Development of a Loss-Resilient Internet Speech Transmission Method

by

Nguyen Tuong Long Le

Submitted to the Department of Electrical Engineering
at
the Technical University of Berlin

in Partial Fulfillment of the Requirements for the
Diploma Thesis

May 1999

Thesis advisors:

Prof. Dr. Adam Wolisz
Department of Electrical Engineering
Technical University Berlin

Dipl.-Ing. Henning Sanneck
Competence Center for Global Networking (GloNe)
GMD Fokus

Die selbständige Anfertigung versichere ich an Eides Statt.

Berlin, den 30.5.1999

Unterschrift

Nguyen Tuong Long Le
Adamstr. 6
13595 Berlin

To my parents and my brother for their love, continued support, and encouragement

To my aunt and my cousins for not treating me as a relative but as a family member

To my grandfather and uncle Dung as the last kiss

Acknowledgments

First and foremost, I thank my family and my friends for their continued support, encouragement, and understanding that have made my education possible and my academic aspiration come true. I wish to thank Prof. Dr. Adam Wolisz and Dr. Mikhail Smirnov for giving me the fortunate opportunity to do this thesis at GMD Fokus[1].

My very special thanks go to my thesis advisor, Henning Sanneck, who has spent many hours discussing with me, guiding me to conduct research, and carefully proof-reading this thesis. I feel fortunate to have him as my thesis advisor. I also like to thank Sebastian Zander, Torsten Ackemann, Matthias Kranz, Sebastian Berg, Jiri Kuthan, and Laurensius Tionardi for many helpful hints and interesting discussions.

I am thankful to the Speech Processing Lab of the Electrical and Computer Engineering Department at Temple University, especially Dr. Wonho Yang and Prof. Dr. Robert Yantorno, for providing me with the Enhanced Modified Bark Spectral Distortion (EMBSD) program to evaluate the efficiency of the speech property-based Forward Error Correction (SPB-FEC) scheme and compare it with other FEC schemes.

Last but not least, I thank my family, my friends, and my colleagues for knowing when not to ask “How is your thesis going?”.

[1] German National Research Center for Information Technology
Research Institute for Open Communications Systems

Contents

List of Figures

List of Tables

Zusammenfassung

Diese Arbeit nutzt bestimmte Spracheigenschaften aus, um Audiopakete über ein verlustbehaftetes, paketvermittelndes und auf dem „Best-Effort"-Prinzip basierendes Netzwerk zu übertragen. Obwohl Paketverzögerung und Paketverlust einen schlechten Einfluß auf die Qualität der Sprachübertragung haben und außerdem korrelieren [SKBD98], beschränkt sich unsere Arbeit nur auf die Probleme der Paketverluste[1]. In dieser Arbeit leisten wir zwei Beiträge zu dem Forschungsgebiet der Sprachübertragung über das Internet:

- Verbesserung des Verfahrens „Adaptive Packetization and Concealment" (AP/C) [Sann98a], [Sann98b]. AP/C paketisiert und versendet die Audiodaten derart, daß die Empfänger ein verlorenes Paket von den korrekt empfangenen und benachbarten Paketen rekonstruieren und somit den Paketverlust verschleiern können. AP/C profitiert von der Tatsache, daß stimmhafte Signale wichtiger für die Sprachqualität als stimmlose Signale sind, und sendet stimmhafte Signale in kleineren Paketen und stimmlose Signale in größeren Paketen. Wenn Paketverlust bezüglich Paketgröße gleichverteilt ist, kommen mehr Abtastwerte von stimmhaften Signalen an und die Sprachqualität wird somit verbessert. Die Neuheit des Verfahrens AP/C ist, daß es die Phase der Sprachsignale bei der Paketisierung berücksichtigt. Dank der AP/Cs phasen-basierten Paketisierung kann keine Diskontinuität in den rekonstruierten Sprachsignalen gehört werden. Wir präsentieren die Verbesserungen, die AP/C helfen, die Periodizität und somit die Phase der Sprachsignale zuverlässiger zu bestimmen. Außerdem entwickeln wir ein Verfahren, das AP/C mit der Methode Interleaving kombiniert, um das Problem der Burstverluste von Paketen zu lösen[2].

- Entwicklung einer spracheigenschaft-basierten Vorwärtsfehlerkorrektur (FEC), um die für die Qualität der Sprachübertragung wichtigen Audiodaten zu schützen. AP/C kann nur Coder unterstützen, die mit feiner Granularität von Sprachblöcken z.B. PCM Coder arbeiten können, und nicht die modernen frame-basierten Coder z.B. G.723.1 [ITU96a] und G.729 [ITU96b], die mit konstanten Längen von Sprachblöcken arbeiten. Unsere Experimente führten uns zu der Erkenntnis, daß der Verlust von Sprachblöcken am Anfang eines stimmhaften Signals eine erhebliche Verschlechterung der Sprachqualität verursacht, während der Verlust von anderen Sprachblöcken gut vom Decoder verschleiert werden kann. Wir haben eine spracheigenschaft-basierte Vorwärtsfehlerkorrektur entwickelt, die nur diese Sprachblöcke schützt[3], um eine bessere Sprachqualität bei geringer Erhöhung der verbrauchten Bandbreite zu erzielen. Unsere Lösung kann eingesetzt werden, um die modernen frame-basierten Coder zu unterstützen, die mit dem AP/C Verfahren nicht arbeiten können.

[1] Die Probleme der Paketverzögerung und ihrer Schwankung wurden mit dem Konzept des Playout-Buffers in [Ramj94], [Schu92], [Schu95] und [MoKT98] behandelt.

[2] Das derzeitige AP/C Verfahren korrigiert nur einzelne Paketverluste, die im Internet und im MBone dominant sind.

[3] Die anderen Sprachblöcken eines stimmhaften Signals sind durchaus wichtig für die Sprachqualität. Jedoch wird der Verlust von diesen Sprachblöcken vom Decoder gut verschleiert, so daß es nicht nötig ist, sie zu schützen.

Abstracts

This thesis exploits certain properties of speech signals to transmit audio packets over a "best effort" packet-switched network. Although both packet delay and packet loss have an adverse impact on the quality of speech transmission and furthermore are correlated [SKBD98], this thesis only concentrates on the problems of packet loss[1]. In this thesis, we present two contributions to the research topic of speech transmissions over the Internet:

- Improvements of the Adaptive Packetization and Concealment (AP/C) [Sann98a], [Sann98b]. AP/C packetizes and transmits audio data in such a way that receivers can reconstruct a lost packet from the correctly received adjacent packets and conceal the packet loss. AP/C exploits the fact that voiced signals are more important to the speech quality than unvoiced signals and sends voiced signals in small-size packets and unvoiced signals in large-size packets. If the packet loss probability is equally distributed regarding to the packet size, more samples of voiced signals are received, leading to a better speech quality. The novelty of AP/C is that the phase of speech signals is taken into account when audio data is packetized at the sender. Thanks to the AP/C's phase-based packetization, no discontinuities can be heard in the reconstructed speech signals. We present some improvements that help AP/C to determine the periodicity and thus also the phase of speech signals more reliably. Besides, we also develop a scheme that combines AP/C and interleaving to cope with the problem of packet burst loss[2].

- Development of a speech property-based Forward Error Correction (FEC) scheme to selectively protect audio data that is important to the quality of speech transmissions. AP/C can only support coders that can operate on flexible size of speech frames, e.g. PCM coder, and cannot support modern frame-based coders, e.g. G.723.1 [ITU96a] and G.729 [ITU96b], that operate on a fixed size of speech frames. Our experiments have led us to the knowledge that the loss of speech frames at the beginning of a voiced signal results in significant degradation of quality of speech transmissions while the loss of other speech frames has a rather harmless effect. We have developed a speech property-based FEC scheme that only protects these speech frames[3] to achieve a better quality of speech transmissions at a small increase of bandwidth consumption. Our speech property-based FEC scheme can be applied to support coders that operate on fixed size of speech frames and cannot be supported by AP/C.

[1] The problems of packet delay and delay variation are addressed by adaptive playout buffer in [Ramj94], [Schu92], [Schu95], and [MoKT98].

[2] The current AP/C scheme assumes that isolated packet loss is predominant in the Internet and in the MBone. Thus, AP/C's performance decreases with increasing packet burst loss.

[3] The other voiced frames are also important to the speech quality. However, the loss of these frames are concealed fairly well by the decoder so that it is unnecessary to protect them.

1. Introduction

1.1 Introduction and Motivation

In recent years, both the general public and the research community have been giving a great interest to speech transmission over the Internet despite its mediocre speech quality. Up until now, this mediocre quality of speech transmission over the Internet has been accepted due to the novelty and the cheaper flat-rate charge compared to that of the traditional telephony. Although its cheap flat-rate charge might not remain in the future, speech transmission over the Internet is still very attractive because it can be combined with other Internet applications to provide interactive multimedia communications that are impossible (or at least very difficult) to deploy over the traditional telephone system. In order to obtain a wide acceptance from the public, however, there is still much work to be done to improve the quality of speech transmissions over the Internet.

The reason for the mediocre quality of speech transmissions over the Internet is that today's packet-switched networks, e.g. the Internet, are based on the "best effort" principle which does not guarantee a minimum packet loss rate and a minimum delay of packet transmission required by speech transmission over the Internet. Data packets sent over the Internet can be lost, corrupted, duplicated, or reordered. The "best effort" principle results in adverse affects on the quality of speech transmission over the Internet, e.g. audio packets can be discarded when routers or gateways are congested. Due to the real-time requirement of speech transmission, it is usually impossible for the receivers to request the sender to retransmit the lost packets. Besides, audio packets that do not arrive before their playout time are considered lost and cannot be played when they are received.

In general, there are two approaches to the above problem:

- network-based approach: this approach is to change the network technology to provide speech transmissions over the Internet with appropriate services such as a guarantee of minimum delay and minimum packet loss rate of the audio data packets. This approach requires a major change in the architecture and infrastructure of the Internet. Solutions for this approach might be available in the long term and is currently being tackled by reservation protocols such as the Resource ReSerVation Protocol (RSVP) [Brad97] and the Internet Integrated Services architecture [BCSh94].

- application-based approach: this approach is to develop and to implement robust algorithms that can cope with the "best effort" transmission service of the Internet. Solutions for this approach can be available in the short term if applications can find ways to cope with several problems of the unreliable service provided by the Internet such as delay, delay variations (also known as jitter), packet corruption[1], packet reordering, packet duplication, and packet loss.

This thesis follows the second approach and aims to exploit the properties of speech signals to transmit audio packets over a lossy packet-switched network based on the "best effort" principle. Although both packet delay and packet loss have an adverse impact on the quality

[1] Data corruption due to media bit errors is not a common problem in the Internet (except for wireless networks). However, it is mentioned for the sake of completeness.

of speech transmission and furthermore are correlated [SKBD98], this thesis only concentrates on the problems of packet loss[1].

1.2 Thesis Contributions

In this work, we present our solutions for speech transmission over the Internet that combine speech processing techniques and network supporting mechanisms. In our solutions, the sender exploits the speech properties and transmits a small amount of redundant information along with the audio data while the receivers apply error recovery/concealment to fill the gap of missing packets.

In this work, we contribute two results of our research to the research topic of speech transmissions over the Internet:

- In chapter 4, we present some improvements of the Adaptive Packetization and Concealment (AP/C) [Sann98a], [Sann98b]. In AP/C, the sender performs some pre-processing and transmits audio data in such a way that the receivers can reconstruct a lost packet from the correctly received adjacent packets and conceal the packet loss. Because voiced signals are in general more important to the speech quality than unvoiced signals, AP/C sends voiced signals in small-size packets and unvoiced signals in large-size packets. If the packet loss probability is equally distributed regarding to the packet size, more samples of voiced signals are received, resulting in a better speech quality. The novelty of AP/C is that AP/C takes the phase of speech signals into account when audio data is packetized at the sender. Thanks to the AP/C's phase-based packetization, no discontinuities are noticeable in the reconstructed speech signals. We present some improvements that help AP/C to determine the periodicity and thus also the phase of speech signals more reliably. Besides, we also develop a scheme that combines AP/C and interleaving to cope with the problem of packet burst loss[2].

- In chapter 5, we demonstrate that the loss of frames at different positions in an audio data stream leads to different levels of degradation of the speech quality. The loss of some frames causes a significant degradation of the speech quality while the loss of other frames are rather harmless. Thus, it is not necessary to protect all frames of an audio data stream but only those whose loss is badly recovered by the decoder's concealment algorithm. We then develop a speech property-based FEC scheme that selectively protects audio data that is important to the quality of speech transmissions. Our speech property-based FEC scheme obtains the same speech quality at a smaller increase of bandwidth consumption compared to other current FEC schemes.

1.3 Thesis Overview

This thesis is organized as follows:

[1] The problems of packet delay and delay variation are addressed by adaptive playout buffer in [Ramj94], [Schu92], [Schu95], and [MoKT98].

[2] The current AP/C scheme assumes that isolated packet loss is predominant in the Internet and in the MBone. Although this is a valid assumption [Hand97], [Bolo96], AP/C's performance decreases with increasing packet burst loss.

In chapter 2, we present some research work related to ours. Among this related work are RTP [Schu96], a transport protocol for real-time applications and NeVoT [Schu92], [Schu95], an audio communication tool over the Internet.

In chapter 3, we briefly discuss some of the important speech properties. These speech properties are exploited in our work.

In chapter 4, we present the AP/C scheme and our contribution to improve the reliability in determining phase and periodicity of speech signals and to combine the AP/C scheme with interleaving to allow for packet burst loss.

In chapter 5, we introduce our speech property-based FEC scheme that only protects speech frames at the beginning of a voiced signal to gain a better quality of speech transmissions over the Internet at a small increase of bandwidth consumption.

In chapter 6, we summarize our work and give an outlook for our future research.

2. Background Information

In this chapter, we review some research topics that are related to our work. In section 2.1, we take a look at the problems of speech transmission over the Internet and some research topics in this area. Section 2.2 provides an introduction to the transport protocol used by most interactive audio and video applications. Section 2.3 presents an introduction to an audio tool that is chosen to be extended with the AP/C scheme.

2.1 Impairments of Speech Transmission over the Internet

As already mentioned in chapter 1, the Internet over that we send audio data can delay, lose, duplicate and corrupt packets. In this section, we take a look at these problems and review the solutions to them.

2.1.1 Delay

Delay is mostly due to propagation time, switching and queueing time in routers or gateways, packetization time at the sender, depacketization time at the receivers, and voice coding and decoding time.

- Propagation time is the time the physical signals need to travel across the links along the path taken by the data packets. Propagation time presents a physical limit that cannot be reduced.
- Switching time is the time the router takes to switch a packet (e.g., extract the destination address from the packet header, use the destination address to check the routing tables for the output port, and send the packet out of the output port).
- Queueing time is the time a packet has to spend in the queues at the input and output ports before it can be processed. Switching and queueing time can be reduced by designing and implementing faster routers. This problem cannot be solved at the application layer and is not addressed furthermore in this work.
- Packetization and depacketization time are the time needed to build data packets at the sender and to strip off packet headers at the receiver. Packetization and depacketization time can be kept small by using simple protocols to reduce necessary protocol processing.
- Voice coding time is the time it takes to digitize speech signals and perform encoding conversion (if necessary) at the sender. If modern coders, e.g. G.723.1 or G.729 coder, are used, there is also an additional time needed to buffer *look-ahead* audio data samples[1] and carry out relatively complex algorithms to extract the redundancies in speech signals[2].
- Voice decoding time is the time needed to perform encoding conversion (if necessary) and convert digital data into analog signals at the receivers. Voice coding and decoding time are an intrinsic problem of digital voice communication and of modern codecs (if used). Furthermore, the processing time is fairly small (thanks to high performance of today's computers) and constant when compared to other delays. In this work, we do not address the problem of voice coding and decoding time.

2.1.2 Jitter

Jitter, also known as delay variation, is caused by packet queueing in routers. When several packets in a router compete for the same outgoing link, only one of them can be processed

[1] Total buffer time is 15 ms for the G.729 coder and 37.5 ms for the G.723.1 coder.
[2] In fact, these coders attempt to model the speech production process.

and forwarded while the others have to be queued. The result of packet queueing is that packets sent by the sender at equidistant time interval arrive at the receiver at non-equidistant time interval.

At the application layer, the impact of jitter can be reduced by keeping the received packets in a playout buffer and adding an extra amount of delay before they are played. This extra amount of delay is an engineering trade-off: it must be small enough to have no impact on interactive audio applications (e.g., audio conference) and it must be large enough to smooth out the jitter and to enable most of the delayed packets to arrive before their playout time. Playout mechanisms have been investigated in [Schu92], [Ramj94], and [Schu95].

2.1.3 Data Corruption

Thanks to today's technologies, a very low bit error rate can be achieved (approximately 10^{-14} with optical networks [Shac90]). Thus, the impact of data corruption is very small and can be neglected in speech transmissions over the Internet (except for wireless network scenarios that are not addressed in this work).

2.1.4 Packet Duplication

Due to routing problems, a single packet sent by the sender can be duplicated in the Internet and arrive several times at the receiver. Packet duplication occurs sometimes in the Internet and especially in the Multicast Backbone (MBone)[1]. The transport protocol for real-time applications (RTP) [Schu96], the protocol typically used for speech transmission over the Internet, provides a message sequence number to detect packet duplication. Packet duplication is a rather harmless problem because duplicate packets can be easily detected and discarded by the receivers.

2.1.5 Packet Reordering

Packet reordering occurs sometimes in the Internet when packets sent by a sender to the same receivers take different paths (or possibly different queues in a router) and the order of packet arrival is different from the order in which packets are sent. Applications can use the message sequence number of transport protocols such as RTP to detect packet reordering and re-sort the packets.

2.1.6 Packet Loss

Packet loss often occurs in the Internet and especially in the MBone when a router goes down or becomes congested, i.e. it receives more packets to forward than it can process. Packet loss in the Internet and in the MBone is a frequent and also the most serious problem that speech transmissions over the Internet have to face. Applications can use message sequence number of transport protocols such as RTP to detect a packet loss.

In order to provide an acceptable quality, a loss recovery must be performed when the packet loss rate exceeds a tolerable limit. In general, there are two classes of mechanisms to perform loss recovery: closed loop and open loop mechanisms.

[1] MBone is an overlay network put on the Internet that supports multicast, by which a sender can send a single copy of its data packets to several receivers.

2.1.6.1 Closed Loop Mechanisms

In closed loop mechanisms such as Automatic Repeat Request (ARQ), receivers request the sender to retransmit the lost packets when they detect a packet loss, i.e. a jump in the message sequence number. These mechanisms cannot be applied in speech transmissions over the Internet due to the real-time requirements of interactive applications. However, these mechanisms are rather attractive for applications without strict real-time requirements such as audio/video on demand. Furthermore, the sender can combine the ARQ mechanisms with Forward Error Correction (FEC) and send a single repair packet to help the receivers in a multicast group to recover different lost packets [RuKT98]. In some hybrid FEC/ARQ schemes, FEC is used to recover from packet losses but when too many errors occur, ARQ is applied to reduce packet loss rate and prevent error propagation[1].

2.1.6.2 Open Loop Mechanisms

In open loop mechanisms, receivers do not request the sender to retransmit the lost packets when they detect a packet loss, i.e. a jump in the message sequence number. Instead, the receivers try to recover the lost packets without further interaction with the sender. The open loop mechanisms are further divided into two classes: sender-supported and receiver-only.

In the sender-supported open loop mechanisms, the sender helps the receivers to recover the lost packets by transmitting some redundant information along with the audio data so that some or all lost packets can be reconstructed from the received data and the redundant information. The redundant information can be sent in separate packets [RuKT98] or piggy-backed in an audio data packet [Bolo97]. Source coding and channel coding are typical methods of the sender-supported open loop mechanisms.

- In source coding methods, the sender transmits the same data samples possibly coded with different encodings several times in different packets that are transmitted with an offset in time. If one of the copies of the same data samples arrives, the receiver uses the corresponding decoder to decode to get the data [Hand95], [PoRM98], [Bolo97].
- In channel coding methods, the sender applies well-known methods of information theory to compute redundant packets and send them along with the data packets to the receivers. If the receiver receives a sufficient number of data and redundant packets, it can recover all lost data packets [Rizz97]. This method is called Forward Error Correction (FEC). Another method is to divide the audio data stream into data units, scramble them, and then put consecutive data units into different packets [Chen97]. These data units are descrambled and put into consecutive order at the receivers. This method is called interleaving and is applied to spread burst packet loss. Recently, there is some research that combines interleaving and FEC to improve the quality of audio transmission [Chen98]. Because burst packet loss has a worse impact on the speech quality than isolated packet loss does, interleaving can achieve a higher performance without redundancy overhead. However, interleaving is only effective when applied to a rather high number of data units. Hence, it significantly increases the packetization delay because it needs to buffer the data units. Thus, interleaving is not suited for interactive applications but for playback applications such as Internet radio or audio/video on demand.

[1] In modern coders such as G.723.1 and G.729, the loss of a data packet also has an impact on the following packets [Rose97].

Besides source coding and channel coding methods, there are mechanisms in that the receivers try to recover lost packets without redundant information or sender's support. These mechanisms are called receiver-only. Typical receiver-only open loop mechanisms use silence or noise, or repeat/interpolate the correctly received packets to fill in the gap of a lost packet [Sann95], [PeHH98].

Clearly, receiver-only open loop mechanisms are the most bandwidth-effective method to recover packet loss. However, because speech signals are at best quasi-stationary over a short period of time and cannot be exactly predicted, these mechanisms only work well when the loss rate is low (typically under 15% [PeHH98]). If the packet loss rate exceeds this limit, speech quality drops significantly (this is also due to an increasing packet loss correlation under high packet loss rates that renders the receivers' prediction/interpolation unusable). Thus, some network mechanisms must be applied and/or some redundant information must be transmitted to reduce the application loss rate, i.e. the loss rate seen by the application after loss recovery has been applied, and deliver an acceptable speech quality. Under a constant loss rate, sending an increased amount of redundant information clearly reduces the application loss rate. However, when the networks are congested, sending too much redundant information aggravates the problems of congestion and worsens the delivered speech quality [PoRM98].

2.2 Transport Protocol for Real-time Applications (RTP)

Currently, most interactive audio and video applications use the real-time transport protocol (RTP) [Schu96] for data transmission with time constraints. RTP itself does not provide Quality of Service (QoS) guarantee or timely delivery of data but relies on lower-layer services to do so. RTP runs on top of existing transport protocols, typically UDP, and provides real-time applications with end-to-end delivery services such as payload type identification, timestamping, message sequence number, delivery monitoring, etc. RTP provides transport of data with a notion of time to enable the receivers to reconstruct the timing information of the sender. Besides, RTP messages contain a message sequence number to allow applications to detect packet loss, packet duplication, or packet reordering.

RTP is extended by the RTP control protocol (RTCP) that exchanges member information in an on-going session. RTCP monitors the data delivery and provides the users with some statistical functionality. The receivers can use RTCP as a feedback mechanism to notify the sender about the quality of an on-going session.

An RTP message contains an RTP header followed by the RTP payload (e.g., audio data or video data). An RTP message of the current version (version 2) is shown in Figure 1.

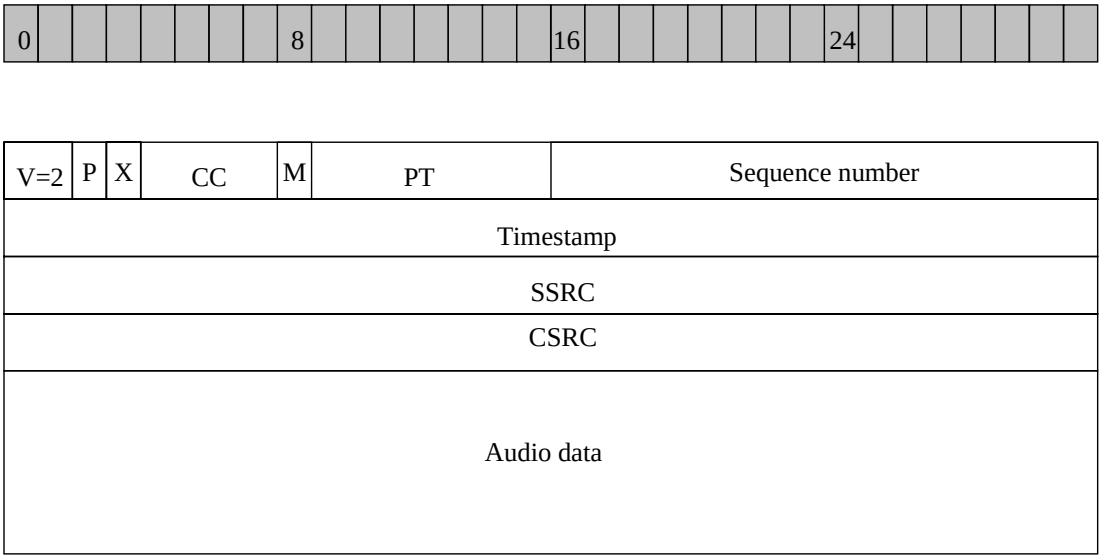


Figure 1. RTP message.

Below is a short explanation for some fields of the RTP message shown in Figure 1. More details can be found in [Schu96].

- CSRC count (CC): 4 bits
  The CSRC count specifies the number of CSRC identifiers contained in the CSRC list.

- Payload type (PT): 7 bits
  The payload type specifies the format of the RTP payload following the fixed header.

- Sequence number: 16 bits
  The sequence number counts the number of the RTP packets sent by the sender and is incremented by one for each transmitted packet. The sequence number allows the receivers to detect packet loss, packet duplication, out-of-order packet delivery, and to restore packet sequence.

- Timestamp: 32 bits
  Timestamp reflects the sampling instant of the first data sample contained in the payload of RTP packets and is incremented by one for each data sample, regardless of whether the data samples are transmitted onto the network or are dropped as silent. Timestamp helps the receivers to calculate the arrival jitter of RTP packets and synchronize themselves with the sender.

- Synchronization source identifier (SSRC): 32 bits
  The SSRC field contains a random value that is chosen by a source. It is used by a source as the identifier for each of its data streams[1] and must be unique within a particular session. RTP specifies a mechanism for resolving collisions in the case that two sources randomly choose the same SSRC.

- Contributing source identifier list (CSRC list): 0 to 15 CSRC items, 32 bits each

[1] If a participant produces multiple data streams in an RTP session (e.g., from separate video cameras), each of them must be identified by a different SSRC.

CSRC list contains a list of SSRC identifiers of the sources whose data is combined by an intermediate system to generate the payload of a new RTP packet. The intermediate system is called a mixer and must use its own SSRC identifier for the new RTP packet.

2.3 Network Voice Terminal (NeVoT)

Among several available audio tools such as rat [RAT], vat [VAT], and FreePhone [Free], NeVoT is chosen to be extended with the AP/C scheme because it is a component of the conference environment of the Multimedia Internet Terminal (MiNT) used at GMD Fokus. Besides, NeVoT also excels in its good source documentation and its good coding style.

NeVoT [Schu92] is an audio tool that supports voice communication over the Internet. It allows users to participate in different audio sessions at the same time. In NeVoT, an audio session is specified by an IP address (unicast or multicast) and a port number of a transport protocol (e.g., TCP or UDP [Stev94]). NeVoT supports a range of audio encodings (e.g., PCM[1], LPC[2] [Dell93], and GSM[3] [Dege96]) that can be changed by users during an audio session and allows participants of an audio session to use different audio encodings at the same time. If the sender and receivers use different audio encodings, NeVoT performs the necessary conversion.

NeVoT has a graphical user interface implemented in Tcl/Tk [Ous95] at that the user can specify his audio encodings, audio transport protocol (RTP or vat), packet size, minimum and maximum playout delay, etc.

Figure 2 presents a snapshot of NeVoT's graphical user interface for audio parameter configuration.

NeVoT typically operates over the protocol stack RTP/UDP[4]/IP[5] [SiSc98]. However, it also supports the vat protocol to enable interoperability with the VAT audio tool from Lawrence Berkeley Laboratory (LBL) [VAT].

NeVoT runs UDP as a transport protocol. In UDP, data is transmitted in datagrams. UDP is a connectionless protocol and provides an unreliable service, i.e. datagrams might not arrive or might arrive out of order. A mechanism to detect and to recover loss or reordering of datagrams is left to the higher protocol layers. In NeVoT, RTP/RTCP runs on top of UDP and uses the message sequence number in the RTP header to detect packet loss, packet duplication, and out-of-order packet delivery. Besides, RTP/RTCP also provides the applications with necessary side information (e.g. timestamp) to carry out control flow functionality and to play audio data arriving at the receivers.

[1] pulse code modulation
[2] linear predictive code
[3] Groupe Speciale Mobile
[4] User Datagram Protocol [Stev94]
[5] Internet Protocol [Stev94]

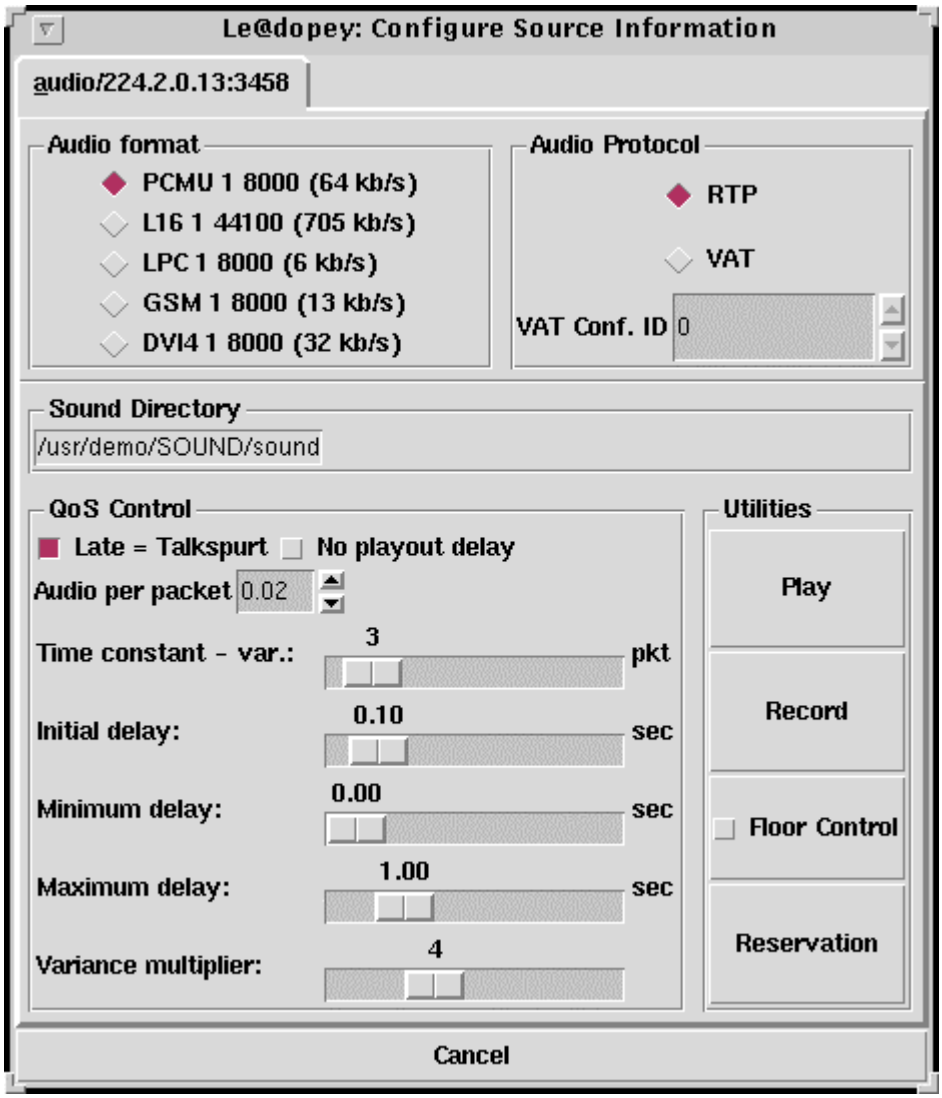


Figure 2. NeVoT's graphical user interface for audio parameter configuration.

Below we present a short discussion about the program structure of NeVoT. More details about NeVoT can be found in [Schu92], [Schu95].

NeVoT functions as an interface between the audio input device (e.g., microphone) and the network interface at the sender and between the network interface and the audio output device (e.g., workstation speaker) at the receivers. The network, over which audio packets are sent, can lose, duplicate, corrupt, or reorder audio packets. The task of NeVoT is to reduce the adverse effects of these problems as far as possible.

Figure 3 depicts the program structure of NeVoT.

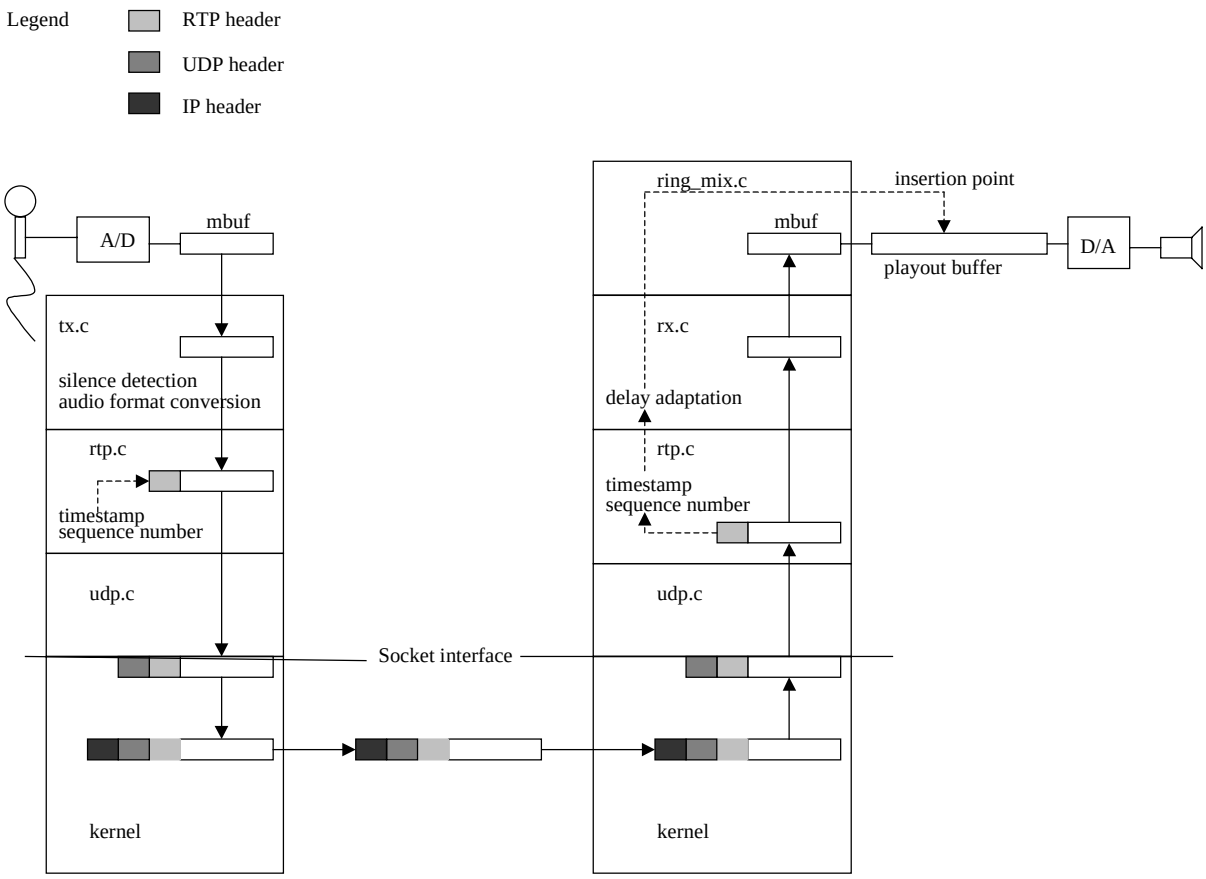


Figure 3. Program structure of NeVoT

NeVoT copies audio data from the audio input device or an audio file to the network interface and from the network interface to the audio output device. These two flows are independent but they are coupled by the following mechanism: Every time a full block of audio data, typically 20 ms or 160 samples of *μ-law* audio data, is delivered by the audio input device, a block of audio data with equal length is copied from the playout (ring) buffer to audio output device.

In NeVoT, the function *tx() (tx.c)* is responsible for copying audio data from the audio input device to a *mbuf*[1]-like buffer. It calculates the energy estimate of the audio data block and decides whether the audio data block is a silent segment or not. Silent segments are suppressed and are not copied to the network interface in order to save bandwidth. A format conversion of the audio data is performed if necessary (e.g., if the audio input device delivers audio data in *μ-law* format while LPC encoding is chosen by the user). If the audio data block is not a silent segment, the *mbuf*-like buffer is passed down to the function *rtp_write() (rtp.c)*. *rtp_write()* prepends the RTP header to the *mbuf*-like buffer to create an RTP packet. It then calls the function *UDP_write() (udp.c)* to transmit the RTP packet onto the network.

In NeVoT, the function *rx() (rx.c)* presents an interface between the network and the audio output device. It calls the function *UDP_read() (udp.c)* to write packets arriving from the network to a *mbuf*-like buffer. The buffer is then passed to the function *rtp_read() (rtp.c)* which reads the information contained in the RTP header and write this information to a data structure called *sync (sync.h)*. The timestamp and sequence number of the last received packet are also kept in *sync* and allow to detect duplicate or reordered packets.

[1] Data structure of buffers organized as a dynamic linked list

Because the network can lose, reorder, duplicate, or deliver audio data packets in non-equidistant time interval (jitter), the function of *rx()* is to detect and compensate or reduce these impairments. Duplicate packets can be detected by the message sequence number and are simply discarded. Packet delay variation is smoothed out by keeping the arriving packets in a playout buffer for an additional amount of time. This additional amount of time is called playout delay. Figure 4 illustrates the idea of playout delay.

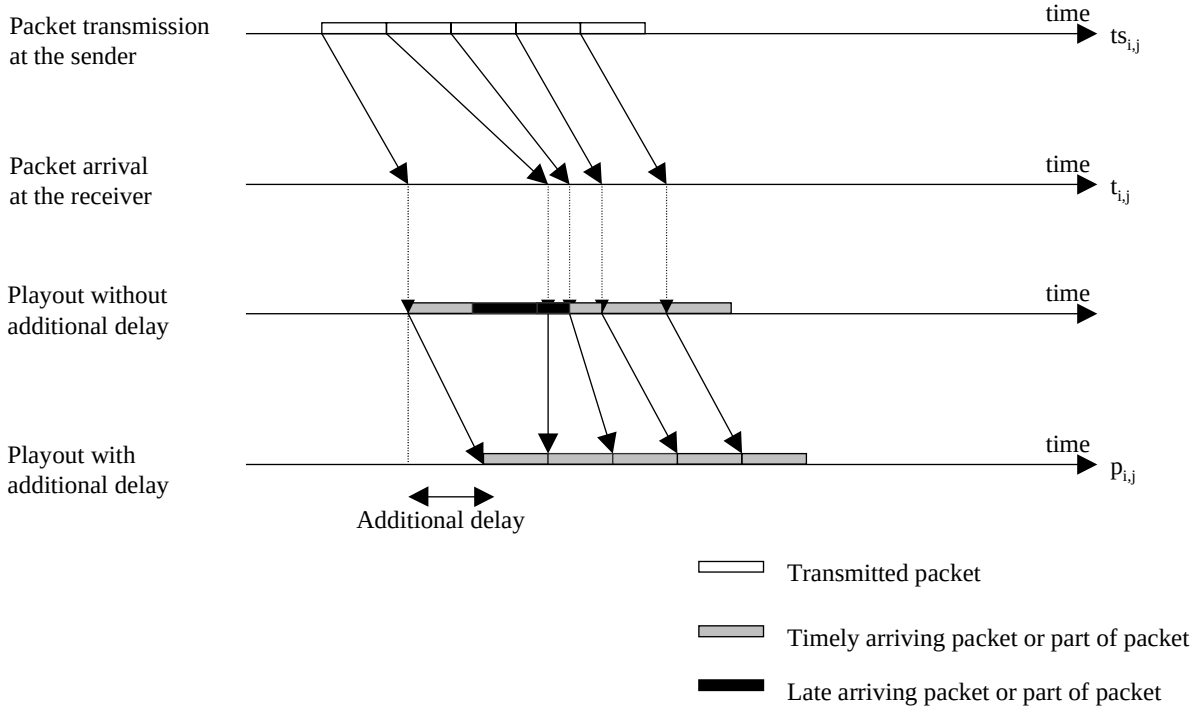


Figure 4. Playout delay at the receiver.

The playout delay is computed at the beginning of each talkspurt and remains constant throughout a talkspurt. Let $ts_{i,j}$, $t_{i,j}$, and $p_{i,j}$ be the timestamp, the arrival time, and the playout time of the *j*th packet of the *i*th talkspurt.

The delay variance of the *j*th packet of the *i*th talkspurt and is computed as:

$$v_{i,\,j} = (1-\alpha)\cdot \left| s_{i,\,j} - s'_{i,\,j} \right| + \alpha \cdot v_{i,\,j-1}$$

where $s_{i,j}$ is the difference between the playout time and the arrival time of the *j*th packet of the *i*th talkspurt[1].

$$s_{i,\,j} = p_{i,\,j} - t_{i,\,j}$$
$$p_{i,\,j} = ts_{i,\,j} + D_i$$
$$s'_{i,\,j} = (1-\alpha)\cdot s_{i,\,j-1} + \alpha \cdot s'_{i,\,j-1}$$
$$\alpha = 0.9$$

The playout delay D_{i+1} of the *(i+1)* talkspurt depends on the estimated delay and the delay variance of the last talkspurt (consisting of *x* packets) and is defined as follows:

[1] This difference is also known as slack time.

$$D_{i+1} = \mu \cdot v_{i,x} + D_{est,i+1}$$
$$\mu = 3..4$$

The estimated delay of the *(i+1)* talkspurt is defined as the minimum of the delay of the *i*th talkspurt's packets.

$$D_{est,i+1} = \min_j(d_{i,j})$$

where $d_{i,j}$ is the transmission delay of the *j*th packet of the *i*th talkspurt.

$$d_{i,j} = t_{i,j} - ts_{i,j}$$

3. Speech Properties

In this chapter, we will briefly discuss some properties of speech signals. In particular, we will take a look at the speech properties that are of major importance to our work, especially the characteristics of voiced and unvoiced sounds. See [Rabi78], [Dell93], and the references therein for more general and detailed discussions.

Speech signals are non-stationary and at best can be considered as quasi-periodic over a short period of time. Thus, they cannot be exactly predicted. Speech signals can be roughly divided into two categories: voiced and unvoiced sounds.

Voiced sounds are produced by pushing air from the lung through the glottis[1] with the shape and the tension of the vocal cords adjusted so that this flow of air causes them to vibrate in a relaxation oscillation. The vibration of the vocal cords results in a sequence of quasi-periodic pulses of air that excites the vocal tract. Thus, voiced sounds can be modeled by exciting a filter modeling the vocal tract with a quasi-periodic signal that reflects the air pulses produced by the vocal cords.

The rate of the vibration of the vocal cords' opening and closing are defined as the fundamental frequency of the phonation. It is often used interchangeably with the term *pitch*. Varying the shape of and the tension of the vocal cords can change the frequency of the vocal cords' vibration, i.e. the pitch.

In contrast to unvoiced sounds, voiced sounds have quasi-periodic characteristics in the time domain. This is a very important speech property that we will exploit in this work. Besides, the energy of voiced sounds is generally higher than that of unvoiced sounds. Furthermore, voiced sounds are more important to the perceptual quality than unvoiced sounds. This is another very important speech property that we will use in our work.

Unvoiced sounds are generated by forcing a steady flow of air at high velocities through a constriction region in the vocal tract to produce a turbulence. The location of the constriction region determines what unvoiced sound is produced. Unvoiced sounds are similar to random signals and have a broad spectrum in frequency domain. Random signals are usually used to model unvoiced sounds.

In speech transmissions over packet networks, the term *talkspurt* is often used to define a contiguous sequence of packets that have an energy higher than an energy threshold. A segment of audio data is defined as a silent segment if its energy is lower than this threshold. Silent segments can be suppressed in order to save bandwidth. However, a number of hangover silent packets immediately preceding or following a talkspurt should be transmitted to avoid clipping [Schu92], [Sann95], [Mino98].

Figure 5 and 6 present a segment of a voiced and an unvoiced sound in the time domain.

[1] The opening between the vocal cords [Rabi78].

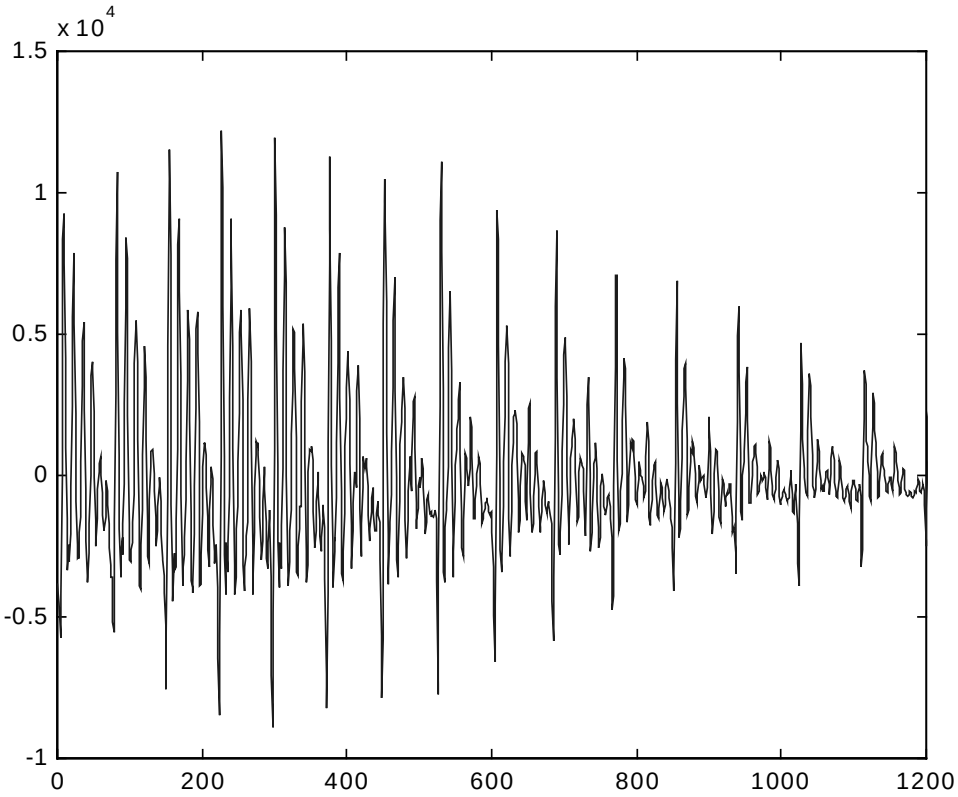


Figure 5. A segment of voiced sound in the time domain.

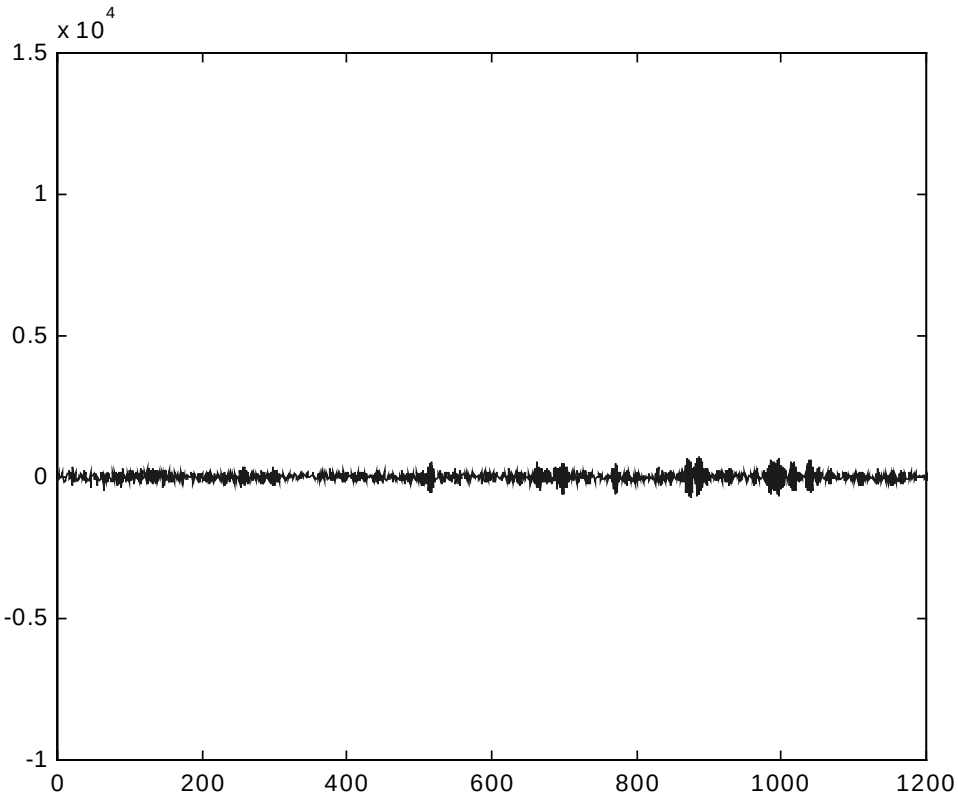


Figure 6. A segment of unvoiced sound in the time domain.

4. Adaptive Packetization and Concealment (AP/C)

4.1 Observations and Considerations

AP/C exploits the properties of speech signals to influence the size of audio packets at the sender and to conceal the packet loss at the receivers. In AP/C, packet size is small or large depending on whether the audio data contained in the packet is more or less important. In general, voiced sound signals are more important to the perceptual quality of speech transmission than unvoiced sound signals. Thus, if voiced sound signals are transmitted in small-size packets and unvoiced sound signals in large-size packets and if the packet loss probability is equally distributed regarding to the packet size, more samples of voiced signals (more important) are received than those of unvoiced signals (less important), resulting in a better speech quality.

The novelty of AP/C is that it takes the phase of speech signals into account when audio data is packetized at the sender. If a packet is lost and is reconstructed from the adjacent packets, no discontinuities can be heard in the reconstructed signal at the receivers thanks to the phase-based packetization.

Recent studies about characteristics of packet loss in the Internet have shown somewhat different results and have drawn somewhat contradictory conclusions. In [Bolo96], Bolot et. al. measured packet loss between the U.K. and France and showed that most packet losses are isolated and burst losses are negligibly small. In [YaKT96], Yajnik et. al. measured packet loss between distinct sites in the U.S. and Europe and concluded that isolated loss dominates although burst losses are rather high and not negligible. In [Hand97], Handley also carried out measurements on the MBone and concluded that isolated loss is predominant and burst loss, although statistically noticeable, is not sufficient to significantly influence the design of most applications.

AP/C makes the assumption that the most packet losses are isolated and that the packets prior and next to the lost packet are correctly received. AP/C conceals the loss of a single packet at the receivers by filling the gap of the lost packet with data samples from the adjacent packets. Due to the fact that voiced sounds are quasi-periodic, reconstruction with sender-supported pre-processing described below works reasonably well for voiced sounds. Reconstruction works less well for unvoiced sounds due to their random nature. However, this is not very critical because unvoiced sounds are less important to the perceptual quality than voiced sounds.

In section 4.6, we present a scheme that combines AP/C with interleaving to cope with the problem of packet burst loss.

4.2 Overview

Coupled with the above observations and considerations, AP/C's concealment of packet loss operates like described below:

At the sender, the pitch period of the audio data is estimated. An audio "chunk" is defined as a segment of audio data that has the length of the estimated pitch period. In order to alleviate the overhead of protocol header, two audio "chunks" are copied into an audio packet and then transmitted onto the network. When a packet loss is detected at the receiver, adjacent

"chunks" of the previous and following packet are used to reconstruct the lost "chunks". Information on intra-packet boundary between the two chunks of the packet and of the previous packet is transmitted as additional information to help the receivers with the concealment process.

Figure 7 shows the overview of the AP/C scheme.

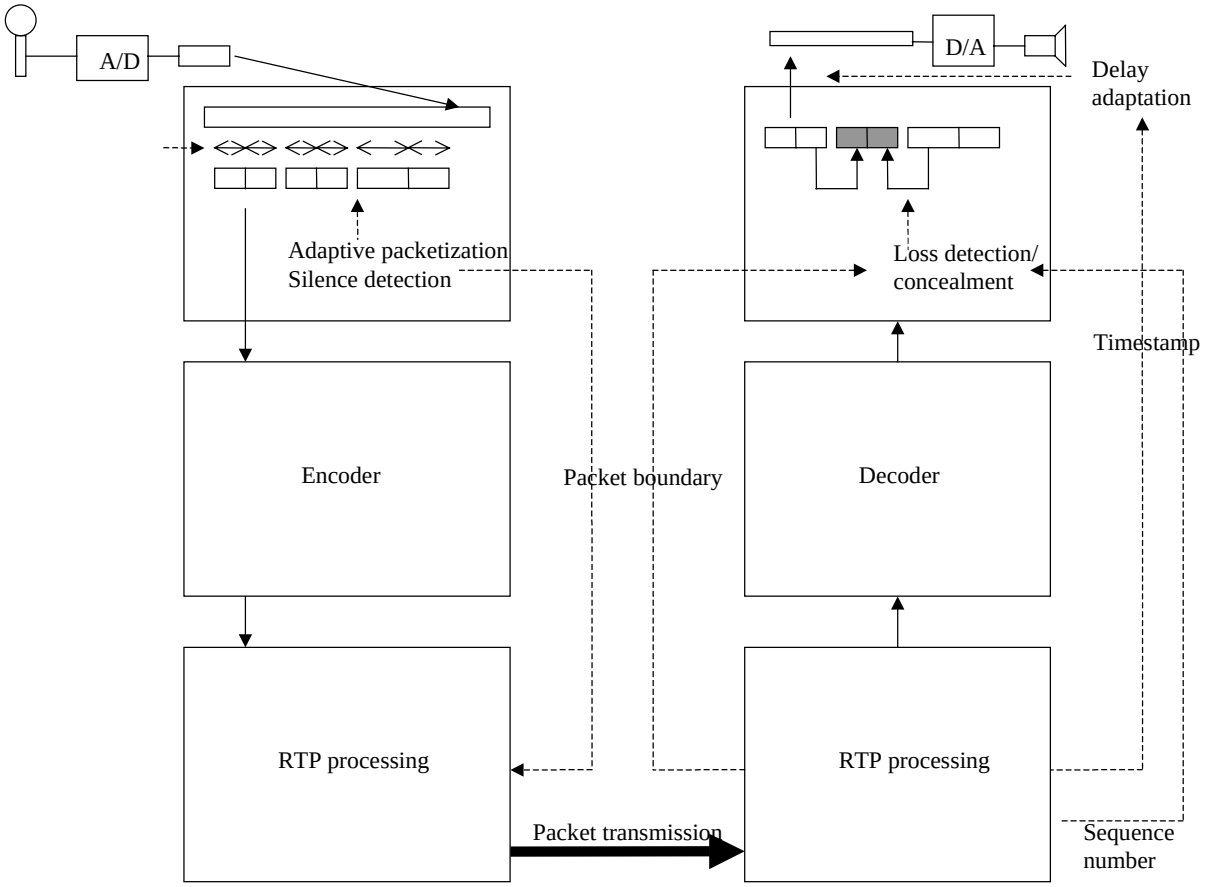


Figure 7. Overview of the AP/C scheme.

4.3 Sender Algorithm

The sender part of AP/C periodically fetches the audio data samples from the audio input device into a buffer and estimates the pitch period of an input segment of $3T_{max}$ samples (T_{max} is the correlation window size[1]). The estimation of the pitch period is performed as follows: At first, the auto-correlation of the input segment is calculated. Then the maximum value second to the maximum value at zero[2] of the auto-correlation is searched for. This sub-maximum value, its position, and the auto-correlation itself help to make the decision whether the input segment is voiced or unvoiced. If the input segment is classified as voiced, the position of this sub-maximum is said to be the estimated value of the pitch period because the input segment shifted by that length of that samples is most similar to itself. A segment of audio data that has the length equal to the found pitch period is defined as an audio "chunk". If the input segment is classified as unvoiced, the sender takes an audio chunk that has a length of T_{max}.

[1] In AP/C, T_{max} is chosen to be 160.
[2] Logically, the absolute maximum value of the auto-correlation is found at 0 because a signal without any shift is most similar to itself.

The found audio "chunk" is copied from the buffer into an audio packet and the start position of the input segment is moved forward by the length of the audio "chunk".

Auto-correlation can be calculated in the time domain according to its definition:

$$r_{xx}(k) = \sum_{n} x(n) \cdot x(n+k)$$

Another way to compute the auto-correlation is to compute the Fourier transform $X(j\omega)$ of the input segment and then use this result to calculate the Fourier transform of the auto-correlation $r_{xx}(k)$ (i.e., $R_{xx}(j\omega)$). The auto-correlation $r_{xx}(k)$ is the inverse Fourier transform of $R_{xx}(j\omega)$. This can be easily shown by a simple proof:

$$r_{xx}(k) = \sum_{n} x(n) \cdot x(n+k) = x(k) * x(-k) \leftrightarrow X(j\omega) \cdot \overline{X(j\omega)}$$
$$\Rightarrow R_{xx}(j\omega) = X(j\omega) \cdot \overline{X(j\omega)}$$

This method is found to be faster and consumes less CPU resource than the first one: Computation in the time domain of K points of the auto-correlation function for an N point window requires on the order of $K \cdot N$ multiplications and additions while computation of the auto-correlation function by the second method requires on the order of $N \cdot \log_2 K$ multiplications and additions [Rabi78].

In our work, we use the C routines of FFTW to compute the Fourier transform. FFTW is an acronym for "Fastest Fourier Transform in the West" and is developed by the Massachusetts Institute of Technology. Besides its very good performance, FFTW also supports Fourier transform of any size and has a very good documentation for installation and functional description [FFTW].

In AP/C, a routine to detect speech transitions is implemented as follows:

$$\Delta p = |\, p(k) - p(k-1) \,| > \Delta T$$

where k is the number of the current audio chunk, p(k) and p(k-1) are the length of the current and previous audio chunk, and ΔT is a pre-configured number[1].

Speech transitions are further divided into voiced→unvoiced (**vu**) or unvoiced→voiced (**uv**):

$$p(k) < T_u \; and \; p(k-1) \geq T_u \; (\text{unvoiced} \rightarrow \text{voiced} : uv)$$
$$p(k) \geq T_u \; and \; p(k-1) < T_u \; (\text{voiced} \rightarrow \text{unvoiced} : vu)$$

where T_u is a pre-configured number[2].

[1] In AP/C, ΔT is defined to be 40.
[2] In AP/C, T_u is defined to be 120.

If a *vu* transition has taken place, the transition chunk is divided into two parts with $p(k_a)$ equal to p(k-1) and $p(k_b) = p(k)-p(k_a)$. If *k mod 2 = 0*, the chunk numbered k-1 is sent in a packet containing only one chunk.

When a uv transition is detected, a backward correlation of the previous chunk is computed to test whether voiced data is already contained in it. If this is true, the previous chunk is divided into two parts with $p(k_b\text{-}1) = p_{backward}(k\text{-}1)$ and $p(k_a\text{-}1) = p(k\text{-}1)-p(k_b\text{-}1)$ ($p_{backward}$ is the result of the backward correlation). Note that this is only possible if *k mod 2 = 0*, otherwise the previous chunk has already been sent. A solution to this problem would be to keep two unvoiced chunks in the buffer and check whether the third one has a speech transition. However, the gain obtained in speech quality when concealment is performed would not compensate the incurred additional delay.

4.4 Protocol Support

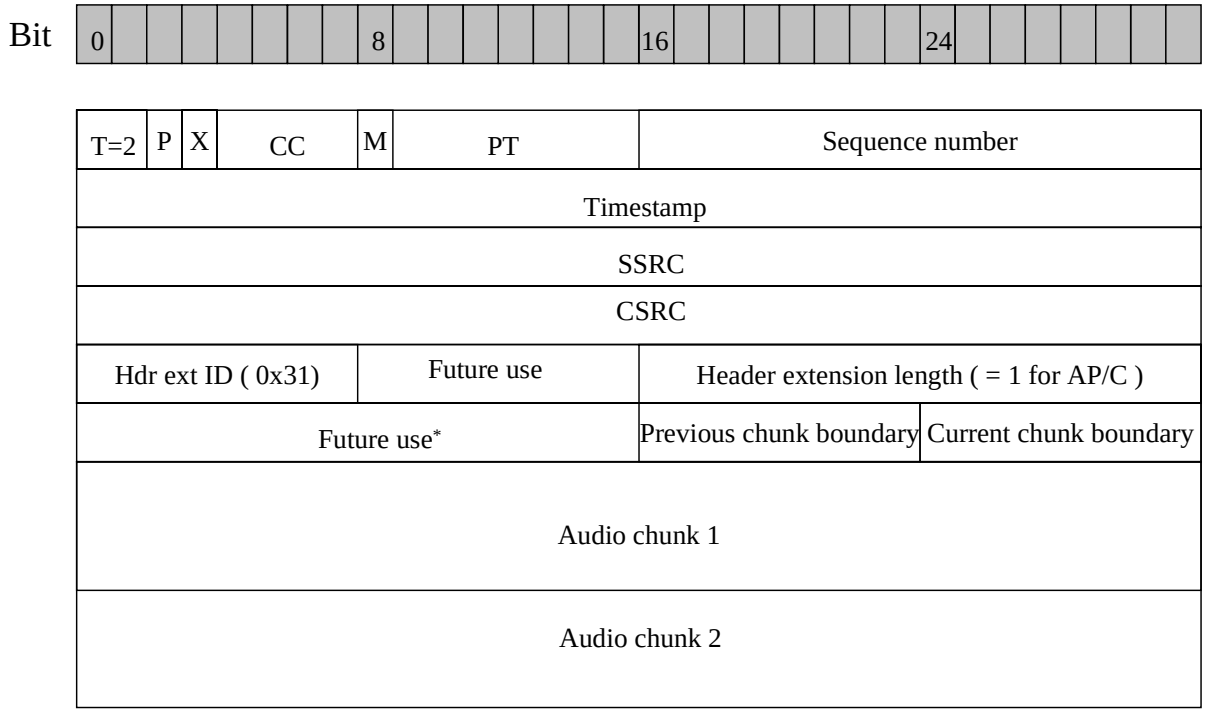
Along with audio data, the sender transmits information on intra-packet boundary in a packet (boundary between two chunks in a packet) to help the receivers determine the chunks' length and perform concealment. There are two ways to transmit this piece of information:

- A new payload according to [PeKH97] can be defined with audio data as the primary data block and the information on intra-packet boundary as redundant data. Although this is a general and recommended method for transmission of redundant information, we argue that it is an overkill for transmission of a two-byte piece of information on intra-packet boundary.
- The information on intra-packet boundary is contained in the RTP header extension and the RTP header extension bit is set.

We have chosen the second method for the sake of simplicity and to maintain backward compatibility with older implementations. In this case, old implementations without the concealment algorithm of AP/C can still correctly decode and play audio data packets with flexible length. Moreover, because voiced sounds are more important than unvoiced sounds and the AP/C sender sends voiced sound segments in small-size and unvoiced sound segments in large-size packets, a better speech quality is achieved even without the concealment algorithm of AP/C, assuming that packet loss probability is equally distributed regarding packet size. Interestingly, some applications (e.g. RAT and VAT) seem not to be able to handle data packets with RTP header extension and/or flexible length. The current version of NeVoT (version 3.35) also needs some small modifications to be able to receive and play audio data packets with RTP header extension and flexible length. FreePhone can correctly handle data packets with RTP header extension and flexible length[1].

Figure 8 shows an audio data packet with the information on intra-packet boundary.

[1] Thanks are due to Henning Sanneck for this interesting hint.



* This field can be used to contain boundary information of other packets.

Figure 8. AP/C packet.

4.5 Receiver Algorithm

The receivers can use RTP's message sequence numbers to detect a packet loss and only perform concealment when an isolated loss is found[1]. When concealment is performed, we use RTP's timestamp and information on intra-packet boundary to determine the length of the lost chunks.

$$l_2 = ts_3 - ts_1 - l_1$$

where ts_1 and ts_3 are the timestamp contained in the previous and following packet and l_1 and l_2 are the length of the previous and lost packet.

Given the information on the intra-packet boundary of the lost chunk (the length of the first lost chunk c_{21}) contained in the following packet, we can determine the length of the second lost chunk c_{22}.

$$c_{22} = l_2 - c_{21}$$

A problem occurs if silence suppression is enabled and there is a silent period between the lost packet and its adjacent packets. This is because RTP's sequence number increments by one for each transmitted packet and RTP's timestamp increments by one for each sampling period, regardless of whether data is sent or dropped as silent. In this case, l_2 is not the length

[1] In section 4.6, we present a scheme that combines AP/C with interleaving to cope with packet burst loss.

of the lost packet but the sum of the silent segment's and the lost packet's length and we can only determine the length of one lost chunk[1].

Due to the sender's pre-processing, we have:

$$c_{21}, c_{22} \leq T_{max} = 160$$

Because a silent period is usually longer than 20 ms or 160 μ-law audio data samples, the above problem can be easily detected when c_{21} or c_{22} is larger than T_{max}. A possible solution to this problem would be to send the two chunks' length along with the following packet. However, because this problem rarely occurs and our experiments have shown that the possible solution does not seem to have any observable improvement in speech quality, we stick to our choice and do not apply the concealment algorithm when this problem is detected.

Due to the pre-processing at the sender, the receiver can make the assumption that the chunks of a lost packet are similar to the adjacent chunks. The adjacent chunks (c_{12} and c_{31} in Figure 9) are resampled by a factor of $L = c_{12}/c_{21}$ and $L = c_{31}/c_{22}$ in the time domain to match the size of the lost chunks and then used to fill the gap of the lost packet. A linear interpolator is used to perform resampling (as in [VaNi89]). The replacement signals produced by the linear interpolator have a correct phase, thus avoiding discontinuities in the concealed signal that would lead to speech distortions, while still maintaining the pitch frequency at the edges. Due to the pre-processing at the sender, the lost and the adjacent chunks are most probably similar. Thus, the concealment operation causes no distortions in the concealed chunks. Figure 9 illustrates the concealment operation in the time domain.

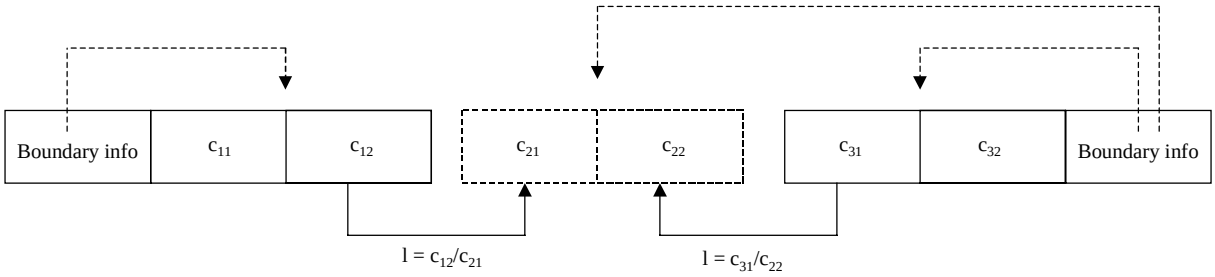


Figure 9. Concealment operation in the time domain.

Because voiced chunks are typically small and unvoiced chunks are typically large, the loss of transition chunks might lead to extreme expansion/compression operations of concealment. An upper bound f_{max} is set to avoid extreme expansion/compression operation of concealment. Resampling is only performed if the following condition is fulfilled: $| 1-L | < f_{max}$ (L is the resampling factor introduced above). In AP/C, f_{max} is chosen to be 50%. If this bound is exceeded, we have an extreme expansion or compression. If L is smaller than 1, we have an extreme expansion. Otherwise, we have an extreme compression.

If a high compression is detected, adjacent samples of the appropriate length are used to insert the gap. This operation is called "segment copy" and is illustrated in Figure 10. An audible discontinuity that might occur in the concealed signal can be avoided by overlap-adding the

[1] Note that we only transmit the position of the boundary between two chunks in a packet instead of the length of each chunks to save bandwidth.

replacement chunk with the adjacent ones. High expansions can be avoided by repeatedly copying a chunk until the gap of the lost chunk is filled ("periodic extrapolation") and then performing overlap-adding.

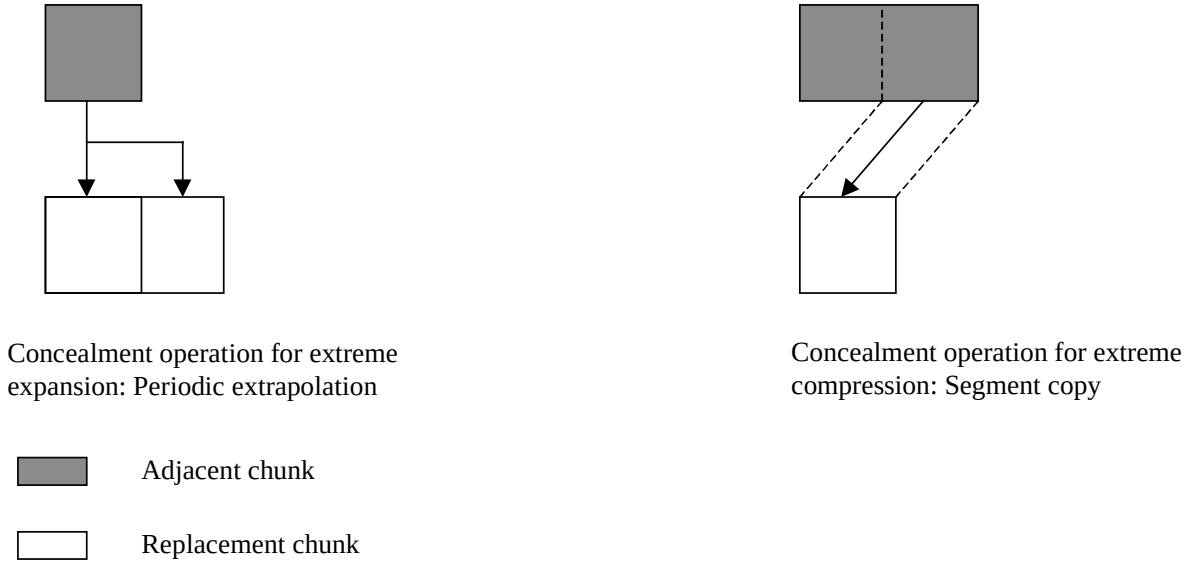


Figure 10. Concealment operation for extreme expansion/compression.

4.6 Improvement of AP/C

4.6.1 Extension of a More Reliable Voiced/Unvoiced Detection

As described in 4.3, the sender in the AP/C scheme computes the auto-correlation of speech signals and uses it to estimate the pitch period of speech signals. The pitch period of speech signals, in turn, helps to determine the size of audio chunks. In the current AP/C scheme, the pitch period of speech signals is said to be found at the maximum value second to the maximum value at zero[1] of the auto-correlation. This algorithm works well for voiced sounds due to its periodic property. However, it works rather poorly for unvoiced sounds because unvoiced sounds are similar to random signals and do not actually have a periodic property (and thus a pitch period). In this case, the position of the auto-correlation's maximum value second to the maximum value at zero of the auto-correlation (and thus the size of audio chunks according to the AP/C scheme) is a random value.

As mentioned in chapter 3, unvoiced sounds are less important to the speech quality than voiced sounds and should be transmitted in large-size packets to obtain low relative overhead for the protocol header. However, due to the rather poor performance of AP/C's algorithm for determining chunk size when applied to unvoiced sounds, the sender do not always send unvoiced sounds in large-size packets and thus incurs more overhead of protocol header than necessary.

In this section, we present some improvements that help the sender to detect unvoiced sounds or silent segments. Unvoiced segments are transmitted in large-size packets and silent segments are dropped (or at least sent in large-size packets) to obtain a lower overhead of

[1] Logically, the absolute maximum value of the auto-correlation is found at 0 because a signal without any shift is most similar to itself.

protocol header and higher efficiency. The idea behind our improvements is to make the voiced/unvoiced and silence detection more reliable. If a speech segment is classified as unvoiced, an audio chunk of the length T_{max} is taken to reduce the overhead of protocol header. A similar idea is proposed in [Rabi78] to recognize unvoiced sound segments by counting zero-crossings of speech signals in the time domain.

As mentioned in chapter 3, unvoiced sounds are similar to random signals and have a broad spectrum in the frequency domain. Due to this property, adjacent samples of unvoiced sounds are, to a large extent, uncorrelated. Thus, the auto-correlation function of unvoiced sounds are typically has more zero-crossings and extrema than that of voiced sounds. In our work, an extremum of the speech signals' auto-correlation function $r_{xx}(k)$ is said to be found at position n if

$$r_{xx}(n) > r_{xx}(n+1) \text{ and } r_{xx}(n) > r_{xx}(n-1)$$

or

$$r_{xx}(n) < r_{xx}(n+1) \text{ and } r_{xx}(n) < r_{xx}(n-1)$$

Figure 11 and 12 present the auto-correlation of a voiced and an unvoiced sound.

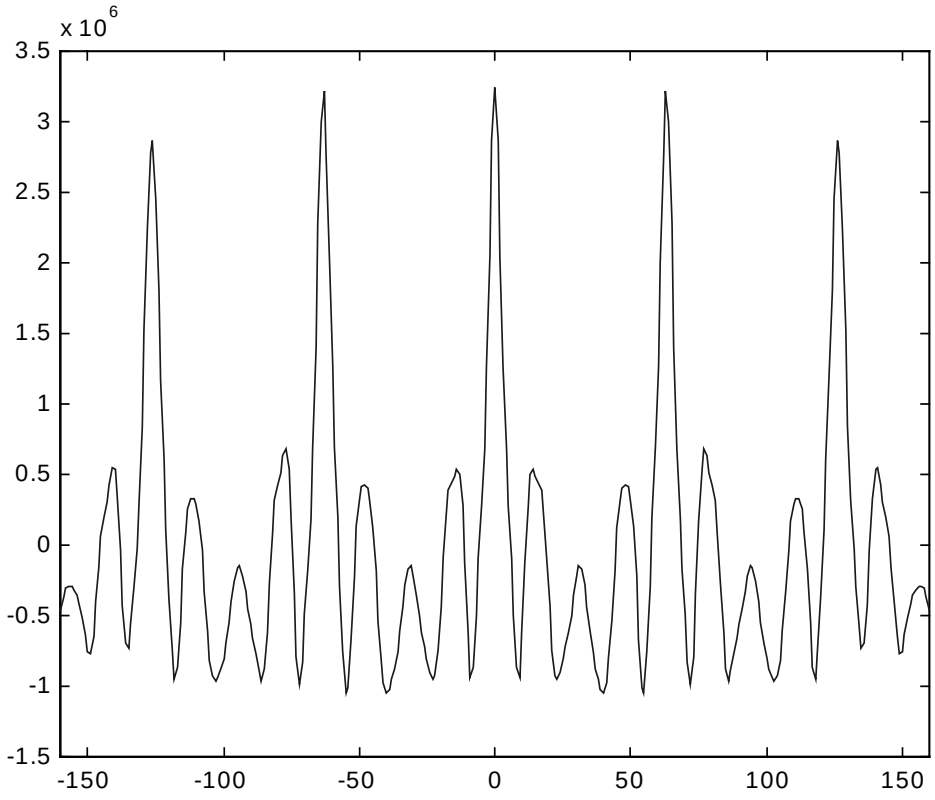


Figure 11. Auto-correlation of a voiced sound segment.

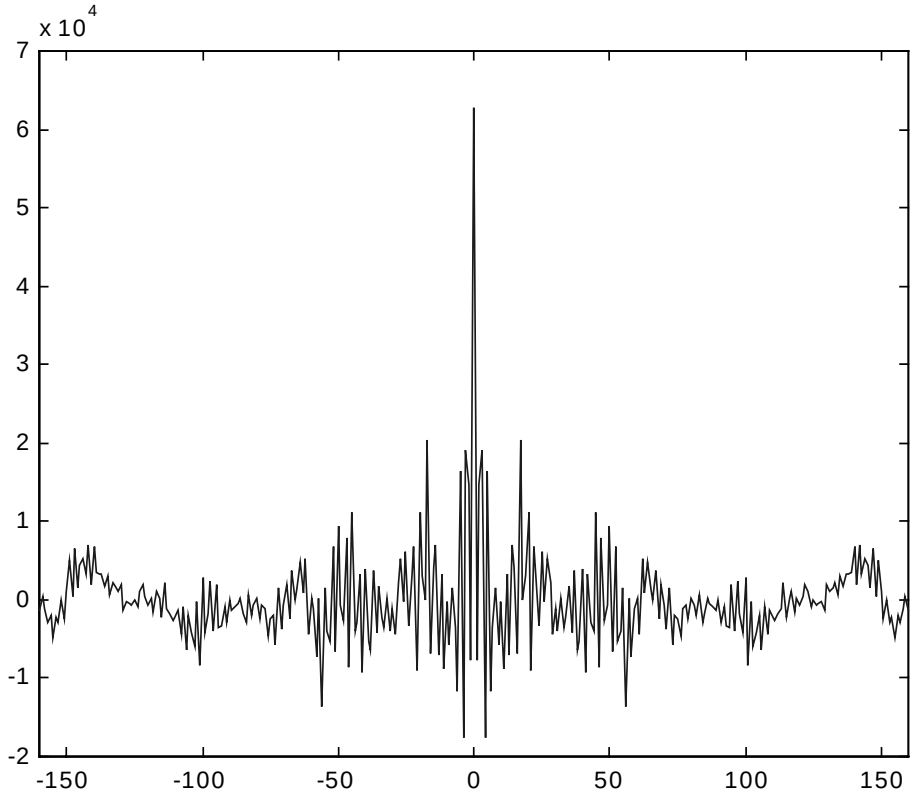


Figure 12. Auto-correlation of an unvoiced sound segment.

Let S_n and Z_n be the number of extrema and zero-crossings of the speech signal's auto-correlation[1] in the interval [0,n]. We typically have:

$S_{160} \leq 110$ for voiced sounds.
$S_{160} > 50$ for unvoiced sounds.
$Z_{30} \leq 10$ and $Z_{160} \leq 55$ for voiced sounds.
$Z_{30} > 5$ and $Z_{160} > 40$ for unvoiced sounds.

We can see that there is an overlap that makes an unequivocal voiced/unvoiced decision impossible. However, we can be sure to have an unvoiced sound if:

$S_{160} > 110$ or $Z_{30} > 10$ or $Z_{160} > 55$.

Besides, we also note that voiced sounds have a periodic structure in the time domain. Thus, the ratio of the auto-correlation's second maximum value (where the pitch period is found) to the auto-correlation's maximum value at zero is typically higher for voiced sounds than for unvoiced sounds. We typically have:

$k > 0.2$ for voiced sounds and $k \leq 0.45$ for unvoiced sounds.

where k is defined as the ratio of auto-correlation's second maximum value (where the pitch period *p* for voiced sounds is found) to the auto-correlation's maximum value at zero:

[1] Auto-correlation is computed over a speech segment of the length $3 \cdot T_{max}$, where $T_{max}=160$.

$$k = \frac{r(p)}{r(0)}$$

Again, there is an overlap that makes a clear voiced/unvoiced decision impossible. However, we can be sure to have an unvoiced sound if k < 0.2. A similar idea is proposed in [Rabi78] to detect unvoiced sounds by computing the auto-correlation of the center clipped speech signals.

4.6.2 Introduction of a Time Offset to Compute the Auto-Correlation

In the current AP/C scheme, the auto-correlation of speech signals is derived from T_{max} current and $2 \cdot T_{max}$ future audio speech samples. The large amount of look-ahead speech samples introduces additional buffer time and also leads to a wrong voiced/unvoiced decision at unvoiced→voiced transition. In this case, the future voiced samples dominate[1] over the current unvoiced samples, resulting in the current unvoiced segment being classified as voiced.

We introduce a time offset to compute the auto-correlation of speech signals. The time offset avoids the above problem and also reduces the buffer time. In our work, the auto-correlation is derived from T_{offset} past, T_{max} current, and $2 \cdot T_{max} - T_{offset}$ future speech samples. A similar approach is taken by the G.729 encoder: The parameters for a frame are determined by analyzing 120 past, 80 current, and 40 future samples.

The experiments carried in our research have shown that

$$T_{offset} = \frac{7}{8} \cdot T_{max}$$

is an appropriate value for T_{offset}.

At an unvoiced→voiced transition, a too large value of T_{offset} would classify the first voiced samples as unvoiced and a too small value of T_{offset} would classify the last unvoiced samples as voiced.

4.6.3 Extension of a Silence Detection

In AP/C, the sender algorithm should run in parallel or should be combined with the silence detector to suppress silent segments and save bandwidth. However, if the silence detector is disabled or doesn't work reliably for some reasons[2], the following simple algorithm can be performed or combined with the silence detection algorithm to drop silent segments (or at least to send them in large-size packets):

After computing the auto-correlation of speech signals, the maximum value is searched for in the range $[T_{min}, T_{max}]$. If the maximum value is found at T_{min}, we check whether it is a real extremum. If not, the position where the maximum value is found is obviously not a pitch period and the audio chunk is classified as silent.

[1] This is because voiced signals typically have a higher energy than unvoiced signals.
[2] For low signal-to-noise ratio environments, it is not easy to distinguish between weak unvoiced sounds and background noise based on the energy of speech signals [Rabi78].

Figure 13 shows the typical auto-correlation of a silent segment. In this case, the auto-correlation's maximum value in the range $[T_{min},T_{max}]$ is found at T_{min}. Because this maximum value is not an extremum, the speech segment is classified as silent.

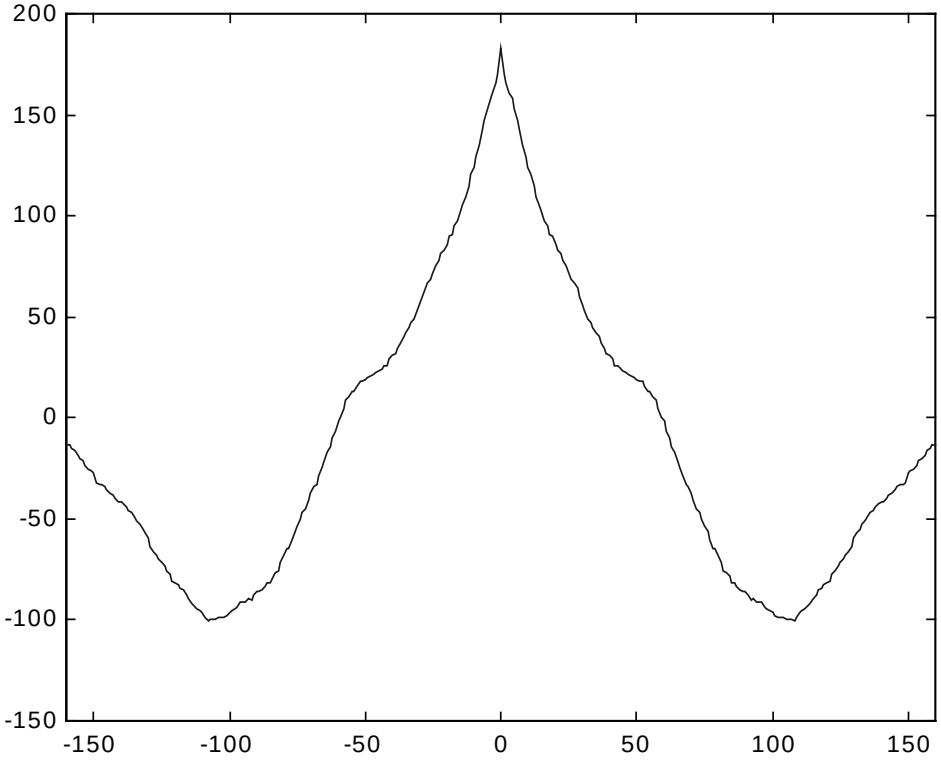


Figure 13. Auto-correlation of a silent segment.

4.6.4 Combination of AP/C and Interleaving

The current AP/C scheme assumes that isolated packet loss is predominant and packet burst loss is negligible in the Internet and in the MBone. If this assumption holds, the receivers can exploit the sender's pre-processing to reconstruct the majority of lost packets from their adjacent packets. However, if the assumption doesn't hold, the receivers cannot perform loss concealment and AP/C doesn't deliver a significantly better speech quality than other loss concealment schemes.

In our work, we develop a combination of the AP/C scheme and interleaving to cope with burst loss. This is achieved by buffering one audio chunk (c_{12} and c_{32} in Figure 14) and interleaving a block of four chunks. A block of more chunks would conceal longer burst loss but would incur additional delay.

Interleaving helps to spread the burst loss and thus enables the receivers to perform loss concealment. Figure 14 illustrates the idea of combining the AP/C scheme and interleaving. Clearly, interleaving presents an engineering trade-off: it helps the receivers to cope with packet burst loss but also causes them to suffer an additional delay[1].

[1] In AP/C, the length of an audio chunk varies from 30 to 160 PCM samples at a sampling rate of 8000 Hz, corresponding to a range from 3.75 ms to 20 ms in duration.

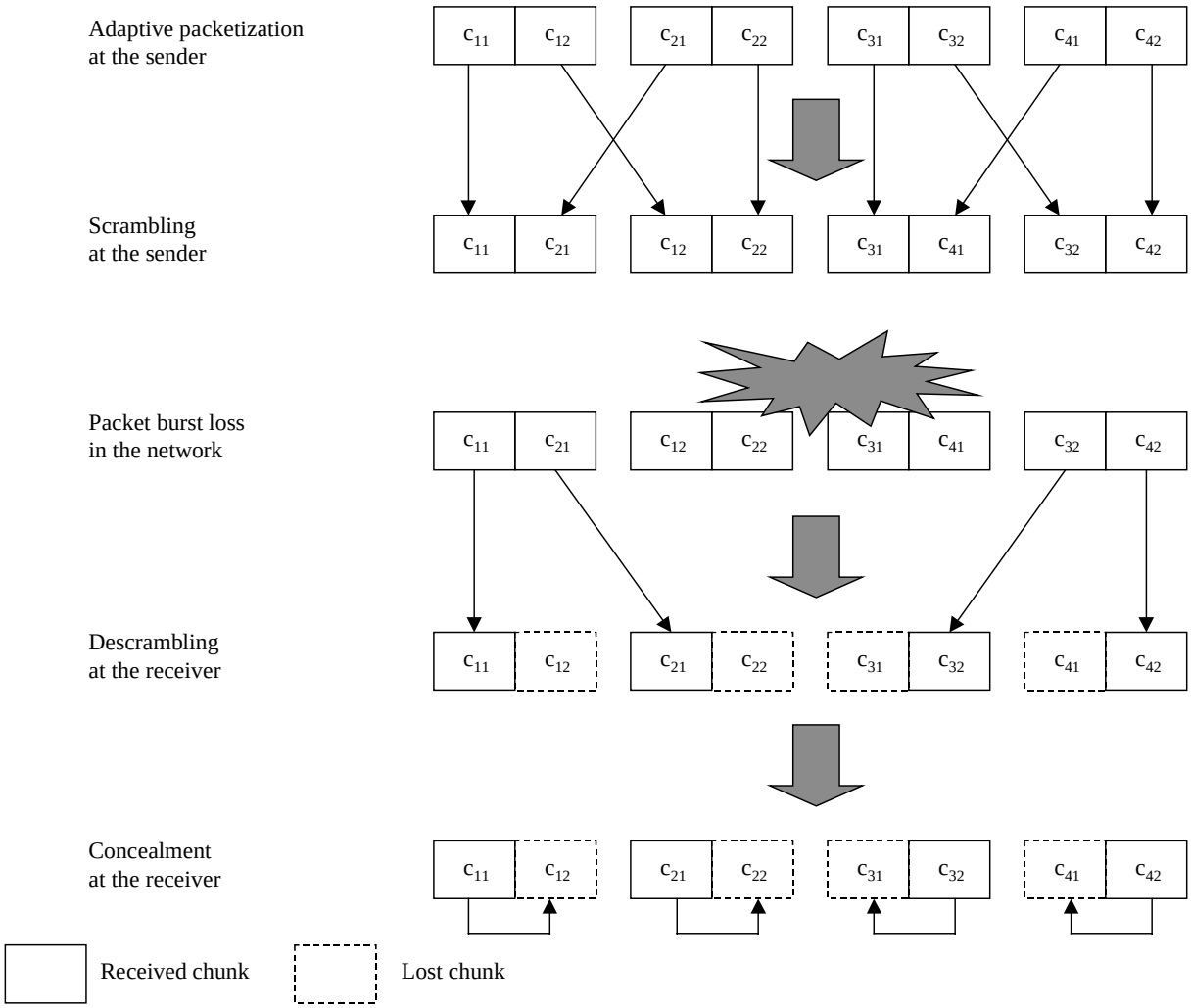


Figure 14. Combination of AP/C and interleaving.

5. Speech property-based FEC

In order to reduce bandwidth consumption in the transmission of speech signals, speech coding is employed to compress the speech signals, i.e. to use as few bits as possible to represent them. In general, speech coding techniques are divided into three categories: waveform coders, voice coders (vocoders), and hybrid coders.

Waveform coders try to directly encode speech signals in an efficient way by extracting redundancies and exploiting the temporal and/or spectral characteristics of speech signals.

Vocoders and hybrid coders attempt to model speech signals by a set of parameters and then try to efficiently encode these parameters. Vocoders and hybrid coders usually operate on a fixed size of speech frames. Hence, they are also called frame-based coders. Vocoders and hybrid coders typically operate at a lower bit rate than waveform coders at the cost of higher complexity.

In vocoders and hybrid coders, the vocal tract is modeled by a linear filter. In vocoders, the linear filter is excited with a white noise signal for unvoiced sounds and with a periodical train of pulses for voiced sounds. The period of the train of pulses is equal to the pitch period. Vocoders operate at a bit rate of around 2.4 kbps or lower and produce speech that is intelligible but not natural. Hence, they are mainly used in military applications where natural sounding is not very important.

In hybrid coders, the excitation signal for the linear filter is chosen in such a way that the perceived distortion is as small as possible. Hybrid coders deliver a better speech quality than vocoders at the cost of a higher bit rate. Hybrid coders represent a compromise of different interdependent attributes: bit rate, complexity, and buffer delay. These attributes are traded off against each other, e.g. a very low bit rate could result in a too high complexity and a too large buffer delay which are undesirable. Furthermore, hybrid coders used for speech transmission over the Internet should also be robust against loss of frames. They are promising candidates to be used for speech transmission over the Internet thanks to their relative low bit rate (ranging from 4.8 to 16 kbps) and good speech quality.

Table 1 provides some features of common waveform and hybrid coders (extracted from [Mino98], [Span94]).

The two modern frame-based coders G.723.1 and G.729 are very attractive for speech transmissions over the Internet because bandwidth saving plays an important role in choosing audio coders due to the fact that bandwidth is currently scarce and will still be scarce in the future. Their high complexity is not a great concern[1] because the power of microprocessors grows by an order of magnitude every five years [Mino98].

When attempting to modify the AP/C scheme to support frame-based coders, we face two problems:

- Due to its requirements for fine granularity of frame size, AP/C cannot be directly applied to support frame-based coders (the chunk bounds are generally different from the frame bounds, making an interoperating between AP/C and frame-based coders difficult).

[1] According to [Mino98], a 33 MHz 80486 runs at 27 MIPS and a 266 MHz Pentium II runs at 560 MIPS.

- During a loss of frames, synchronization between the encoder and decoder is lost and error propagates in the following frames until the decoder is resynchronized with the encoder [Rose97]. Thus, it is difficult to conceal the loss of frames from the output signal of a dis-synchronized decoder.

There are several possible solutions to the first problem: One solution is to determine audio chunks in such a way that the chunk bounds are also the frame bounds [Le99]. Another solution is to round up audio chunks to the next frame bound [Sann98a]. However, these solutions cannot solve the problem of synchronization. Moreover, when frames are lost, the decoder already applies its concealment algorithm using its internal state information from the last good frames. Due to the lack of this internal state information, a "concealment over concealment" does not necessarily improve the speech quality.

Coder	G.723.1 (hybrid coder) [ITU96a]	G.729[1] (hybrid coder) [ITU96b]	G.727 (waveform coder) [ITU90]
Coding scheme	Algebraic-Code-Excited Linear Prediction (ACELP) or Multipulse Maximum Likelihood Quantization (MP-MLQ)	Conjugate-Structure Algebraic-Code-Excited Linear-Prediction (CS-ACELP)	Adaptive Differential Pulse Code Modulation (ADPCM)
Bit rate (kbps)	5.3 or 6.3	8	40, 32, 24, or 16
Quality	Good	Good	Good
Complexity (MIPS[2])	14-20	20	≅2

Table 1. Bit rate, speech quality, and complexity of some common waveform and hybrid coders.

In this chapter, we will develop a new FEC scheme to support frame-based coders like G.723.1 and G.729 while keeping in mind the idea of AP/C to exploit speech properties to influence the "network" parameters. However, unlike AP/C influencing the packet sizes, we exploit the speech properties to influence the amount of redundant data while leaving the packet size constant. Objective quality measures show that our speech property-based FEC scheme achieves almost the same speech quality as current FEC schemes at a much smaller amount of redundant data.

[1] Annex A of Recommendation G.729 also describes a reduced-complexity version of G.729 that can interoperate with the G.729 coder.
[2] MIPS: Million instructions per second.

Although we only investigate the interoperating of the G.729 coder and our speech property-based FEC method, we do believe that a similar gain in speech quality can be expected when our method is applied to support the G.723.1 coder due to the similar way of working of these two frame-based coders (in particular, G.723.1 also incorporates an algorithm very similar to that of G.729 to conceal the loss of a frame using the parameters of the previous frames. See 5.1.2 for more details).

In section 5.1, we present an overview of G.729. In section 5.2, we will investigate the impact of frame loss at different positions in an audio data stream. In section 5.3, we present our speech property-based FEC method. In section 5.4, we conclude this chapter with an objective quality measures that evaluate the efficiency of our SPB-FEC scheme.

5.1 Overview of G.729

In this section, we present an overview of the G.729 coder. More detailed information on G.729 can be found in [Rose97], [ITU96b], [Mino98].

G.729 is also known as Conjugate Structure Algebraic Code Excited Linear Prediction (CS-ACELP) and operates at 8 kbps. Input data for the coder must be 16-bit linear PCM data sampled at 8000 Hz. G.729 is based on a model for human speech production. In this model, the throat and the mouth have the function of a linear filter (synthesis filter) and speech signals are produced by exciting this filter with an excitation vector.

In G.729, a speech frame is 10 ms in duration, corresponding to 80 speech samples at a sampling rate of 8000 Hz. For each frame, the G.729 encoder analyzes the input data and extracts the parameters of the Code Excited Linear Prediction (CELP) model such as linear prediction filter coefficients and excitation vectors. The approach for determining the filter coefficients and the excitation is called analysis by synthesis: The encoder searches through its parameter space, carries out the decode operation in each loop of the search, and compares the output signal of the decode operation (the synthesized signal) with the original speech signal. The parameters that produces the closest match are chosen. These parameters are encoded and then transmitted to the receivers. At the receivers, these parameters are used to reconstruct the original speech signals. The reconstructed speech signals are then filtered through a post-processing filter that reduces the perceived noise by emphasizing the spectral peaks (formants) and attenuating the spectral valleys [Mino98].

5.1.1 G.729 Encoder

For each 10-ms frame, the encoder performs a linear predictive analysis to compute the linear prediction filter coefficients. The linear predictive analysis is based on the idea that a speech sample can be approximated as a linear combination of past speech samples, i.e. the result of past speech samples being passed through a linear filter. By minimizing the sum of the squared differences between the actual speech samples and the approximated ones (over a number of speech samples), a set of filter coefficients can be found[1]. A linear filter that has that set of filter coefficients is called the analysis filter, i.e. when a speech signal is passed through it, we get the excitation for that speech signal. The synthesis filter is obtained by inverting the analysis filter. When we filter the excitation through the synthesis filter, the result is the original speech signal.

[1] The filter coefficients are the weighting coefficients in the linear combination.

For the sake of stability[1] and efficiency, the linear-prediction filter coefficients are not directly quantized but are transformed into line spectral pairs and quantized using predictive two-stage vector quantization[2].

The excitation for the speech signal is computed per 5-ms subframe (corresponding to 40 speech samples at a sampling rate of 8000 Hz) and has two components: fixed and adaptive-codebook.

Firstly, an open loop pitch delay is estimated once per 10-ms frame. This estimation is based on the auto-correlation of the weighted speech signal that is derived from filtering the speech signal through a perceptual weighting filter[3].

The adaptive-codebook contribution models the long-term correlation of speech signals and is expressed in a closed-loop pitch delay and a gain. The closed-loop pitch delay is searched for around the open loop pitch delay by maximizing the term:

$$R(k) = \frac{\sum_{n=0}^{39} x(n) \cdot y_k(n)}{\sqrt{\sum_{n=0}^{39} y_k(n) \cdot y_k(n)}}$$

where

- k is the searched delay.
- x(n) is the target signal and is obtained by filtering the linear prediction residual signal r(n) through the combination of the quantized synthesis filter and the weighting filter.
- $y_k(n)$ is the past weighted filtered excitation at delay k.

Once the closed-loop pitch delay has been found, the adaptive-codebook gain is computed as:

$$g_p = \frac{\sum_{n=0}^{39} x(n) \cdot y(n)}{\sum_{n=0}^{39} y(n) \cdot y(n)} \quad , \quad \textit{bounded by} \quad 0 \le g_p \le 1.2$$

where

- x(n) is the target signal.
- y(n) is the weighted filtered adaptive-codebook vector.

The fixed-codebook vector and the fixed-codebook gain are searched by minimizing the mean-squared error between the weighted input speech signal and the weighted reconstructed speech signal.

The adaptive-codebook gain and the fixed-codebook gain are then jointly vector quantized using a two stage vector quantization process.

[1] A direct quantization may move some of the poles of the synthesis filter outside of the unit circle, resulting in an unstable synthesis filter.
[2] In order to save bandwidth, the encoder and decoder predict the value of the line spectral pairs via a 4th order moving average. After prediction, the difference is computed and then vector quantized.
[3] The perceptual weighting filter is based on the linear prediction filter coefficients and reflects the perceptual distortion of the reconstructed/synthesized speech signal.

5.1.2 G.729 Decoder

The G.729 decoder at the receivers extracts the following parameters from the arriving bit stream: the line spectral pair coefficients, the two pitch delays, the two fixed-codebook vectors, and the two sets of adaptive- and fixed-codebook gains. The linear spectral pair coefficients are interpolated and transformed back to the linear prediction filter coefficients for each subframe. Then, for each subframe the following operations are performed:

- The excitation is the sum of the adaptive- and fixed-codebook vectors multiplied by their respective gains.

- The speech signal is obtained by passing the excitation through the linear prediction synthesis filter.

- The reconstructed speech signal is filtered through a post-processing filter that incorporates an adaptive postfilter based on the long-term and short-term synthesis filter, followed by a high-pass filter and scaling operation. These operations reduce the perceived distortion and enhance the speech quality of the reconstructed/synthesized speech signals.

5.1.2.1 Concealment of Frame Losses[1]

So far, we have glossed over the fact that a frame might be corrupted or lost. When this occurs, the G.729 decoder uses the parameters of the previous frame to interpolate those of the lost frame and performs loss concealment to reduce the degradation of speech quality of the reconstructed speech signal. In particular, the following steps are taken:

- The linear spectral pair coefficients from the last good frame is repeated.
- The adaptive- and fixed-codebook gain are taken from the previous frame but they are damped to gradually reduce their impact.
- If the last reconstructed frame was classified as voiced, the fixed-codebook contribution is set to zero. The pitch delay is taken from the previous frame and is repeated for each following frame. If the last reconstructed frame was classified as unvoiced, the adaptive-codebook is set to zero and the fixed-codebook contribution is randomly chosen.

5.1.2.2 Error Propagation

When a frame loss occurs, the decoder cannot update its state, resulting in a divergence of encoder and decoder state. Thus, errors are not only introduced in the current frame but also in the following ones. Below is a list of state information contained in the decoder:

- The 4th order moving average predictor filter memories for the line spectral pairs.
- The past excitation.
- The fixed- and adaptive-codebook gains.
- The 10th order linear prediction synthesis filter memories.

5.2 Impact of frame loss at different positions

[1] Frame loss is also called frame erasure in G.729 terminology.

In [Rose97], Rosenberg investigated the issues of error resiliency and recovery and measured the resynchronization time of the G.729 decoder after a frame loss. He pointed out that the energy of the error signal[1] increases considerably and the Mean Opinion Score (MOS) of speech quality decreases significantly when the number of consecutive lost frames jumps from one to two, and gradually from there. He drew the conclusion that a single lost frame can be concealed well by the G.729 decoder but not more. In this section, we take a further step by attempting to answer the question: "How does the speech quality degrade and how does the error propagate when a number of consecutive voiced/unvoiced frames are lost?".

The first experiment we carried out is to measure the resynchronization time of the decoder after a number of consecutive voiced/unvoiced frames are lost. We vary the position of the frame loss to cause a number of consecutive voiced/unvoiced frames to be lost and then count the number of the following frames until the signal-to-noise ratio (SNR) exceeds a certain threshold. The SNR is computed on a frame basis and is defined as:

$$SNR_{Frame} = 10 \cdot \log_{10}\left(\frac{\sum_n x(n)^2}{\sum_n (x(n) - x'(n))^2}\right) = 10 \cdot \log_{10}\left(\frac{\sum_n x(n)^2}{\sum_n e(n)^2}\right) \qquad n \in [1, F]$$

where F is the frame length, x'(n) and x(n) are the decoded signal with and without frame loss and e(n) is the error signal (the difference between the decoded signals).

We consider 20 dB an appropriate value for the threshold[2]. That is the G.729 decoder is said to have resynchronized with the G.729 encoder after the loss of a number of frames when the energy of the error signal falls below one percent of the energy of the decoded signal without frame loss. Figure 15 shows the resynchronization time plotted against the loss position. An unvoiced→voiced (**uv**) transition occurs in the eighth frame.

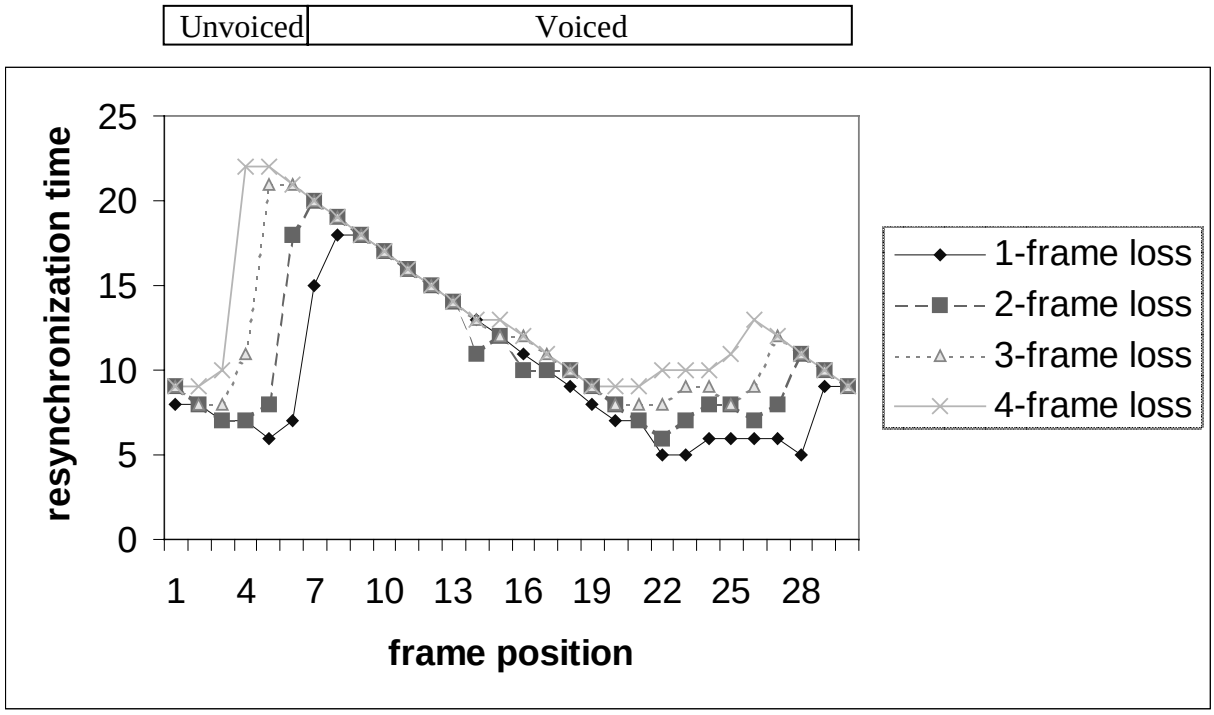


Figure 15. Resynchronization time of the G.729 decoder after the loss of a number of frames.

[1] The difference between the decoded signals with and without frame loss.
[2] This threshold is also used in [Rose97].

The second experiment we carry out is to measure the energy of the error signal over a number of frames after a number of consecutive voiced/unvoiced frames are lost. We vary the position of the frame loss to cause a number of consecutive voiced/unvoiced frames to be lost and then compute the mean SNR over a number of frames. The SNR is determined on a frame basis and then averaged over a number *N* of frames. In our experiment, we measure the mean SNR over *N=15* consecutive frames after the frame loss which we consider an appropriate value for the resynchronization time. We also measure the mean SNR over 10-20 consecutive frames after the frame loss and obtain similar results. (The mean resynchronization time of the G.729 decoder is said to vary from 7 to 10 frames with a standard deviation ranging from 6 to 9 frames depending on the burst size [Rose97]. Our first experiment also shows that the resynchronization time ranges from 5 to 22 frames depending on the position of the frame loss and the burst size. Thus, 10 and 20 are approximate lower and upper bound of *N*.).

Figure 16 shows the SNR plotted against the frame loss position. A uv transition takes place in the eighth frame.

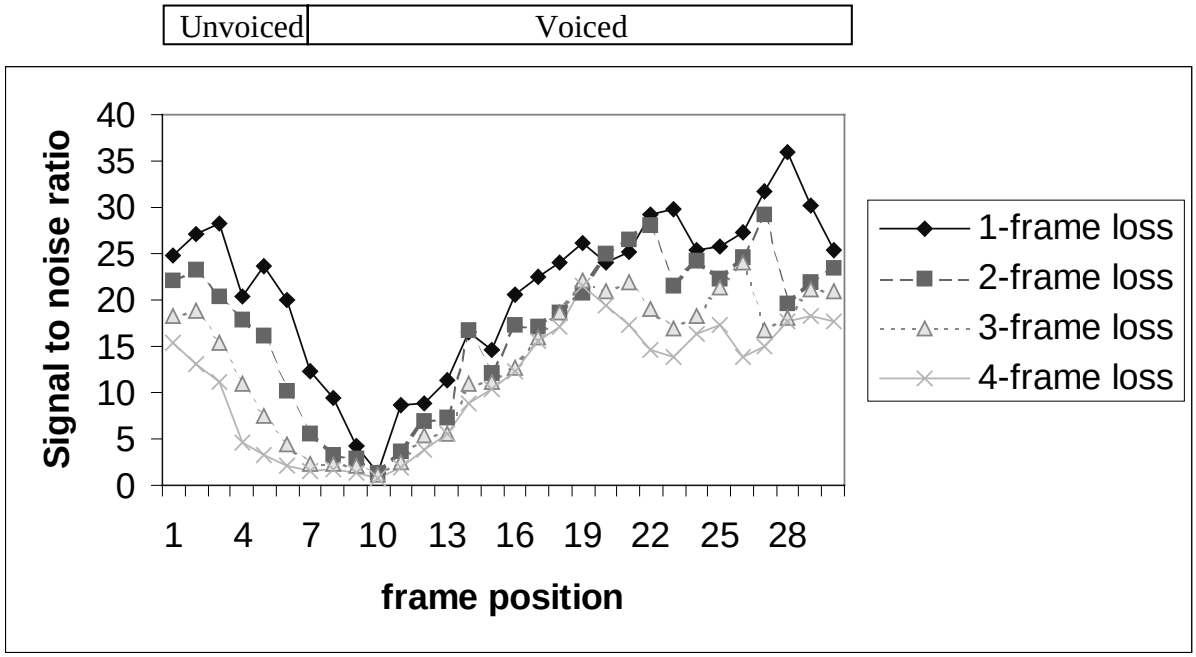


Figure 16. SNR of the G.729's decoded speech signal after the loss of a number of frames.

We can see from the Figure 15 and 16 that a loss of a consecutive number of frames at different position has different grade of impact on the speech quality of the lost and the following frames. The loss of unvoiced frames seems to have rather little impact on the speech quality and the decoder recovers the state information rather fast thereafter. In contrary, the loss of voiced frames causes a larger degradation of the speech quality and the decoder needs more time to resynchronize with the sender. Moreover, the loss of voiced frames at a uv transition leads to a significant degradation of speech quality while the loss of other voiced frames can be concealed rather well by the decoder. We repeat our two above experiments for different male and female speakers and obtain similar results.

Figure 17 presents decoded speech segments without frame loss and with frame loss at different positions. It confirms the conclusion we drew from Figure 15 and 16.

The above phenomenon could be explained as follows:

- Because voiced sounds are more important to the speech quality than unvoiced sounds, the loss of voiced frames causes a larger degradation of speech quality than that of unvoiced frames.
- Due to the periodic property of voiced sounds, the decoder can conceal the loss of voiced frames well once it has obtained sufficient information on them.
- The decoder fails to conceal the loss of voiced frames at a uv transition because it attempts to conceal the loss of voiced frames using the filter coefficients and the excitation for an unvoiced sound. Moreover, because the G.729 encoder uses a moving average filter to predict the values of the line spectral pairs and only transmits the difference between the real and predicted values, it takes much time for the decoder to resynchronize with the encoder once it has failed to build the appropriate linear prediction filter.

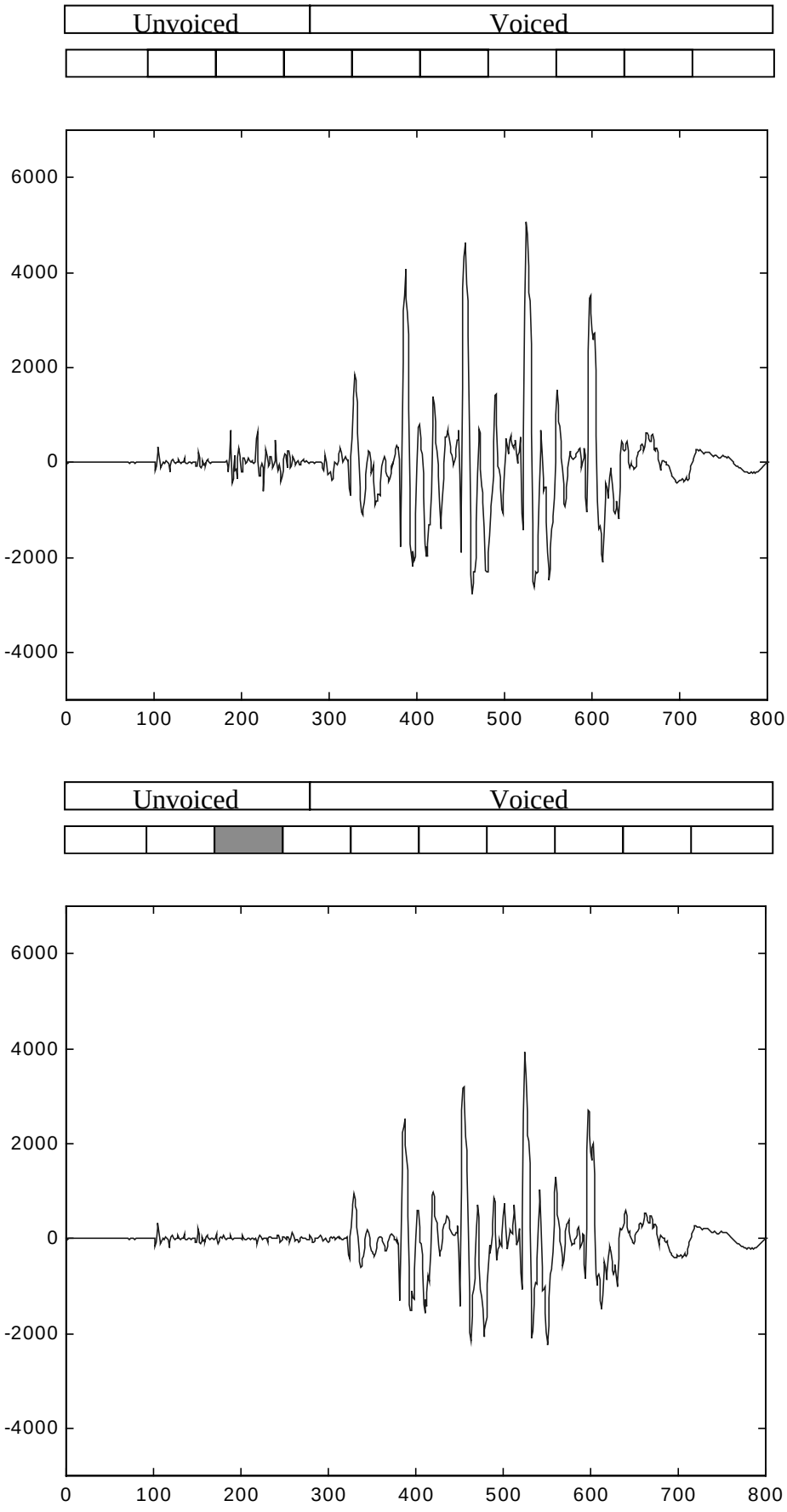


Figure 17. Decoded speech signal without and with frame loss at different position.

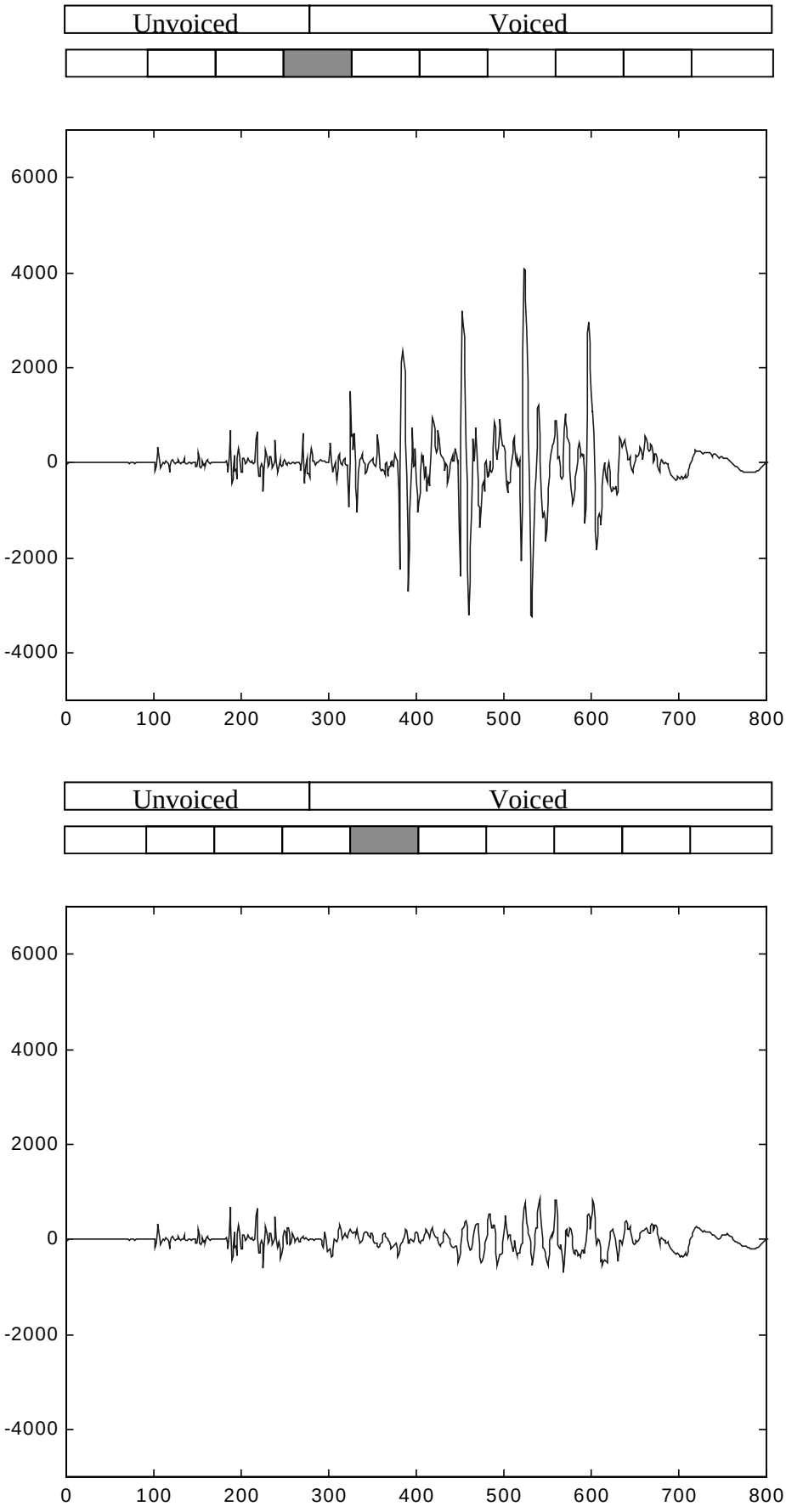


Figure 17. Decoded speech signal without and with frame loss at different position (continued).

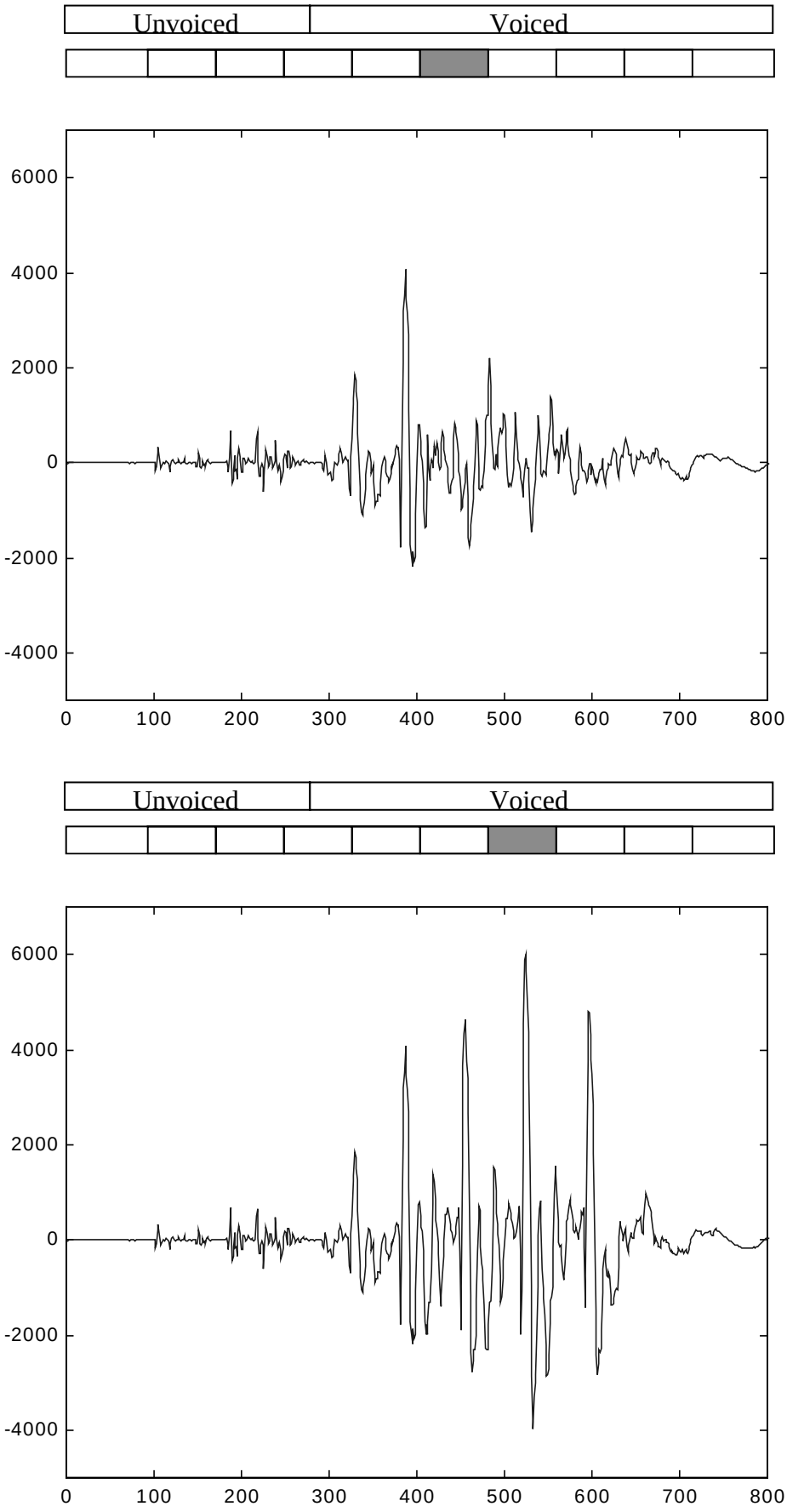


Figure 17. Decoded speech signal without and with frame loss at different position (continued).

5.3 Speech Property-Based FEC Scheme

In recent years, much research has been done to develop different FEC schemes for audio transmissions over the Internet [Bolo97], [PoRM98], [BoFT99] (just to name a few). Simulations and experiments have been carried out to find the optimal amount of redundant information and how to optimally piggy-back the redundant information of a packet (possibly coded with different audio encodings) on the following packets. However, up until now, the sender "blindly" sends the redundant information to protect audio data without knowing which audio packets are essential to the speech quality. Thus, the sender unnecessarily consumes bandwidth under light network load and aggravates the congestion in the Internet under heavy network load. Moreover, if redundant data of a packet is coded with different audio encodings and piggy-backed on the following packets and if a loss of an important frame occurs, all decoders suffer loss of synchronization and deliver decoded speech signals with bad quality [Rose96].

An example for the above idea about decoders' loss of synchronization is illustrated in Figure 18 where the sender transmits PCM *μ-law* audio data as primary data and G.729 audio data as redundant data. When a data packet arrives at the receiver, the PCM *μ-law* audio data is played and the G.729 audio data is passed to the G.729 decoder to keep it synchronized with the G.729 encoder at the sender. The output of the G.729 decoder is discarded if the previous packet is also received. If a packet is lost (e.g., packet (n) in Figure 18 containing frame (n) of *μ-law* audio data and frame (n-1) of G.729 audio data) and the following packet is received (e.g., packet (n+1) in Figure 18 containing frame (n+1) of *μ-law* audio data and frame (n) of G.729 audio data), the receiver plays frame (n) from the G.729 decoder. However, because the G.729 decoder does not receive the frame (n-1) of G.729 audio data, it suffers a loss of synchronization, resulting in a bad quality in speech signal of frame (n).

The above examples indicates that it is probably not the best strategy to equally distribute the amount of redundant data on all audio data packets to protect them using multiple encodings. This leads us to a new FEC scheme presented in this section that does not use equally distribute the redundant data and use multiple encodings but focuses the amount of redundant data on speech frames essential to the speech quality.

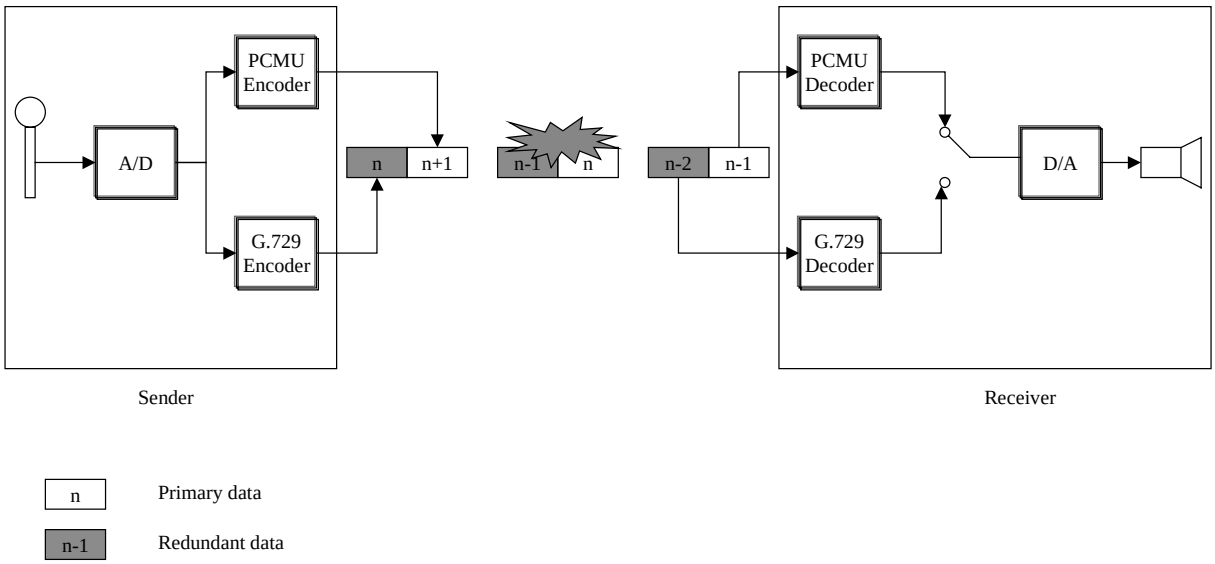


Figure 18. Decoders' loss of synchronization during a packet loss.

The experiments we carried out in section 5.2 have led us to the knowledge that the loss of frames at the beginning of a voiced signal causes a significant degradation in speech quality and a frame-based decoder can conceal the loss of other voiced frames rather well once it has obtained sufficient information on the voiced signals. The loss of unvoiced frames is also concealed well by the decoder. This knowledge is exploited to develop a new FEC scheme called speech property-based FEC (SPB-FEC). The speech property-based FEC scheme uses redundant information to protect frames essential to the speech quality and lets the decoder perform loss concealment (if other frames are lost). In contrary to other FEC schemes that equally distribute the amount of redundant data on all data packets, our FEC scheme focuses the amount of redundant data on the frames essential to the speech quality and lets the decoder's concealment algorithm do its job elsewhere.

Audio tools that use the FEC mechanisms to protect data packets are now enhanced by an analysis module similar to the one used in the AP/C scheme. This module analyzes the audio data stream and only turns on the FEC mechanisms when it detects a uv transition to protect the voiced frames at the beginning of a voiced signal.

Figure 19 illustrates the structure of audio tools with the speech property-based FEC mechanism.

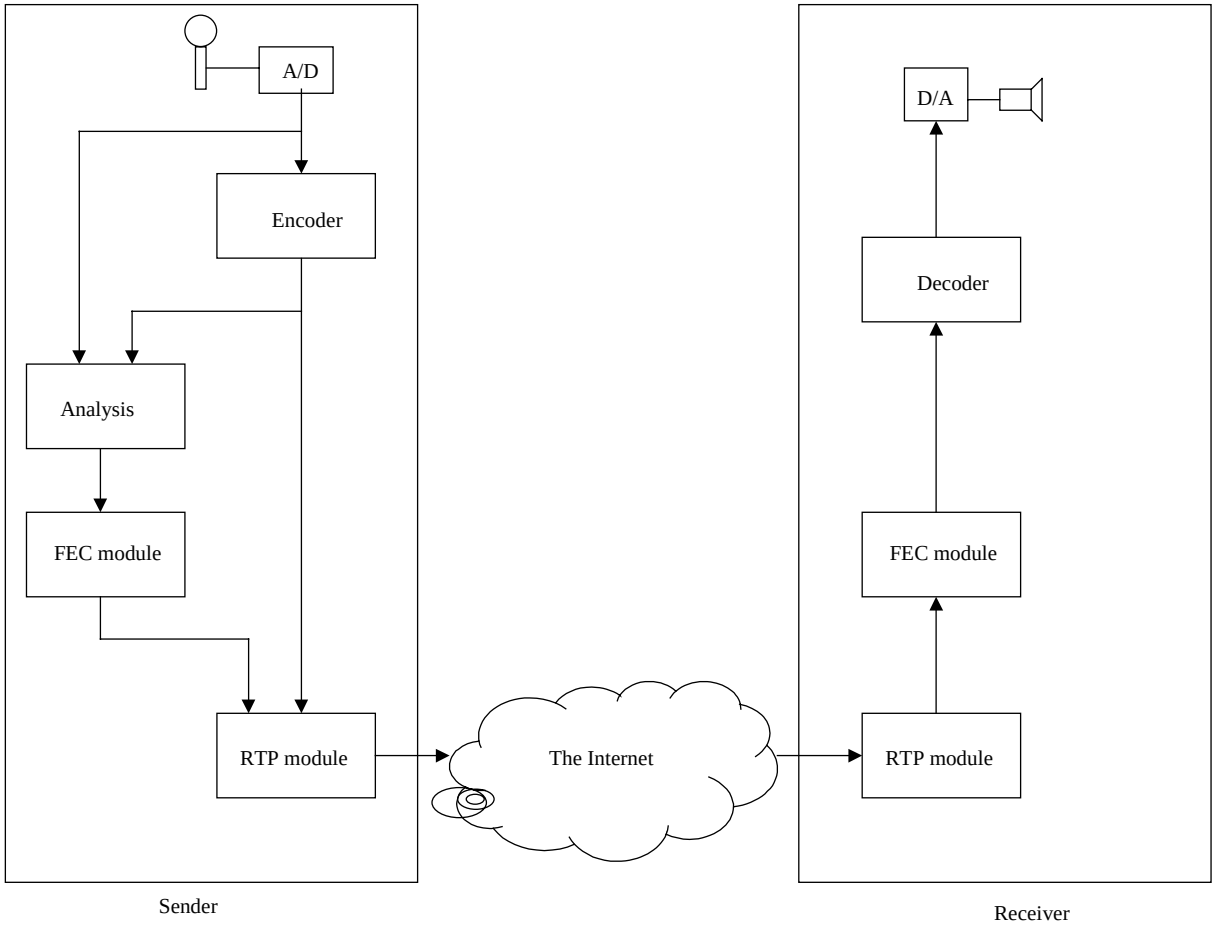


Figure 19. Structure of an audio tool with the speech property-based mechanism.

The following simple mechanism written in a pseudo-code is used to detect a uv transition and protect the voiced frames at the beginning of a voiced signal.

```
protect = 0

foreach (k frames)
        send(k frames)
        classify = analysis(k frames)
        if (protect > 0)
                if (classify == unvoiced)
                        protect = 0
                else
                        sendFEC(k frames)
                        protect = protect-k
                endif
        else
                if (classify == uv_transition)
                        sendFEC(k frames)
                        protect = N-k
                endif
        endif
endfor
```

In the above algorithm, the procedure *analysis()* is used to classify a block of *k* frames as voiced, unvoiced, or *uv* transition[1]. The procedure *send()* and *sendFEC()* are used to send a block of *k* frames and redundant information to protect these frames. *N* is a pre-defined value[2] and defines how many frames at the beginning of a voiced signal are to be protected. Our simulations show that the range from 6 to 16 are appropriate values for *N* (depending on the network loss condition).

We deliberately do not specify a more detailed algorithm here to leave room for applications and improvements in our future work. In particular:

- The number of frames transmitted in a packet (*k* in the above algorithm) is not specified. In our experiments presented in section 5.4, we choose k=2, a typical value for interactive speech transmissions over the Internet. Every 20 ms, an audio data packet with 40 bytes RTP/UDP/IP header and 20 bytes audio data is sent. A larger number of k would help to reduce the relative overhead of the protocol header but also increases the buffer delay and makes concealment in case of packet loss (due to large loss gap) more difficult which might be undesirable.
- Procedure *send()*: Applications can choose to interleave frames to spread burst losses. In our research, we attempt to interleave blocks of four frames[3] to spread burst losses but there is surprisingly no noticeable improvement in speech quality. This could be explained as follows: Given that the resynchronization time is rather long (5-18 frames[4]), a good frame received directly after a frame loss does not help the decoder much to recover the state information. Thus, there is no much

[1] The *vu* transition is unimportant in our algorithm and is classified as unvoiced.
[2] In future work, we plan to define *N* as a variable that is adapted to the network loss condition.
[3] A larger number of frames would introduce more buffer time and is not attractive for interactive speech transmissions over the Internet.
[4] The resynchronization time depends on whether the lost frame is classified as voiced, unvoiced, or *uv* transition.

difference or improvement between losing two frames in a row and losing one frame, receiving the next one, and then losing another one.

- Procedure *sendFEC()*: Applications can choose to send redundant information of a packet in a separate stream or to piggy-back it on the following packet(s) [PeKH97]. The first method is backward compatible but has a higher overhead of protocol header and requires an out-of-band mechanism to couple the audio stream with its associated stream of redundant information. The second method has a lower overhead of protocol header but is not necessarily backward compatible. For simplicity, we choose the second method and piggy-back the content of packet (n) (if it is an important packet) on packet (n+2) to reduce the impact of burst losses of packets. Clearly, the receivers in our solution suffer an additional delay in case of loss of important packets. In this case, we trade off lower loss rate of important frames against an additional delay. In future work, we plan to couple the sendFEC() procedure with the network loss condition like [BoFT99]. In this case, the sender receives feedback information on the network loss condition from the receivers and uses this piece of information to determine the amount of redundant data and how to piggy-back it on the following data packets.
- Procedure *analysis()*: Senders can choose to run a parallel algorithm for voiced/unvoiced decision or to couple this algorithm with the encoder's operation. The first method is a general approach but usually consumes more CPU resource. The second method is coder-specific but requires less CPU resources. For simplicity, we choose the second method. Unfortunately, the voiced/unvoiced decision in G.729 is made in the decoder so that the sender also has to run a decoder to decode its own frames and detect voiced/unvoiced transition.

5.4 Evaluation of the Speech Property-Based FEC Scheme

5.4.1 Introduction to Quality Measures of Speech Signals

After our speech property-based FEC scheme has been developed, we carry out simulation to evaluate its speech quality-based efficiency. In general, there are two ways to measure the speech quality: subjective and objective test.

- In subjective tests, listeners listen to a set of speech signals without being told about their nature. The listeners are asked to evaluate the quality of the speech signals by giving a score. The score typically ranges from 1 for bad to 5 for excellent quality. The listeners' score is averaged, resulting in a Mean Opinion Score (MOS) that represents the speech quality.
- In objective tests, the speech quality is evaluated by measuring the distortion of the decoded speech signals compared to the original speech signals. The most widely-used objective test method is to use the SNR to assess the speech quality. There are several ways to compute the SNR. The most common ones are: overall (or "classical") SNR method that computes the SNR over the whole signals and frame-based SNR method that computes the SNR on a frame basis and then averages the results. The frame-based SNR is said to provide a much better estimate of the subjective quality than the overall SNR [Dell93]. However, it is widely agreed that the SNR does not adequately reflects the subjective quality for frame-based coders. Hence, there has been a continuos effort to develop sophisticated objective quality measures for frame-based coders. These objective quality measures attempt to estimate the subjective quality as closely as possible

by modeling the human auditory system. They are developed to replace the time-consuming and expensive subjective tests.

Because subjective quality measures are expensive and time-consuming, we choose to use the objective quality test methods. In particular, two objective quality measures are used: the Enhanced Modified Bark Spectral Distortion (EMBSD) [YaKY99] and the Measuring Normalizing Blocks (MNB) described in the Appendix II of the ITU's Recommendation P.861 [ITU98]. These two objective quality measures are reported to have a very high correlation with subjective tests and are suitable for the evaluation of speech degraded by transmission errors in real network environments such as bit errors and frame erasures[1] [YaKY99].

5.4.2 Simulation Overview

In our simulation, we use a simple network model to drop audio data packets. The idea for the simple simulated network to drop data packets is taken from [BoFT99], [SaCa99]. The network is simulated by a Gilbert model that has two states reflecting whether the previous packet is received (state 0) or lost (state 1).

Let p be the probability for the network model to drop a packet given that the previous packet is delivered, i.e. the probability for the network model to go from state 0 to state 1. Let q denote the probability for the network model to drop a packet given that the previous packet is dropped, i.e. the probability for the network model to stay in state 1. This probability is also known as the *conditional loss probability (clp)*. Let p_0 and p_1 denote the probability of the network model to be in state 0 and state 1, we have:

$$p_1 = p_0 \cdot p + p_1 \cdot q$$
$$p_0 + p_1 = 1$$
$$\Rightarrow p_0 = \frac{1-q}{p+1-q} \quad , \quad p_1 = \frac{p}{p+1-q}$$

The probability for a packet to be dropped regardless whether the previous packet is delivered or dropped, i.e. the *unconditional loss probability (ulp)*, is exactly the probability for the network model to be in state 1 (p_1). Figure 20 presents the Gilbert model with its transition probability.

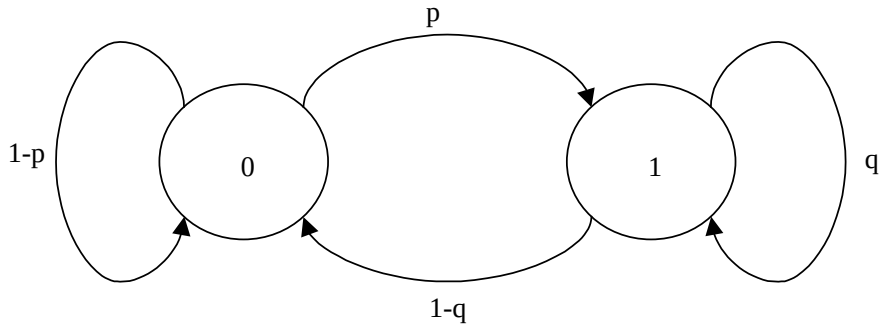


Figure 20. Gilbert model.

[1] Unfortunately, insufficient information is available to us whether these objective quality measures are suitable for packet loss.

Our simulation is schemed as follows:

- At first, a network is simulated to drop audio data packets containing two frames, i.e. 20 ms speech segments, without redundant information. The audio data stream with frame loss is decoded. Then, the decoded speech signals with and without frame loss are fed into the objective quality measures to evaluate the speech quality. This step gives speech processing non-experts an impression on the objective quality measures' reliability and the meaning of the objective quality measurements' results.
- In the second step, audio data packets with our speech property-based FEC scheme (SPB-FEC), with two other FEC schemes, and a scheme without redundant data are applied to the simulated network to drop audio data packets. Each audio data packet contains two frames, i.e. 20 ms speech segments, and possibly some redundant data depending on the respective FEC scheme. The FEC schemes are then used to recover some of the lost frames. The audio data streams (possibly still with some frame losses) are decoded. These decoded speech signals and the decoded speech signal without frame loss are applied to the objective quality measures to evaluate the speech quality-based efficiency of the FEC schemes.

The two simulation steps for the evaluation of the FEC schemes are illustrated in Figure 21.

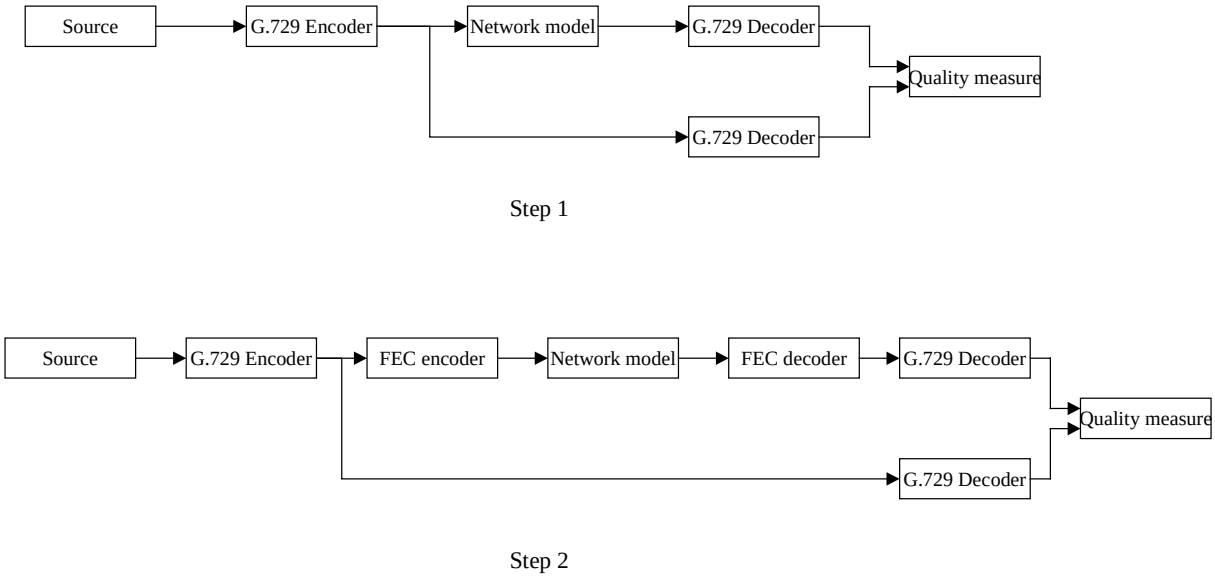


Figure 21. Simulation steps for the evaluation of the FEC schemes.

Besides our speech property-based FEC scheme, we also apply two other FEC schemes to the above simulation and use the speech quality of their decoded speech signals as reference to evaluate our speech property-based FEC scheme.

- In the first FEC scheme, the two frames of the packet (n) are piggy-backed on the packet (n+2) (we do not piggy-back the two frames of the packet (n) on the packet (n+1) to avoid the effect of packet burst loss). This FEC scheme has a redundancy overhead of 100%.
- In the second FEC scheme, the four frames of the packet (n) and (n+1) are XORed and the result is piggy-baked on the packet (n+2). If the packet (n+2) and one of the packet (n) or (n+1) arrive at the receiver, the lost packet can be recovered. In

turn, the four frames of the packet (n+2) and (n+3) are XORed and the result is piggy-backed in the packet (n+4). This FEC scheme has a redundancy overhead of 50%.

Our speech property-based FEC scheme is similar to the reference FEC scheme 1. However, in our scheme, only when a uv transition is detected, the FEC mechanism is turned on to protect the voiced frames at the beginning of a voiced signal, resulting in a redundancy overhead of about 41.9%.

Figure 22 illustrates the two reference FEC schemes.

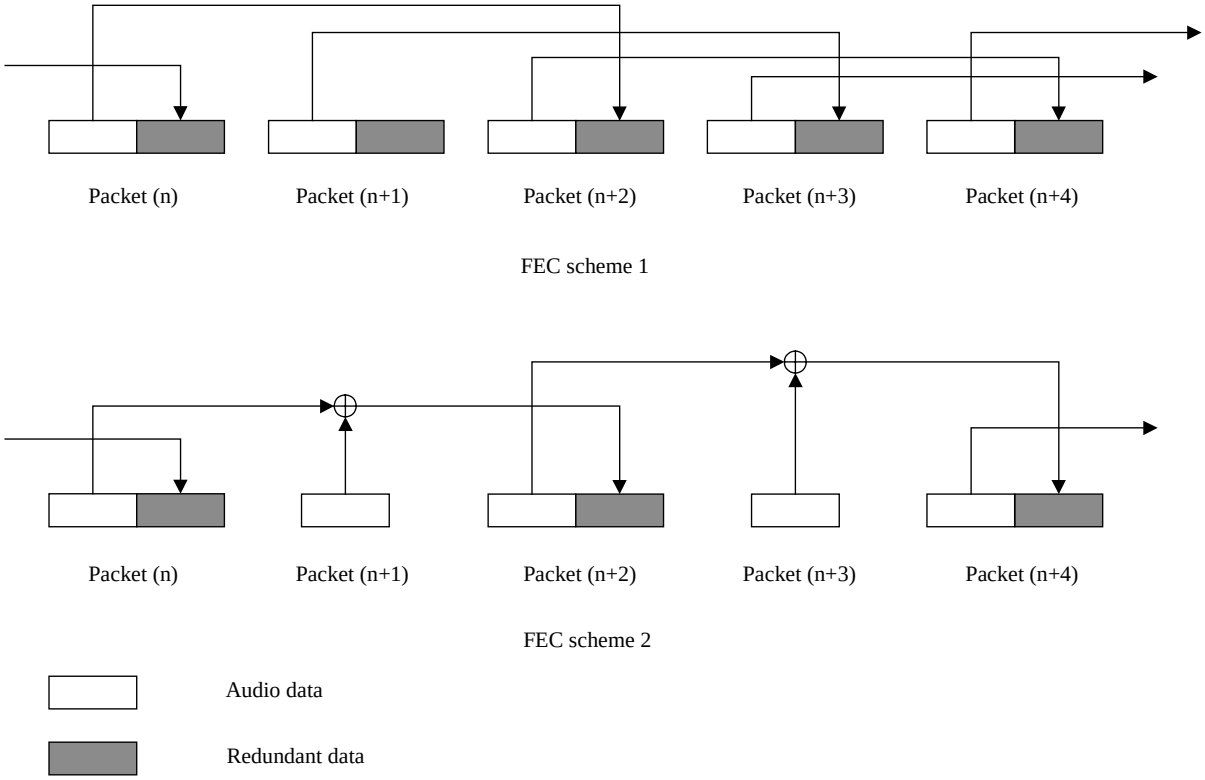


Figure 22. Two reference FEC schemes.

5.4.3 Simulation Details

The two steps of the simulation described above are now carried out. For each of these 2 steps, we vary the two parameters *p* and *q* of the network model to simulate different network loss conditions. For each pair of *p* and *q*, we apply the same speech sample containing male and female voices to our simulation but use different seeds for the random process to have different loss patterns. This is rather important because different loss patterns can have different levels of impact on the speech quality, e.g. a loss pattern dropping only voiced frames results in a worse speech quality than a loss pattern losing only unvoiced frames. By averaging the result of the objective quality measure for several loss patterns, we have a reliable indication for the performance of the G.729 coder and the FEC schemes under a certain network loss condition.

In the first simulation step, we vary *p* and *q* in constant steps to obtain an impression on the reliability and the meaning of objective quality measurement results. Table 2 shows the

network loss rate (unconditional loss probability) associated with the pairs of *p* and *q* in the first simulation step. In the second simulation step, we assign to *p* and *q* approximated values reflecting real network loss conditions measured in the Internet [Bolo93]. Table 3 and 4 show the network loss rate (unconditional loss probability) and the application loss rate, i.e. the loss rate seen by the G.729 decoder after FEC decode has been performed, associated the with the pairs of *p* and *q* in the second step of the simulation.

Figure 23 shows the application loss rate of different FEC schemes and pairs of *p* and *q*.

	p=0.1	p=0.2	p=0.3	p=0.4	p=0.5
q=0.1	0.1	0.1818	0.25	0.3077	0.3571
q=0.2	0.1111	0.2	0.2727	0.3333	0.3846
q=0.3	0.125	0.2222	0.3	0.3636	0.4167
q=0.4	0.1429	0.25	0.3333	0.4	0.4545
q=0.5	0.1667	0.2857	0.375	0.4444	0.5

Table 2. Network loss rate (unconditional loss probability) in simulation step 1.

Network loss condition 1	Network loss condition 2	Network loss condition 3	Network loss condition 4	Network loss condition 5
p=0.05, q=0.2	p=0.1, q=0.3	p=0.15, q=0.4	p=0.2, q=0.5	p=0.25, q=0.6
0.0611	0.1231	0.1974	0.2807	0.3829

Table 3. Network loss rate (unconditional loss probability) in simulation step 2.

	Network loss condition 1	Network loss condition 2	Network loss condition 3	Network loss condition 4	Network loss condition 5
	p=0.05, q=0.2	p=0.1, q=0.3	p=0.15, q=0.4	p=0.2, q=0.5	p=0.25, q=0.6
FEC scheme 1	0.0053	0.0207	0.0507	0.0999	0.1757
FEC scheme 2	0.0198	0.0561	0.1139	0.19	0.2949
SPB-FEC	0.0388	0.081	0.1367	0.2053	0.2947
No FEC	0.0611	0.1231	0.1974	0.2807	0.3829

Table 4. Application loss rate in simulation step 2.

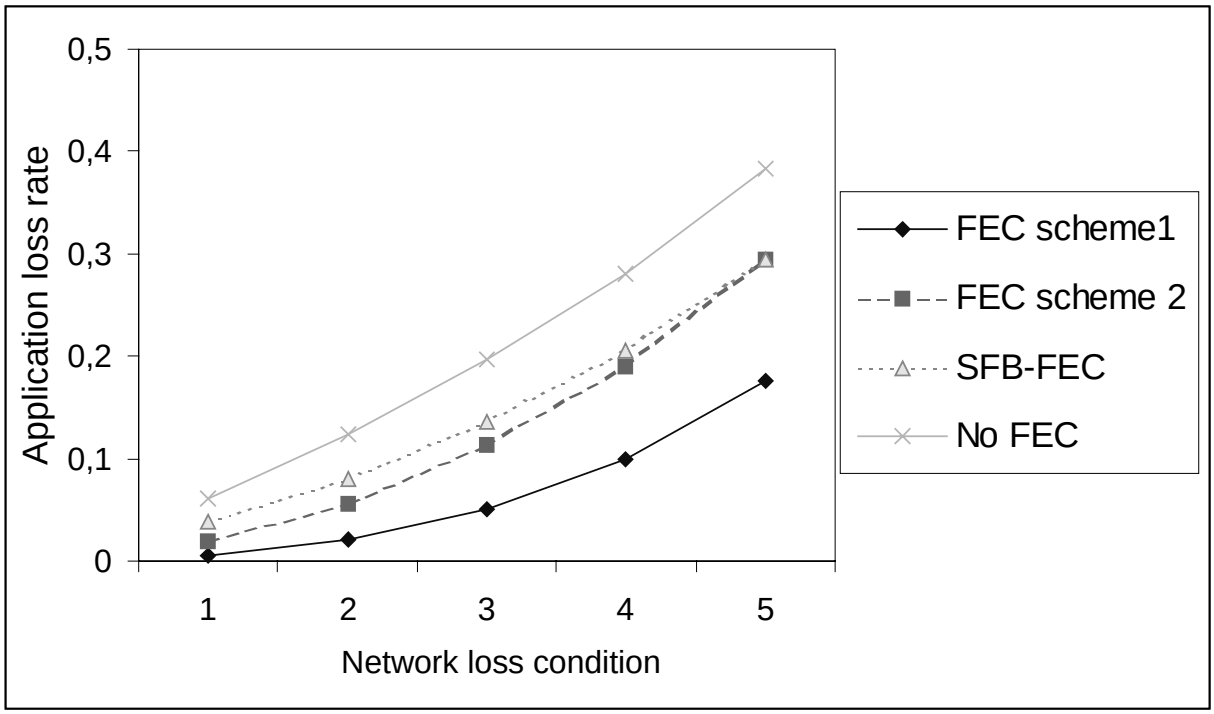


Figure 23. Application loss rate of different FEC schemes and network loss conditions.

We can clearly see from Figure 23 that the more redundant data is transmitted, the lower the application loss rate.

5.4.4 Evaluation

5.4.4.1 Evaluation Based on MNB

In MNB, the perceptual difference between the test and the reference signal is measured at different time and frequency scales. The perceptual distance, also known as Auditory Distance (AD), between the two signals is a linear combination of the measurements where the weighting factors represent the auditory attributes. The higher AD is, the more the two signals are perceptually different. Table 5 and 6 and Figure 24 and 25 show the auditory distance evaluated by MNB resulting from the simulation steps presented in 5.4.2 and 5.4.3.

	p = 0.1	p=0.2	p=0.3	p=0.4	p=0.5
q=0.1	1.3558	2.0614	2.5425	2.8926	3.1951
q=0.2	1.4068	2.1054	2.6002	2.9448	3.2351
q=0.3	1.4558	2.1658	2.6611	3.0314	3.3373
q=0.4	1.4844	2.445	2.7779	3.1073	3.4593
q=0.5	1.578	2.3495	2.9259	3.2298	3.6635

Table 5. Auditory distance from simulation step 1 evaluated by MNB.

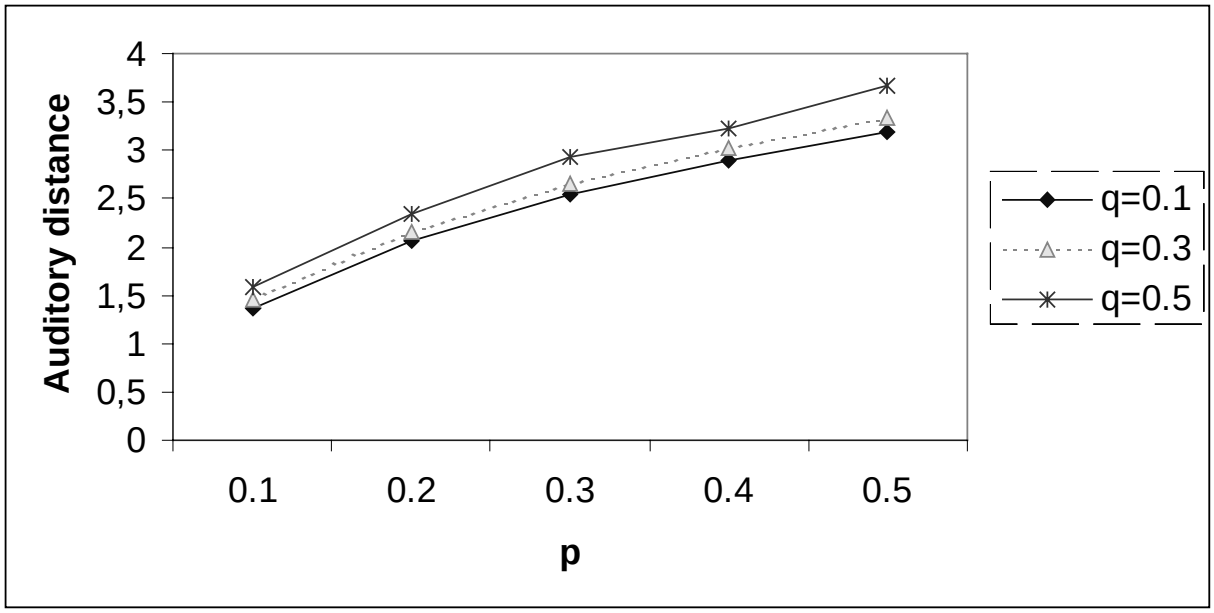


Figure 24. Auditory distance from simulation step 1 evaluated by MNB.

	Network loss condition 1	Network loss condition 2	Network loss condition 3	Network loss condition 4	Network loss condition 5
	p=0.05, q=0.2	p=0.1, q=0.3	p=0.15, q=0.4	p=0.2, q=0.5	p=0.25, q=0.6
FEC scheme 1	0.0971	0.326	0.6537	1.0883	1.5954
FEC scheme 2	0.2681	0.6632	1.1315	1.6269	2.1463
SPB-FEC	0.2005	0.4678	0.8206	1.2431	1.7543
No FEC	0.856	1.4558	1.9027	2.3495	2.8253

Table 6. Auditory distance of FEC schemes evaluated by MNB.

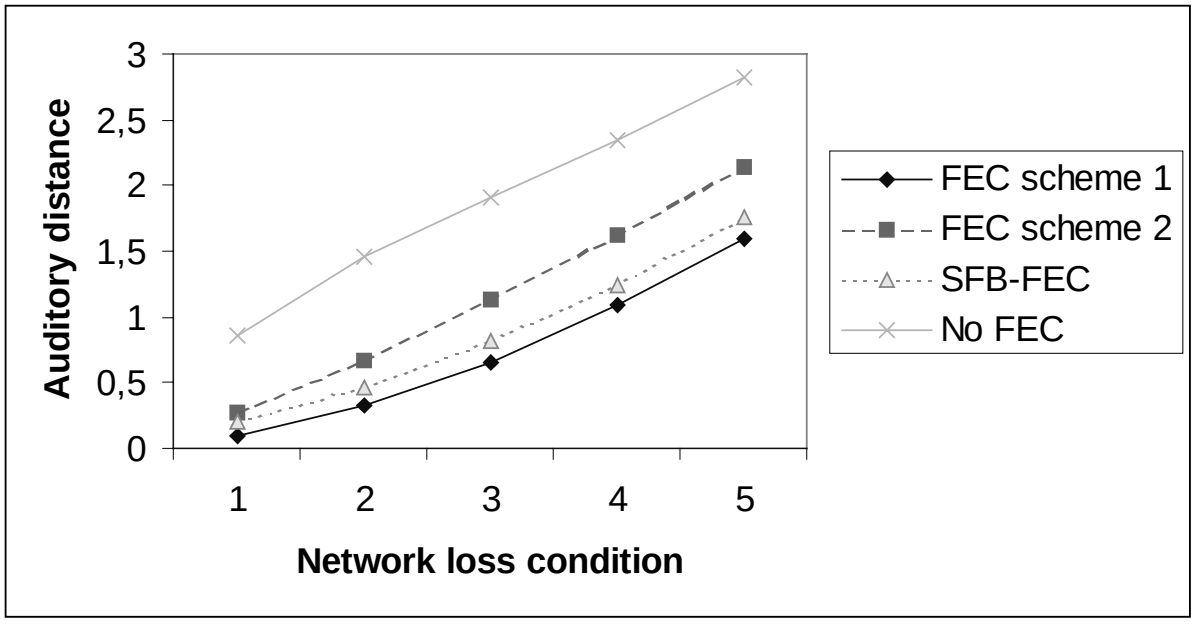


Figure 25. Auditory distance of FEC schemes evaluated by MNB.

5.4.4.2 Evaluation Based on EMBSD

In the Bark Spectral Distortion measure (BSD) [WaSG92], it is assumed that speech quality is directly related to speech loudness, which is a psychoacoustical term, defined as the perceived feeling for a given frequency and sound pressure level [YaBY98], [Novo96]. The BSD measure is the perceptual distortion computed as average squared Euclidean difference between the estimated loudness of the test and the reference signal. The MBSD [YaBY98] introduces a noise masking threshold into the BSD and only takes into account perceptual

distortion above the threshold. This noise masking threshold replaces the old one used in the BSD that is empirically derived. In the MBSD, the perceptual distortion is defined as the average difference of estimated loudnesses and not as average squared Euclidean distance of estimated loudnesses. The MBSD is enhanced by using 15 loudness components, developing a new cognition model based on postmasking effects, normalizing loudness vectors, and removing the spreading functions in noise masking threshold calculation [YaKY99]. The enhanced MBSD is also known as EMBSD. Table 7 and 8 and Figure 26 and 27 show the perceptual distortions evaluated by the EMBSD resulting from the two simulation steps presented in section 5.4.2 and 5.4.3.

	p = 0.1	p=0.2	p=0.3	p=0.4	p=0.5
q=0.1	2.2408	3.4368	3.9752	4.5975	5.1828
q=0.2	2.5255	3.8305	4.5412	5.2409	5.6317
q=0.3	2.7023	4.1026	4.866	5.5706	6.1318
q=0.4	2.9797	4.4532	5.1951	6.4097	6.9466
q=0.5	3.4707	4.7844	5.736	6.6715	7.4645

Table 7. Perceptual distortion from simulation step 1 evaluated by EMBSD.

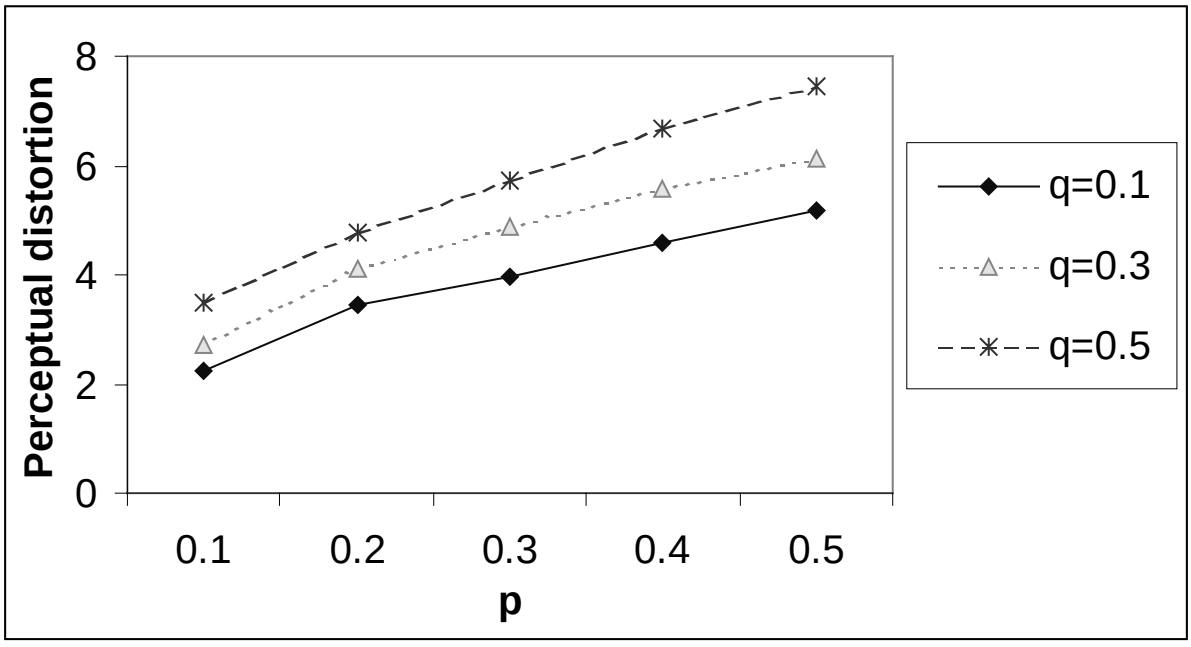


Figure 26. Perceptual distortion from simulation step 1 evaluated by EMBSD.

	Network loss condition 1	Network loss condition 2	Network loss condition 3	Network loss condition 4	Network loss condition 5
	p=0.05, q=0.2	p=0.1, q=0.3	p=0.15, q=0.4	p=0.2, q=0.5	p=0.25, q=0.6
FEC scheme 1	0.1671	0.4485	1.15	2.1256	3.3709
FEC scheme 2	0.6662	1.1302	2.4093	3.7653	5.3211
SPB-FEC	0.2449	0.6191	1.257	2.2873	3.388
No FEC	1.6346	2.7405	3.6397	4.772	6.0635

Table 8. Perceptual distortion of FEC schemes evaluated by EMBSD.

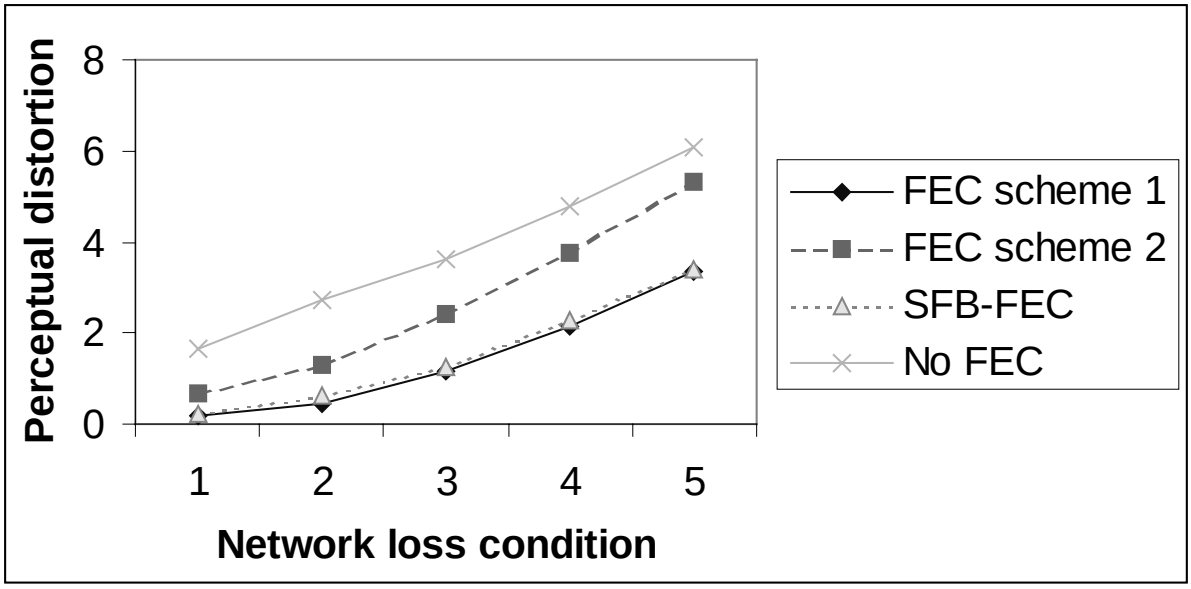


Figure 27. Perceptual distortion of FEC schemes evaluated by EMBSD.

5.4.4.3 Conclusion of Simulation

The results of MNB and EMBSD for the first simulation step (Table 5 and 7 and Figure 24 and 26) show that the higher the parameters *p* and *q* in the network model (and thus the packet loss rate) is, the higher are the auditory distance (in case of MNB) and the perceptual distortion (in case of EMBSD), i.e. the lower the speech quality of the decoded speech signals. These results indicate that the two objective quality measures are reasonably reliable and can be used for the evaluation of the FEC schemes.

The results of MNB and EMBSD for the second simulation step (Table 6 and 8 and Figure 25 and 27) show the quality of the decoded speech signals of the FEC schemes. We can see that the decoded speech signal without FEC has the highest auditory distance (in case of MNB) and the highest perceptual distortion (in case of EMBSD) and thus the worst speech quality. This is very obvious because the scheme without FEC transmits no redundant data and has the highest application loss rate. The interesting thing is that the auditory distance (in case of MNB) and the perceptual distortion (in case of EMBSD) of our SPB-FEC is significantly lower than those of the FEC scheme 2 even though it has a higher application loss rate. More interestingly, the auditory distance (in case of MNB) and the perceptual distortion (in case of EMBSD) of our SPB-FEC come very close to those of the FEC scheme 1 although its application loss rate is much higher. These results validate the strategy of our SPB-FEC scheme that does not equally distribute the amount of redundant data on all packets but concentrates it on frames that are essential to the speech quality.

6. Summary and Outlook

6.1 Summary

In this thesis, we address the problems of packet loss when transmitting speech over the Internet and make two contributions to the research topic of speech transmissions over the Internet.

- We present improvement to the AP/C scheme to help determine the pitch period (and thus the audio chunk size) more reliably. In particular, our improvement can recognize unvoiced sound and silent segments more reliably and send them in large-size packets, resulting in a lower relative overhead of the protocol header. We also develop a combination of the AP/C scheme and the interleaving method to cope with the problem of packet burst loss that is not uncommon in the Internet and in the MBone.
- We investigate the impact of frame loss at different position on the speech quality and gain the knowledge that the loss of voiced frames at the beginning of a voiced signal leads to significant degradation in speech quality while the loss of other frames are concealed rather well by the decoder's concealment algorithm. We then exploit this knowledge to develop a speech property-based FEC scheme (SPB-FEC) that protects the voiced frames essential to the speech quality while letting the decoder's algorithm do its job if other frames are lost. Simulation and objective quality measures show that our FEC scheme performs almost as good as other FEC schemes at a lower redundancy overhead.

6.2 Outlook

Although our speech property-based FEC scheme shows very promising results, there is still much room for improvement and further research. In particular:

- Although our SPB-FEC scheme helps to recover some lost packets, transmitting redundant data also adds more load to the network and thus worsens the congestion problems in the Internet. Thus an adaptive speech property-based FEC scheme like the one presented in [BoFT99] is highly desirable. In such a scheme, the sender receives feedback information on the network loss conditions from the receivers and uses this kind of information to determine the optimal amount of redundant data and how to optimally send it.
- In our current algorithm, two G.729 frames, corresponding to 20 ms, are transmitted in a packet, resulting in 20 bytes data and 40 bytes RTP/UDP/IP protocol header. Undoubtedly, a lower relative overhead of protocol header is highly desirable. Although research for header compression is currently underway [CaJa99], we ask ourselves whether it is more efficient to send more frames in a packet. Large-size packets transmitted in large intervals help to save bandwidth and also reduce the correlation of packet loss of consecutive packets when the network is congested (if the buffer queue of a router is full and a packet is dropped, a following packet transmitted in a short interval thereafter still encounters a congested router and will also be discarded [Bolo93]). However, transmitting large-size packets incurs additional buffer delay. Moreover, large-size packets are difficult to conceal due to the non-stationary properties of speech signals (in case of a packet loss) and the repair packets might arrive after the playout time (due to

the large buffer delay of large-size packets), being useless. Clearly, more research is needed to find an optimal solution[1].

- As mentioned in chapter 5, we attempt to interleave blocks of 4 frames to spread out packet burst loss but no noticeable improvement in speech quality can be heard. It is currently unclear to us whether a larger block of frames, possibly coupled with the idea to send large-size packets and/or some sophisticated scrambling algorithms (e.g. like those presented in [Chen97]), would help to improve the speech quality. Because interleaving also adds more buffer delay, we ask ourselves whether it would be a worthwhile tradeoff for speech quality.
- Our FEC scheme cannot eliminate but only reduces the possibility of losing important frames. Moreover, if no packets are lost in the Internet, all redundant data is useless. This lets us ask ourselves: "Are there other methods that protect the important voiced frames more reliably and more bandwidth-effectively?" A solution could be to develop a speech property-based queue management that gives the important voiced frames a higher priority class than other frames [SaCa99]. A similar idea is to exploit the knowledge we gain in this thesis in the Internet Integrated Service (IntServ) or Differentiated Service (DiffServ) architecture [BeBB99], [BBCD98]. That is the voiced frames essential to the speech quality are assigned to a higher service class than other frames. This idea is rather easy to implement in the DiffServ architecture due to its per-packet basis but difficult in the IntServ architecture due to its per-flow basis.
- In our current solution, we pass the frames created by the G.729 encoder to the G.729 decoder and use it to detect unvoiced/voiced transition. We find out that the G.729 decoder sometimes works unreliably, i.e. it sometimes classifies unvoiced frames as voiced and vice versa. In our speech property-based FEC scheme, an unvoiced frame classified as voiced is undesirable but harmless because it simply adds redundant data and thus increases redundancy overhead. However, a voiced frame classified as unvoiced might degrade the speech quality if it is lost because it is not protected by our SPB-FEC scheme. Thus, a more reliable voiced/unvoiced detection is very desirable. Besides, we also want a generic algorithm that can support not only G.729 but all frame-based coders. Although the problem of pitch estimation and voiced/unvoiced decision is partially solved, it is still, to some extent, a difficult problem where complexity, reliability, and buffer delay are traded off against each other [Span94].
- A problem related to the previous one is how to detect a transition between two voiced signals, i.e. how to detect a vowel-vowel transition. This problems seldom occurs but leads to bad speech quality if a voiced/voiced transition frame gets lost (in this case, the decoder attempts to conceal the loss of the new voiced signal using the parameters of the old voiced signal but that does not work well). Because we currently know of no solution in speech processing to this problem, we solve it by simply turning the FEC mechanism on and off in constant intervals after a uv transition to protect some voiced frames. Obviously, our current solution is not only inefficient but also unreliable.

To sum up it can be said that our solution takes a step in the right direction but there is still much room for improvement to find an optimal solution where the sender performs pre-processing of speech signals, the receiver performs loss concealment, and the network provides supporting mechanisms.

[1] This issue is party addressed in [Mino98].

References

[BBCD98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. An Architecture for Differentiated Service. Internet Request for Comments RFC 2475, December 1998.

[BCSh94] R. Braden, D. Clark, S. Shenker. Integrated Services in the Internet Architecture: an Overview. IETF Request for Comments, RFC 1633, 1994.

[BeBB99] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. E. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang, W. Weiss. A Framework for Differentiated Service. Internet Draft, draft-ietf-diffserv-framework-02.txt, February 1999.

[BoFT99] J. C. Bolot, S. Fosse-Parisis, D. Towsley. Adaptive FEC-Based Error Control for Interactive Audio in the Internet. Proceedings of Infocom 1999, New York, NY, March 1999.

[Bolo93] J. C. Bolot. Characterizing End-to-End Packet Delay and Loss in the Internet. Journal of High-Speed Networks, vol. 2, no. 3, pp. 305-323.

[Bolo95] J. C. Bolot, H. Crepin. Analysis and Control of Audio Packet Loss over Packet-Switched Networks. The fifth International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 95), Durham, New Hampshire.

[Bolo96] J. C. Bolot, A. Vega-Garcia. Control Mechanisms for Packet Audio in the Internet. Proceedings of Infocom 1996.

[Bolo97] J. C. Bolot, A. Vega-Garcia. The Case for FEC-Based Error Control for Packet Audio in the Internet. ACM Multimedia Systems, 1997.

[Brad97] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP). IETF Request for Comments, RFC 2205, September 1997.

[CaJa99] S. Casner and V. Jacobson. Compressing IP/UDP/RTP Headers for Low-Speed Serial Links. IETF Request for Comments, RFC 2508, February 1999.

[Chen97] J. H. Yao, Y. M. Chen. Experiments of Real-Time MPEG Audio over the Internet. Fujitsu Scientific and Technical Journal, vol. 33, no. 2, December 1997.

[Chen98] Y. M. Chen, H. Wang, T. Taniguchi, G. Sasaki. Robust Internet Transmission of Subband-Encoded Audio. Proceedings of International Symposium on Internet Technology (ISIT 98), Taipei, Taiwan, April 1998.

[Dege96] J. Degener, GSM 06.10 lossy speech compression. Documentation, TU Berlin, KBS, October 1996, http://kbs.cs.tu-berlin.de/~jutta/toast.html.

[Dell93] J. R. Deller, J. G. Proakis, J. H. L. Hansen. Discrete-Time Processing of Speech Signals. Maxwell Publishing Company, 1993.

[FFTW] Fastest Fourier Transform in the West. FFTW 1.3 User's Manual. Available from http://theory.lcs.mit.edu/~fftw.

[Free] Freephone.
INRIA. http://zenon.inria.fr/rodeo/fphone.

[Hand95] M. Handley, V. Hardman, M. A. Sasse, and A. Watson. Reliable Audio for Use over the Internet. Proceedings INET 95.

[Hand97] M. Handley. An Examination of MBone Performance. Technical Report, UCL and ISI, January 1997.

[ITU90] International Telecommunications Union. 5-, 4-, 3-, and 2-bits Sample Embedded Adaptive Differential Pulse Code Modulation (ADPCM). ITU-T Recommendation G.727, 1990.

[ITU96a] International Telecommunications Union. Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s. ITU-T Recommendation G.723.1, March 1996.

[ITU96b] International Telecommunications Union. Coding of Speech at 8 kbit/s Using Conjugate-Structure Algebraic-Code-Excited Linear-Prediction (CS-ACELP). ITU-T Recommendation G.729, March 1996.

[ITU98] Objective Quality Measurement of Telephone-Band (300-3400 Hz) Speech Codecs. ITU-T Recommendation P.861, February 1998.

[Le99] N. T. L. Le. Development of a Loss-Resilient Speech Transmission Method. Intermediate Report for Diploma Thesis. Department of Electrical Engineering, Technical University Berlin, January 1999.

[Mino98] Delivering Voice over IP Networks. D. Minoli, E. Minoli. John Wiley & Sons, Inc., 1998.

[MoKT98] S. B. Moon, J. Kurose, D. Towsley. Packet Audio Playout Delay Adjustments: Performance Bounds and Algorithms. ACM/Springer Multimedia Systems, 5:17-28, January 1998.

[Novo96] R. J. Novorita. Improved Mean Opinion Score Objective Prediction of Voice Coded Speech Signals. Master's thesis, Department of Electrical Engineering and Computer Science, University of Illinois at Chicago, 1996.

[Ous95] J. K. Ousterhout. Tcl and the Tk toolkit, Addision Wesley, 1994.

[PeHH98] C. Perkins, O. Hodson, V. Hardman. A Survey of Packet Loss Recovery Techniques for Streaming Audio. IEEE Network September/October 1998.

[PeKH97] C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J. C. Bolot, A. Vega-Garcia, S. Fosse-Parisis. RTP Payload for Redundant Audio Data. IETF Request for Comments, RFC 2198, September 1997.

[PoRM98] M. Podolsky, C. Romer, S. McCanne. Simulation of FEC-Based Error Control for Packet Audio on the Internet. Proceedings of Infocom 1998.

[Rabi78] L. R. Rabiner, R. W. Schafer. Digital Processing of Speech Signals. Prentice Hall, 1978.

[Ramj94] R. Ramjee, J. Kurose, D. Towsley, H. Schulzrinne. Adaptive Playout Mechanisms for Packetized Audio Applications in Wide-Area Networks. Proceedings of IEEE Infocom 1994, pp. 680-688.

[RAT] Robust Audio Tool. UCL, Department of Computer Science. http://www-mice.cs.ucl.ac.uk/mice/rat.

[Rizz97] L. Rizzo. Effective Erasure Codes for Reliable Computer Communication Protocols. ACM Computer Communication Review, April 1997.

[Rose96] J. Rosenberg. Reliability Enhancements to Nevot, Technical Report, Department of Computer Science, Columbia University, May 1996.

[Rose97] J. Rosenberg. G. 729 Error Recovery for Internet Telephony. Technical Report, May 1997.

[RuKT98] D. Rubenstein, J. Kurose, D. Towsley. Real-Time Reliable Multicast Using Proactive Forward Error Correction. Technical Report, Department of Computer Science, University of Massachusetts at Amherst.

[SaSB96] H. Sanneck, A. Stenger, K. Ben Younes, and B. Girod. A New Technique for Audio Packet Loss Concealment. Proceedings IEEE Global Internet 1996, London, England, 1996.

[Sann95] H. Sanneck. Fehlerverschleierungsverfahren für Sprachübertragung mit Paketverlust (Error Concealment for Speech Transmission with Packet Loss). Master's thesis, Department of Telecommunications, University Erlangen-Nürnberg. June 1995.

[Sann98a] H. Sanneck. Adaptive Loss Concealment for Internet Telephony Applications. Proceedings INET 98, Geneva/Switzerland, July 1998.

[Sann98b] H. Sanneck. Concealment of Lost Speech Packets Using Adaptive Packetization. Proceedings IEEE Multimedia Systems, Austin/TX, June 1998.

[SaCa99] H. Sanneck and G. Carle. A Queue Management Algorithm for Intra-Flow Service Differentiation in the "Best Effort" Internet. Technical Report. GMD Fokus, Berlin, May 1999.

[Shac90] N. Shacham and P. McKenney. Packet Recovery in High-Speed Networks Using Coding and Buffer Management. Proceedings ACM SIGCOMM '90, pages 124-131, San Francisco, CA, June 1990.

[Schu92] H. Schulzrinne. Voice Communication Across the Internet: A Network Voice Terminal. Research Report, Department of Electrical and Computer Engineering, University of Massachusetts at Amherst, July 1992.

[Schu95] H. Schulzrinne. Guide to NEVOT 3.33. GMD Fokus, Berlin, October 1995. Available from ftp://ftp.fokus.gmd.de/pub/glone/nevot/nevot-3.33.tar.gz (nevot.ps).

[Schu96] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. IETF Request for Comments, RFC 1889, January 1996.

[SiSc98] D. Sisalem, H. Schulzrinne. The Multimedia Internet Terminal (MInT). Special Issue on Multimedia of the Journal of Telecommunication System, vol. 9, number 3, 98.

[SKBD98] S. B. Moon, J. Kurose, P. Skelly, D. Towsley. Correlation of Packet Delay and Loss in the Internet. Technical Report, Department of Computer Science, University of Massachusetts at Amherst.

[Span94] A. Spanias. Speech coding: A Tutorial Review, October 1994.

[Stev94] W. R. Stevens. TCP/IP Illustrated, The Protocols, vol. 1. Addison Wesley, Reading 1994.

[VaNi89] R. Valenzuela, C. Animalu. A New Voice Packet Reconstruction Technique. Proceedings ICASSP, pages 1334-1336, May 1989.

[VAT] Visual Audio Tool.
http://www-nrg.ee.lbl.gov/vat.

[WaSG92] S. Wang, A. Sekey, and A. Gersho. An Objective Measure for Predicting Subjective Quality of Speech Coders. IEEE Journal of Selected Areas in Communications, vol. SAC-10, 1992.

[YaBY98] W. Yang, M. Benbouchta, R. Yantorno. Performance of the Modified Bark Spectral Distortion as an Objective Speech Quality Measure. ICASSP, vol. 1, Seattle 1998.

[YaKT96] M. Yajnik, J. Kurose, D. Towsley. Packet Loss Correlation in the MBone Multicast Network. Technical Report, Department of Computer Science, University of Massachusetts at Amherst, 1996.

[YaKY99] W. Yang, K. R. Krishnamachari, and R. Yantorno. Improvement of the MBSD Objective Speech Quality Measure Using TDMA Data, submitted to IEEE Speech Coding Workshop, 1999.