied to any playout algorithm with any codec type. Although the degree of improvement is different, IAAP actually improves the voice quality for any codec and playout algorithm. Thus, adopting the G.711 codec and comparing with MDV are sufficient to evaluate the IAAP. The results are evaluated under two interactive versus non-interactive ratios, they are 6:4 ratio and 4:6 ratio. The former is there are sixty percent of duration is in the interactive mode and forty percents of duration is in the non-interactive mode, and the latter is converse, that is, the period of interactive communication is longer than the period of one-way communication.



(a) Simulation (b) Emulation

Fig. 8 The results of IAAP and MDA for G.711 codec at 6:4 interactive to non-interactive ratio and 4:6 interactive to non-interactive ratio (a) Simulation results (b) Emulation results.

In Fig. 8 (a), simulation results show, IAAP has an average 8% to 12% out-performance over MDV for 6:4 ratio, and 13% to 17% improvement for 4:6 ratio. In Fig. 8 (b), emulation results show that IAAP has 6% to 10% improvement over the MDV for 6:4 ratio and 10% to 13% improvement for 4:6 ratio. Simulation and emulation results are quite consistent although the improvements in the former are slightly less than those in the emulation. And we can also observe that the improvement of IAAP increases as the ratio of interactive mode over non-interactive mode decreases.

C. Performance Analysis

In our survey, since the MDV algorithm is invented, the playout algorithms are difficult to gain improvement with the jitter-oriented approaches; even the best algorithm GBA only has marginal improvement We are the first recent algorithm to break this bottleneck, because we think in other way, we consider the codec and interactive mode with perceptual quality.

Moreover our performance measurement is done thoroughly in the speech quality for both simulation and emulation which means our results are more objective and accurate than prior papers. In our evaluation, ECAAP and SCAAP outperform MDV algorithm by 7% to 20% and 5% to 10% on average. IAAP also shows 10% and 15% improvements in the 6:4 and 4:6 ratios respectively. We can also combine these two approaches in a system. Such a system will has more out-performance than MDV. Therefore, our algorithms can gain more improvement than the previous best algorithm.

VI. CONCLUSION

In this paper we have proposed two novel approaches to optimize playout delay, as well as a new LMOS-DMOS measurement for two-way communication quality. Because the jitter-oriented algorithm is difficult to improve the performance, the prior adaptive algorithms out-performance over the basic adaptive algorithm MDV is marginal.

In our emulation and simulation, CAAP and IAAP are superior to the prior playout algorithms. Our CAAP and IAAP are capable of detecting the codec types and communication interactive mode and considering perceptual quality, thus they can improve more than conventional approaches. CAAP has two sub-stages in the implementation that is SCAAP and ECAPP, The former uses single estimated playout delay and ECAAP is more sophisticated by choosing the one among multiple estimated playout delays. Consequently, the ECAAP has better performance than SCAAP. The second IAAP intends to save more packet loss in the non-interactive mode. If the network is in large jitter and the system is also in non-interactive mode, the playout delay can be increased more than that it were in interactive mode.

For the quality measurement, we proposal a novel LMOS-DMOS measurement of playout algorithms that can model the loss and delay effects and is applicable to measure speech quality of two-way communication. In addition, if the more precise LMOS and DMOS parameters are defined, the more accurate result is yielded.

REFERENCES

[1] R. Ramjee; J. Kurose; D. Towsley; H. Schulzrinne, “Adaptive Playout Mechanisms for Packetized Audio Applications in Wide-Area Networks”, Proceedings of IEEE INFOCOM, Toronto, Canada, pp. 680 - 686, June 1994.

[2] M. Roccetti; V. Ghini; G. Pau; P. Salomoni; M. E. Bonfigli, “Design and Experimental Evaluation of an Adaptive Playout Delay Control Mechanism for Packetized Audio for use over the Internet”, November 1998.

[3] H. Schulzrinne; S. Casner; R. Frederick; V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[4] J. Pinto; K. J. Christensen, “An Algorithm for Playout of Packet Voice based on Adaptive Adjustment of Talkspurt Silence Periods”, 1999, http://citeseer.nj.nec.com/pinto99algorithm.html.

[5] P. DeLeon; C. J. Sreenan, “An Adaptive Predictor for Media Playout buffering”, Stanford, March, 2001, http://citeseer.nj.nec.com/deleon99adaptive.html.

[6] M. Kalman, “Analysis of Adaptive Media Playout for Stochastic Channel Models”. Final Report, Clarr Project EE 368c, Stanford University, March 2001.

[7] Y. J. Liang; N. Färber; and B. Girod, “Adaptive playout scheduling using time-scale modification in packet voice communications”, In 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings ICASSP01, May 2001. Salt Lake City, UT.

[8] Y. Mansour; B. Patt-Shamir, O. Lapid., “Optimal smoothing schedules for real-time streams”, Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing, Portland, Oregon, USA, July 16-19, 2000.

[9] P. Goyal; H. M. Vin; C. Shen; P.J. Shenoy, “A Reliable, Adaptive Network Protocol for Video Transport”, In Proceedings of INFOCOM'96, San Francisco, Pages 1080-1090, March 1996.

[10] N. Laoutaris; I. Stavrakakis; "Adaptive Playout Strategies for Packet Video Receivers with Finite Buffer Capacity", IEEE International Conference on Communications (ICC'01), Helsinki, June 11-14, 2001.

[11] ITU-T Recommendation G.107, “The Emodel, a computational model for use in transmission planning”, December 1998.

[12] ITU-T Recommendation G.108, “Application of the Emodel: a planning guide”, September 1998.

[13] ITU-T P.861 (1996): "Objective quality measurement of telephone-band (300 - 3 400 Hz) speech codecs“,02.1998.

[14] ETSI EG 201 377-1 V1.1.1, “Specification and measurement of speech transmission quality”, 02.2001.

[15] ITU-T G.862, “Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrow band telephone networks and speech codecs”, 02.2001.

[16] ETSI, TR101 329 v1.2.5, “General aspects of Quality of Service (QoS)”, 10.1998.

[17] L. Cottrell, Stanford Linear Accelerator Center (SLAC), “QoS on Best-Effort Networks”, Presented at April 25-27, 2001, http://www.slac.stanford.edu/grp/scs/net/talk/qos-itu-apr01/.

[18] ITU-T G.107 (05/00): “The E Model, A Computational Model for use in Transmission Planning”.

[19] ITU-T G.113, “Transmission Impairments”, 02.1996.

[20] ITU-T G.114, “One-way transmission time”, 05.2000.

[21] ETSI TR41.4-01-02-005, “Passive Monitoring for Voice over IP Gateways”, Costa Mesa, CA, February 2001.

[22] J. Rosenberg; L. Qiu; H. Schulzrinne, “Integrating packet FEC into adaptive voice playout buffer algorithms on the Internet”, INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, Volume: 3 , 2000 Page(s): 1705 -1714 vol.3.

[23] C. Padhye; K.J. Christensen; W. Moreno, “A new adaptive FEC loss control algorithm for voice over IP applications” Performance, Computing, and Communications Conference, 2000. IPCCC '00. Conference Proceeding of the IEEE International, 2000 Page(s): 307 – 313.

[24] H. Sanneck, “Concealment of lost speech packets using adaptive packetization “, Multimedia Computing and Systems, 1998. Proceedings. IEEE International Conference on , 1998, Page(s): 140 - 149.

[25] V. Abreu-Sernandez; C. Garcia-Mateo, “Adaptive multi-rate speech coder for VoIP transmission”, Electronics Letters , Volume: 36 Issue: 23 , 9 Nov. 2000 Page(s): 1978 –1980.

Kuo-Kun Tseng received his M.S. degree in Computer Science from National Chiao Tung University in 2002. He is presently a Ph.D. candidate in the Computer and Information Science at the above mentioned university. He has participated in projects of cable modem, VoIP gateway and QoS router as a senior R&D engineer from 1998 to 2002, His research interests are the algorithm and architectural designs on the network gateway.

Yaun-Cheng Lai received his B.S and M.S. degrees in Computer Science and Information Engineering from National Taiwan University in 1988 and 1990, respectively, and his Ph.D. degree in Computer and Information Science from National Chiao Tung in 1997. He joined the faculty of the Department of Computer Science and Information Science at National Cheng Kung University in August 1998. He then joined the faculty of the Department of Information Management at National Taiwan University of Science and Technology in August 2001 and has been an associate professor since August 2003. His research interests include performance analysis and protocol design on networks.

Ying-Dar Lin was born in Hsi-Lo, Taiwan, in 1965. He received the Bachelor's degree in Computer Science and Information Engineering from National Taiwan University in 1988, and the M.S. and Ph.D. degrees in Computer Science from the University of California, Los Angeles in 1990 and 1993, respectively. At UCLA Computer Science Department, he worked as a Research Assistant from 1989 to 1993 and worked as a Teaching Assistant from 1991 to 1992. In the summers of 1987 and 1991, he was a technical staff member in IBM Taiwan and Bell Communications Research, respectively. He joined the faculty of the Department of Computer and Information Science at National Chiao Tung University in August 1993 and is Professor since 1999. His research interests include design, analysis, and implementation of network protocols and algorithms, wire-speed switching and routing, quality of services, network security, and content networking. He has been a consultant for several high-tech companies and authored two books including a textbook on Computer Network Experiments. Dr. Lin is a member of ACM and IEEE. He is the founder and director of Network Benchmarking Lab (NBL) which reviews the functionality, performance, conformance, and interoperability of networking products.