72

Internet Telephony: Optimizing Protocols, Packet Recovery, and Packet Size

by

Grant Ho

S.B., Massachusetts Institute of Technology (1997)

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1998
[February 1999]

Author ....
Department of Electrical Engineering and Computer Science
May 22, 1998

Certified by ....
Marion R. Baraniecki
Manager, Voiceband Processing Department, Comsat Laboratories
Thesis Supervisor

Certified by ....
David H. Staelin
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Internet Telephony: Optimizing Protocols, Packet Recovery, and Packet Size

by

Grant Ho

Submitted to the Department of Electrical Engineering and Computer Science
on May 22, 1998, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Internet Telephony, or Voice-over-the-Internet Protocol (VoIP), allows users to access the Internet like a public-switched telephone network. However, the Internet Protocol's packet-switching framework often leads to packet loss and network delay, resulting in VoIP systems suffering from degradations in speech quality and impractical interactivity. This thesis describes the design of an Internet Telephony system that maximizes speech quality and minimizes delay through protocol, packet recovery, and packet size optimizations.

Protocol optimization introduces the Real-Time Transport Protocol that combines the reliability of the Transmission Control Protocol and speed of the User Datagram Protocol for transmitting voice over the Internet. Packet recovery optimization leads to an original lost-frame recovery technique that combines three algorithms — linear interpolation, selective energy attenuation, and energy tapering — for maximizing VoIP speech quality. For randomly distributed packet loss rates up to 15%, this technique is shown to achieve up to a 20% increase in speech quality over G.723.1 error concealment by eliminating unnatural-sounding speech, metallic-sounding artifacts, high-energy spikes, and choppy synthesized speech. Based on informal listening tests, it is also shown that 66% of the tested population prefers the quality of improved lost-frame recovery over G.723.1 error concealment. Since speech packets experience both loss and delay, a frame-resequencing technique is then implemented with improved lost-frame recovery at the decoder. Based on these two techniques, packet size optimization employs divide-and-conquer to determine the optimal packet and decoder buffer sizes for transmitting voice over three multi-hop networks, each simulated under randomly distributed packet loss rates up to 15%. As the number of network hops and packet loss rate are increased, it is shown that speech quality is maximized and delay is minimized by sending larger speech packets and increasing the decoder buffer size.

Overall, the major contributions of this thesis include the following: (1) a detailed treatment on TCP/IP protocols and low bit rate digital speech coding techniques, and (2) the design, implementation, and joint optimization of a lost-frame recovery and frame-resequencing technique to treat lost and delayed VoIP speech packets.

Thesis Supervisor: Marion R. Baraniecki
Title: Manager, Voiceband Processing Department, Comsat Laboratories

Thesis Supervisor: David H. Staelin
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

There are lots of people to thank, but just too little space. So first off, I'd like to thank those whose names can't fit on this page. You know who you are!

So why did I get into electrical engineering in the first place? I always thought I was going to be a doctor. Asian parent thing. But then I watched "The Firm" and wanted to be a lawyer, and when I read "The Bell Jar" I wanted to be a writer. But for the longest time I've wanted to be a concert violinist. I'd dream of owning a Strad and playing unaccompanied Bach at Carnegie Hall, or recording the Sibelius Violin Concerto with Kurt Masur and the New York Phil. Even busking in Harvard Square, clad in Levis and Birks, serenading beautiful women sipping iced capuccino under the evening moonshine seemed nice. So what am I trying to get at? I never thought I'd become an electrical engineer.

But things changed when I got to MIT. And things changed again when I got into VI-A. At first it was a peer thing. Engineering seemed pretty cool, looked promising, and overall, seemed like the right thing to do. And you know what? I like it. Since then, MIT's given me the theoretical and technical background of becoming a successful engineer, and Comsat's provided me with the experience and challenge of being one.

So as this thesis marks the latest chapter in my life as a student, many thanks to the following people who've made my years at MIT and general growing-up an unforgettable experience. First, Mom and Dad. Thanks for letting to go to MIT, to meet such wonderful people and partake in such wonderful things; I only wish you had been there to share them with me. Mom, thanks for making me all that spaghetti sauce to keep me going; you know it's my favorite food. And Dad, thanks for getting me those REA Problem Solver books when you used to walk alone to the bookstore in those cold, wintry nights. I am forever indebted to the both of you for your love and support — thank you for taking care of me the past twenty-three years.

Next, thanks to all of those people who've made my graduate VI-A experience at Comsat a truly exciting one. To my manager Marion, thanks for giving me the opportunity to work on the cutting-edge of technology and for your endless technical support and friendship. To my thesis advisor, Professor Staelin, and Mike O., thank you both for proofreading my thesis — your comments and suggestions were invaluable. To Suat for teaching me everything I needed to know about ABS speech coding and VQ, to Ming for helping me battle the stubborn SoundBlaster^{TM} card, and to Qiang for your advice on improvements to my thesis, your time patience, and unlimited guidance have all helped me immensely throughout the last six months. Thank you.

And finally, to my wonderful friends at MIT and at home, thanks for always being there for me. From Carolyn and the boys at St. Mike's, to the Shads I've grown to love, to Ronnie and Sidd on Third East, and to all my fellow EECS sufferers, thank you all for your support and friendship the past four years and beyond. To my fellow MIT musicians Julia, Andrew, Yukiko, Alan, and Peter — thank you for giving me the chance to perform with you especially when electrical engineering seemed like the last thing I wanted to do. And last but not least to Laurel, Ed, Joel, and Rich, thanks for keeping my spirits alive the past four years and for telling me that there's more important things to life than just work, work work. You have all been an important and special part of my life and the experiences we've shared together will be forever engraved in my memories.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Internet Telephony, or Voice-over-the-Internet Protocol (VoIP), effectively allows users to access the Internet like a public-switched telephone network, thereby providing a low-cost alternative to traditional telecommunications handled by long-distance carriers. Although this makes VoIP both attractive and affordable, the quality of service (QoS) offered from most present-day Internet Telephony vendors leaves much to be desired. The Internet Protocol's packet-switching framework often leads to packet loss and network delay resulting in VoIP systems suffering from degradations in speech quality and impractical interactivity. This thesis describes the design of an Internet Telephony system that maximizes speech quality and minimizes delay through protocol, packet recovery, and packet size optimizations.

Protocol optimization investigates using the Transmission Control Protocol (TCP) [24] and User Datagram Protocol (UDP) [24] as target protocols for transmitting voice packets over the Internet. Although TCP is reliable and maximizes speech quality, its reliability depends on acknowledging received packets and retransmitting missing or corrupted ones; this makes TCP inappropriate for delay-sensitive data such as real-time voice. Similarly, although UDP minimizes delay by providing a service without packet retransmission or congestion control, its unreliability leads to degradations in speech quality when packets are missing or corrupted. To derive the benefits from both protocols, the real-time transport protocol (RTP) is introduced and evaluated. Described by RFC 1889[1][21], RTP provides application-level sequencing to maximize speech quality and works on top of UDP to mini-

[1]All the official standards in the internet community are published by the Internet Engineering Standards Group (IESG) as *Request for Comments*, or *RFC*.

mize delay. The fundamentals of TCP/IP protocols and the issues of protocol optimization are presented in Chapters 3 and 4.

Packet recovery optimization leads to an original technique that further improves VoIP speech quality based on the ITU-T G.723.1 dual rate speech coder [9]. First, informal listening tests are conducted to assess the quality of G.723.1 error concealment for the following rates of randomly distributed packet loss: 1%, 3%, 6%, 10%, and 15%. As G.723.1 error concealment is shown to generate unnatural-sounding speech, metallic-sounding artifacts, high-energy spikes, and choppy synthesized speech, an improved lost-frame recovery (ILFR) technique combining three algorithms — linear interpolation, selective energy attenuation, and energy tapering — is then developed in this thesis to treat these problems.

Linear interpolation eliminates metallic-sounding artifacts by introducing a buffer at the decoder to interpolate the missing frame's speech model parameters using information from the previous and future speech frames. Selective energy attenuation eliminates high-energy spikes from the synthesized speech by attenuating a frame's signal energy to a comfortable level every time a spike is detected. Finally, energy tapering eliminates choppy speech by gradually reducing the signal energy for a recovered frame over consecutive frame erasures. Based on informal listening tests, ILFR is shown to achieve up to a 20% increase in speech quality over G.723.1 error concealment and to be preferred by 66% of the tested population. The principles of low bit rate digital speech coding and the methods and results for packet recovery optimization are described in Chapters 5 and 6.

Since voice packets experience both packet loss and network delay, a frame-resequencing technique is then implemented with ILFR at the decoder. Frame-resequencing extends the length of the decoder buffer to capture incoming frames, examines their sequence numbers, and sorts disordered frames in the correct sequence before decoding or lost-frame recovery is initiated. Based on these two techniques, the goal of packet size optimization is to determine the optimal packet size, P_{opt}, and decoder buffer size, B_{opt}, for transmitting voice packets, among one, two, three, or four frames of speech, across the Internet.

The advantages and disadvantages of varying the packet and decoder buffer sizes are summarized as follows: First, smaller packets maximize speech quality since less information is lost when packets are missing or delayed; however, the effective bandwidth decreases as an increasingly greater proportion of packet overhead is introduced into the network. On the other hand, larger packets increase bandwidth efficiency since less packet overhead is introduced into the network, but speech quality degrades as more information is lost when packets are missing or delayed. Secondly, smaller decoder buffers minimize delay but may inadequately resequence disordered speech frames during high rates of packet loss, thereby resulting in poorer speech quality. Larger buffer sizes, on the other hand, have a higher probability of correctly resequencing out-of-order frames, however, at the expense of increased end-to-end delay.

A divide-and-conquer strategy is applied to determine P_{opt} and B_{opt} over three multi-hop, terrestrial networks, each simulated under the following rates of randomly distributed packet loss: 1%, 3%, 6%, 10%, and 15%. The results from packet size optimization lead to the following monotonically increasing relationships: First, as the number of network hops is increased, speech quality is maximized and delay is minimized by transmitting larger voice packets. Secondly, as the rate of packet loss is increased, the system's QoS is again maximized by increasing the size of the decoder buffer. The methods and results for packet size optimization are explored in Chapter 7.

Chapter 2

Internet Telephony Communications

The following chapter provides an overview of Internet Telephony principles including simplex, half-duplex, and full-duplex transmission as well as the importance of speech quality and end-to-end delay when designing an Internet Telephony system. By considering the issues of packet loss, echo, and conversational dynamics, three design methods are then presented for maximizing the speech quality and minimizing the end-to-end delay for an Internet Telephony system: (1) protocol optimization, (2) packet recovery optimization, and (3) packet size optimization.

2.1 Internet Telephony Principles

The transmission of voice packets over the Internet involves two systems: (1)source encoding and (2) source decoding. Consider an Internet Telephony system as illustrated in Figure 2-1.

At the source encoding system, the microphone's analog signal is first input to the analog-to-digital converter, generating 16-bit digital values sampled at a rate of 8 kHz. The resulting 128 kb/s pulse-code modulated (PCM) waveform is then input to the speech encoder. Assuming operation of the ITU-T G.723.1 speech coder [9] at 6.3 kb/s, the encoder compresses every 30 ms of speech, or 240 samples of the digital waveform, using a technique known as Linear Predictive Coding [10, 11, 22, 27]. This generates 189-bit frames of compressed speech which are delivered to the packetizer. The packetizer groups a prede-

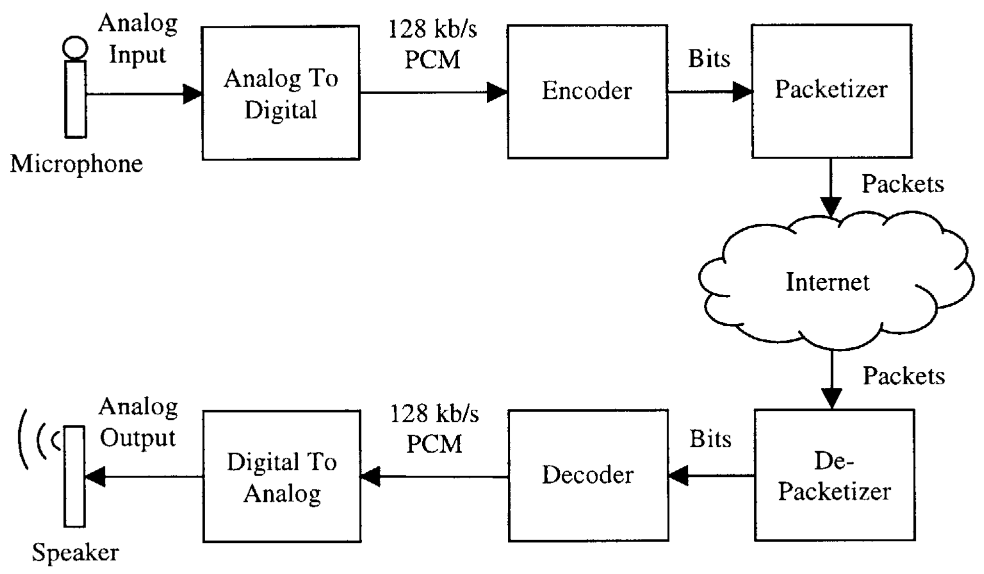

Figure 2-1: Internet Telephony System.

termined number of compressed frames into a speech packet and sends the packet over the Internet.

At the speech decoding system, the reverse process occurs. First, the de-packetizer receives the incoming speech packet from the Internet, extracts the predetermined number of compressed speech frames and sends these frames to the decoder. The decoder decodes the 189-bit compressed frames by generating 240 samples of the output 128 kb/s digital waveform for every frame. The digital waveform is then input to the digital-to-analog converter where the analog output is finally sent to the loudspeaker. Given the operation of an Internet Telephony system, the following subsections describe three modes of VoIP communication.

2.1.1 Simplex

The Internet Telephony system shown in Figure 2-1 permits the transmission of information in one direction only, from the source encoding system to the source decoding system. This is known as simplex transmission [23]. Since Internet Telephony allows users to access the Internet like a public-switched telephone network (PSTN), such a system only becomes practical if information can be exchanged in both directions.

2.1.2 Half-Duplex

Half-Duplex [23] refers to transmission in which two interconnected parties wish to exchange information alternately; for example, if one of the parties returns information in response to the other's request. Clearly, both parties must be able to switch between send and receive modes after each transmission. Although half-duplex Internet Telephony is more practical than simplex, fully interactive VoIP is not achieved in either case.

2.1.3 Full-Duplex

Duplex, or full-duplex [23], refers to transmission in which two interconnected parties wish to exchange information simultaneously in both directions, as in a telephone conversation. Since full-duplex Internet Telephony represents a fully interactive system, this thesis assumes full-duplex Internet Telephony unless stated otherwise.

2.2 Design Criteria

In designing any digital system, the final product must conform to certain requirements usually specified by organizations such as the ITU-T for PSTN and Integrated Services Digital Network (ISDN) systems or by national or private governing bodies in the case of mobile and private applications [22]. Many design criteria can be applied in defining these requirements, but since the design capacity of a digital system is often dictated by the target application, only a select number of criteria can be usefully applied when developing practical systems.

The primary objective when designing an interactive system is to maximize the system's overall quality of service (QoS), a subjective assessment that is only determined by the user. For Internet Telephony, the QoS involves two important design criteria: (1) speech quality and (2) end-to-end delay. Specifically, maximizing the QoS of Internet Telephony requires maximizing the speech quality and minimizing the end-to-end delay. Each of these design criteria is described in the following subsections. It must also be noted that robustness and complexity are additional design criteria to consider when designing an Internet Telephony system. However, since robustness directly affects speech quality under different channel conditions and complexity affects end-to-end delay, this thesis thus considers robustness and complexity as subsets of speech quality and end-to-end delay respectively.

2.2.1 Speech Quality

Speech quality is one of the most important design criteria for maximizing the QoS of Internet Telephony. The assessment of speech quality is based upon subjective listening evaluations and the results are measured in terms of a mean opinion score (MOS) [22]. Based on the MOS, speech quality is then graded as either toll (good-excellent quality), communications (fair-good quality), or synthetic (poor-fair quality). Toll quality ranges from 16 kb/s Low-Delay Code-Excited Linear Prediction (LD-CELP) to 64 kb/s PCM; these coders are traditionally used over PSTN networks. Communications quality ranges from 6.4 kb/s Improved Multi-Band Excitation (IMBE) to 13 kb/s Regular Pulse-Excited Long Term Prediction (RPE-LTP); these coders are generally suited for mobile and cellular communications services and storage-mail applications. Finally, synthetic quality ranges from 2.4 kb/s LPC10e to 4.8 kb/s CELP; these coders are primarily targeted for secure voice applications [10, 11, 22]. Clearly, a decrease in the bit rate of the speech coder leads to a degradation in speech quality.

Over non-congested networks, VoIP speech quality is equivalent to that generated by the voice compression algorithm. However, over congested networks, the quality of delivered speech increasingly degrades as packet loss and network delay increase. Lost-frame recovery strategies must thus be implemented at the decoder to successfully recover missing or delayed packets in order to maintain an acceptable degree of speech quality for the user.

2.2.2 End-to-End Delay

The other important design criteria for maximizing the QoS of Internet Telephony is end-to-end delay, τ_{ee}. End-to-end delay is defined as the total delay required for an input speech sample to be encoded, transmitted through the network, and decoded at the receiver [10]. τ_{ee} is expressed as follows,

$$\tau_{ee} = \tau_{enc} + \tau_{dec} + \tau_{trans} \tag{2.1}$$

where τ_{enc} and τ_{dec} represent the total encoder and decoder delays respectively, and τ_{trans} represents the transmission delay. Specifically, τ_{enc} includes the delays for buffering the input speech samples for analysis and packetization, and τ_{dec} includes the delays for de-packetization and synthesis. Although $\tau_{enc} \geq \tau_{dec}$ for most low bit rate speech coders [10], this thesis assumes $\tau_{enc} = \tau_{dec}$ for simplicity, and that each of these delays is equivalent to the number of speech frames per packet.

τ_{trans} is a function of the transmission medium and distance between the source encoding and decoding systems. As reported in the ITU-T Recommendation G.114 [8], the transmission delay across a purely digital network is calculated as follows,

$$\tau_{trans} = 0.004 \times (\text{distance in kilometers}) \tag{2.2}$$

When maximizing the QoS of Internet Telephony, minimizing τ_{ee} is highly desirable for the following two reasons: (1) echo reduction and (2) improved conversational dynamics. Each of these reasons is discussed below.

Echo

Echoes, due to hybrid mismatches at exchange connections, become annoying for both the speaker and listener on circuits having long delays in transmission paths [22]. Echos occurring on circuits with one-way delays of more than 25 ms are perceived as conversational sidetones; when delays exceeds 75 ms, echo is noticeable; for delays exceeding 125 ms, speech perception may be severely impacted if echo control is not used [2, 25]. In mobile communication systems, there exists an inherent transmission delay of approximately 270 ms [8]; thus, any additional delay may degrade speech quality. Although echo cancellers have been developed to eliminate the effects of long transmission delays, reducing the disturbing effects of echo clearly requires minimizing end-to-end delay.

Dynamics of Conversation

The second reason for minimizing end-to-end delay is to improve the dynamics of conversation. As reported in the ITU-T Recommendation G.114, a subjective test intended to evaluate the effects of pure delay on echo-free telephone connections was completed by volunteer Telco customers in 1991 [8]. The calls from these customers were routed through a laboratory where varying amounts of one-way delay between 0 ms and 750 ms were added. Standard overall MOS quality was then used to subjectively measure the following characteristics on the dynamics of conversation: (1) the ease of interruption, (2) the necessity of repeating utterances, and (3) the attentiveness, responsiveness, and helpfulness of the partner. The tests results showed that calls with 0 ms of one-way inserted delay were rated "good," 250 ms of inserted delay were rated "fair," and 500 ms of inserted delay were rated "poor." These results are illustrated in Figure 2-2 [8] and clearly show that an increase in end-to-end delay degrades the overall MOS quality on the dynamics of conversation.

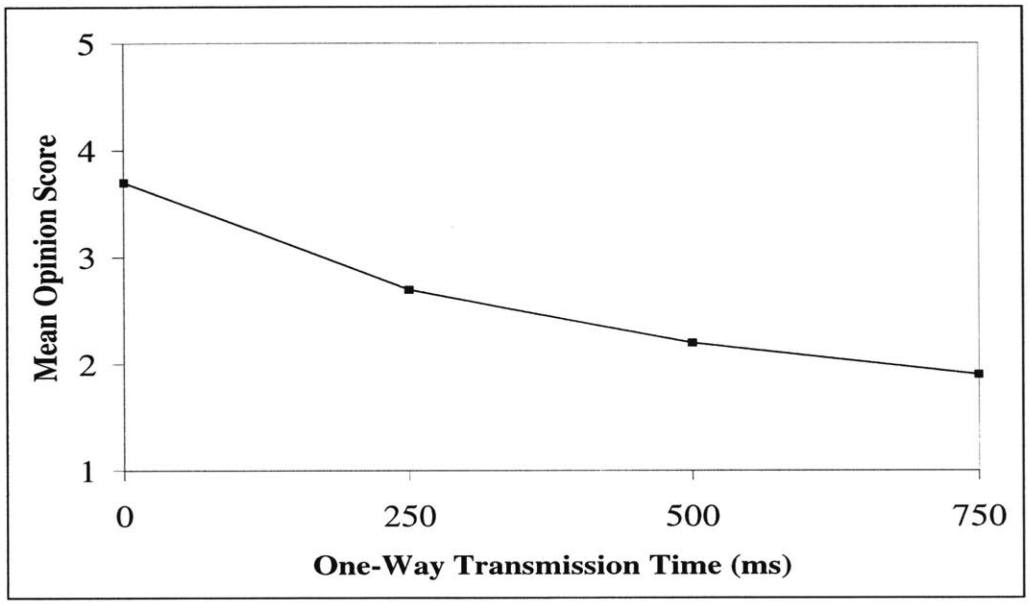

Figure 2-2: MOS for Four Delay Conditions.

2.3 Design Methods

Since maximizing the QoS of Internet Telephony requires maximizing speech quality and minimizing end-to-end delay, three important questions arise when designing a VoIP system based on these criteria. The following subsections present these questions and introduce the design methods required for solving each of them.

2.3.1 Protocol Optimization

The first question to be asked in designing an Internet Telephony system is the following: In transmitting speech packets over the Internet, what set of rules or protocols must be followed by the sender and receiver in order to maximize speech quality and minimize end-to-end delay? The answer to this question is explored in Chapters 3 and 4 and leads to the first design method in maximizing the QoS of Internet Telephony — protocol optimization.

2.3.2 Packet Recovery Optimization

The second question to be asked in developing an Internet Telephony system is the following: Given the Internet's inherent nature of packet loss, how can missing packets be recovered at the receiver in order to maximize speech quality? The answer to this question is presented in Chapters 5 and 6 and leads to the second design method in maximizing the QoS of Internet Telephony — packet recovery optimization.

2.3.3 Packet Size Optimization

Finally, the third question to be asked in developing an Internet Telephony system is following: To maximize speech quality and minimize end-to-end delay, how many frames of speech should be grouped into a voice packet for transmission across the Internet? The answer to this question is explored in Chapter 7 and leads to the third design method in maximizing the QoS of Internet Telephony — packet size optimization.

Chapter 3

TCP/IP Protocol Fundamentals

The following chapter is one of the main contributions of this thesis and provides a detailed treatment on TCP/IP protocols. In the following sections, the four layers of the TCP/IP protocol stack — the application layer, the transport layer, the network layer, and the link layer — are first examined. Next, the encapsulation and demultiplexing of TCP/IP data is described, followed by a presentation on the roles of clients, servers, and routers within a TCP/IP networking environment. This chapter thus provides the necessary fundamentals for understanding the issues of protocol optimization in Chapter 4.

3.1 The TCP/IP Protocol Suite

A protocol is similar to a language, conveying meaning and understanding through some form of communication. Computer communication protocols are a set of rules and message exchanges such that for one computer to "talk" to another, each must be able to understand the other's language or protocol. The TCP/IP protocol suite, which forms the backbone for Internet communication, allows for computers of all sizes, from many different computer vendors running completely different operating systems, to communicate with one another.

3.2 TCP/IP Layering

Networking protocols are normally developed in layers with each layer responsible for a different aspect of communication. The TCP/IP protocol suite is the combination of different protocols and is normally considered to be a four-layer stack as illustrated in Figure

3-1 [24]. Working down the protocol stack from the application layer to the link layer, the responsibility of each layer is described in the following subsections.

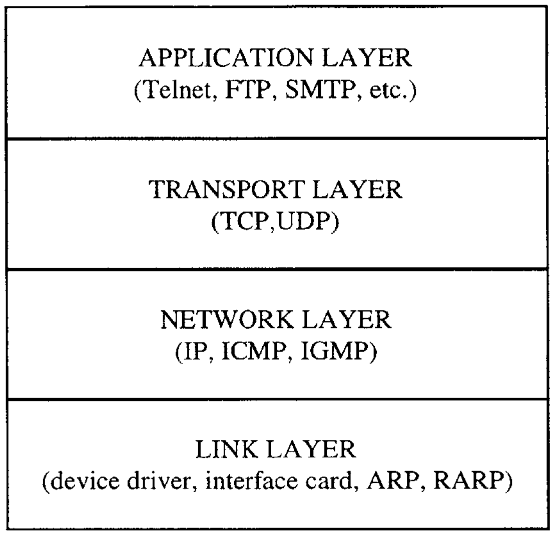

Figure 3-1: Four Layers of the TCP/IP Protocol Stack.

3.2.1 Application Layer

The application layer manages the user program or hardware device generating data to the lower layers [24]. Examples of common applications included in every TCP/IP implementation are the following: (1) Telnet for remote login, (2) FTP for file transfer, and (3) SMTP for email.

3.2.2 Transport Layer

The transport layer provides the flow of data between two application layer entities. In the TCP/IP protocol suite, there are two different transport layer protocols: (1) the Transmission Control Protocol (TCP) and (2) the User Datagram Protocol (UDP) [24].

Transmission Control Protocol (TCP)

TCP provides a reliable flow of information by establishing a bi-directional or full-duplex connection between two systems before data transmission begins. Since TCP is connectionoriented and reliable, information delivered from one system to the other is guaranteed to arrive error-free and in order [15, 24].

The primary responsibilities of TCP include the following: (1) breaking data into segments passed to it from the application layer, (2) computing checksums for each segment, (3) adding sequence numbers to each segment for identification, (4) acknowledging received segments from the other system, (5) setting timeouts to ensure the other system acknowledges information that is sent, and (6) retransmitting lost or corrupted segments if necessary [24]. Thus, TCP provides end-to-end flow control by ensuring reliable, in-time, and error-free delivery of information.

In addition, several variants upon the original TCP implementation have been developed for improving the algorithms for packet retransmission and congestion control. In chronological order, these variants include TCP Tahoe, TCP Reno, and TCP Vegas [3]; their unique features are presented in Chapter 4.

User Datagram Protocol (UDP)

UDP is inherently uni-directional or simplex and does not require connection establishment between two systems before data transmission begins. Since UDP is connectionless and unreliable, information delivered from one system to the other is not guaranteed to arrive error-free or in-order [15, 24].

The major responsibilities of UDP include breaking data into datagrams passed to it from the application layer and computing checksums to detect corrupted datagrams. Unlike TCP, UDP does not provide retransmission of corrupted information nor does it resequence disordered datagrams. UDP only delivers datagrams from one system to the other without guaranteeing error-free arrivals at the other end [24]. Thus, given the lack of packet retransmission or congestion control strategies, the end-to-end delay associated with transmitting data over UDP is typically less than that with transmitting data over TCP.

3.2.3 Network Layer

The network layer, otherwise known as the internet layer, handles the movement and routing of information throughout the network. The network layer protocols include the Internet Protocol (IP), the Internet Control Message Protocol (ICMP), and the Internet Group Management Protocol (IGMP) [24].

Internet Protocol (IP)

IP is the central protocol at the network layer; the goal of IP is to route TCP or UDP data throughout a network until it reaches its destination. Thus, all TCP or UDP data is received by IP at the source and destination systems and at every intermediate system within the network. However, IP may occasionally lose information somewhere within the network, deliver corrupted information, or choose to route different messages on different paths, thereby resulting in data to arrive disordered at the destination [24]. By only providing a "best effort service" [15], IP thus relies on the transport or application layers to add any desired reliability to the system.

One of the key parameters delivered by the transport layer to IP is the data's destination address. Before information is sent between two systems, IP prepends a header to the data containing both the source and destination addresses. The destination address is then examined at every node within the network to determine how to route the data to the final destination. These 32-bit addresses are known as Internet addresses, or IP addresses, and are normally represented in human readable, dotted-decimal notation form (e.g., 134.133.40.6) [15]. Every system on a network interface must have a unique 32-bit IP address in order to singularly identify it from any other system on any other network interface.

3.2.4 Link Layer

The link layer, otherwise known as the data-link or the network interface layer, includes the operating system's device driver, the network interface card, and the following protocols: (1) the Address Resolution Protocol (ARP) and (2) the Reverse Address Resolution Protocol (RARP) [24]. ARP and RARP are specialized protocols that convert between IP addresses and hardware addresses at the link layer. Together, the device driver and network interface card handle all the necessary details for interfacing with the transmission cable or other communications medium. Physical communication media at the link layer include Ethernet, Token Ring, and Fiber Distributed Data Interface (FDDI) [24].

3.3 Moving Up and Down the TCP/IP Protocol Stack

Data which is sent down the four layers of the TCP/IP protocol stack is finally transmitted as a stream of bits across the network. Each layer adds prepends a header and sometimes appends trailer information to the data that is passed to it. This process is known as data encapsulation [24] and is illustrated in Figure 3-2.

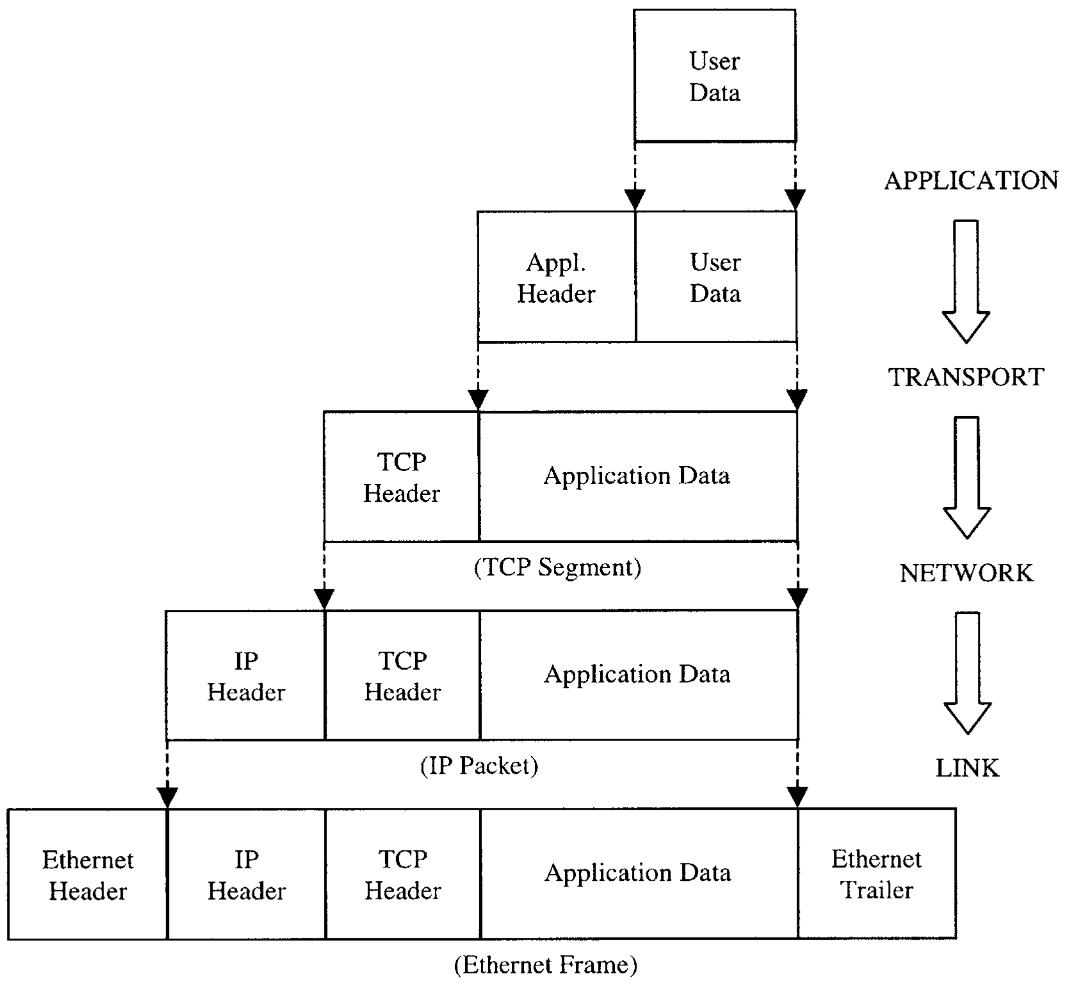

Figure 3-2: TCP/IP Data Encapsulation.

3.3.1 Data Encapsulation

Data which is sent from the transport layer to the network layer either arrives as a TCP segment or a UDP datagram. The unit of data that IP then sends to the link layer is called an IP packet while the stream of bits that flows across the transmission medium, such as Ethernet, is known as a frame [24].

TCP and UDP Header

As shown in Figure 3-2, TCP prepends a header to the data that is passed down from the application layer. The TCP header is 20 bytes in length and contains the following major identifiers: (1) 16-bit source port number, (2) 16-bit destination port number, (3) 32-bit sequence number, (4) 32-bit acknowledgment number, (5) 16-bit window size, and (6) 16-bit TCP checksum. The UDP header is 8 bytes in length and contains the following major identifiers: (1) 16-bit source number, (2) 16-bit destination number, and (3) 16-bit UDP checksum [24].

Since different system applications send data to either TCP or UDP, each transport protocol uses its 16-bit source and destination port numbers to identify the application to which the data belongs. For instance, the SMTP email application always waits for a TCP connection on port 25. Other well-known ports assigned for the most common TCP/IP applications include TCP port 23 for Telnet and TCP port 21 for FTP [15].

IP Header

The IP header is 20 bytes in length and contains the following major identifiers: (1) 8-bit protocol type, (2) 16-bit IP checksum, (3) 32-bit source IP address, and (4) 32-bit destination IP address. Since TCP and UDP both send data to the network layer, IP stores a value in its 8-bit protocol type field to identify the transport protocol to which the data belongs. A value of 6 denotes the segment was sent by TCP and a value of 17 denotes the datagram was sent by UDP [15].

Ethernet Header

When the link layer receives data, the Ethernet driver prepends header and trailer information to the IP packet. The Ethernet header is 14 bytes in length and contains the following major identifiers: (1) 48-bit destination hardware address, (2) 48-bit source hardware address, and (3) 16-bit frame type. The Ethernet trailer is 4 bytes in length and consists of a cyclical redundancy check (CRC) required for error-checking [24]. It is important to note that Ethernet's 48-bit source and destination hardware addresses are used to identify the network interface to which the source and destination systems belong. These addresses are not the same as the 32-bit source and destination IP addresses which explicitly identify the names of the source and destination systems on the network interfaces. Like the transport and network layers, when the link layer sends information on behalf of IP, ARP, and RARP, Ethernet stores a value in its 16-bit frame type field to identify the network or link layer protocol to which the packet belongs.

3.3.2 Demultiplexing

When data is first received by the destination system, the Ethernet frame travels up the TCP/IP protocol stack where all the corresponding headers are examined and removed at the appropriate layer. Each protocol examines the identifiers in its header to determine which upper-layer protocol should receive the data. This process is known as demultiplexing [24]. Beginning at the link layer, the Ethernet device driver examines the 16-bit frame type field in the Ethernet header to determine whether to send the packet to IP, ARP, or RARP. If the data is sent to the network layer, IP then examines the 8-bit protocol type field in its header to determine whether to send the segment to TCP or the datagram to UDP. Finally, TCP or UDP examines the 16-bit destination port number in its header to determine the application to which the data should be delivered.

3.4 Clients, Servers, and Routers

By considering a collection of networks, or an internet, as a medium for information transmission, a common model for sending and receiving data is the client-server model [15, 24]. The client-server model consists of three components: (1) the client, (2) the server, and (3) the network. A client is a software application that runs on a user's system while a server is a software application that runs on a provider's system. A server is thus analogous to a television station which broadcasts its information to anyone wishing to receive it; a client is analogous to a television set which tunes to the station for a desired broadcast.

The simplest way of interconnecting a client and server, existing on different networks, is with a router. A router, otherwise known as a hop or a node, is a special purpose hardware box for interconnecting many different types of physical networks including Ethernet, Token Ring, and FDDI. Since an internet is a collection of networks that use the same protocol suite, any system on an Ethernet network for instance, can communicate with any system on a Token Ring network [24].

In the client-server model, both the client and server are known as end systems while the router is known as an intermediate system. Specifically, the application and transport layers of the client and server use end-to-end protocols for communication whereas the network layer uses hop-by-hop protocols at both end systems and at every intermediate system. Thus, each layer on one system can communicate with its peer layer on another system [24]. This is illustrated with a simple FTP client-server application in Figure 3-3.

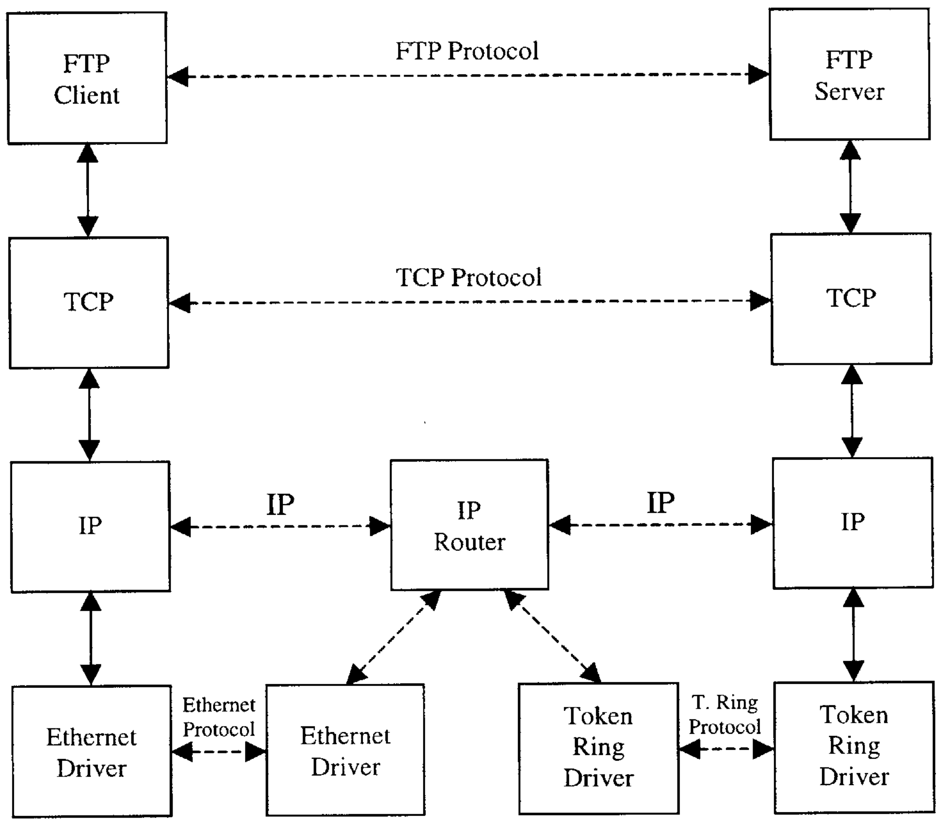

Figure 3-3: FTP Client-Server Application.

Chapter 4

Protocol Optimization

Building upon the material presented in Chapter 3, the following sections present an in-depth treatment on the issues of real-time data transmission and the disadvantages of using both TCP and UDP as target protocols for sending voice packets over the Internet. Based on these qualitative findings, the real-time transport protocol, combining the reliability of TCP and speed of UDP, is introduced and evaluated as a near-optimal protocol for real-time voice transmission.

4.1 Real-Time Applications

The purpose of Internet Telephony is to deliver speech packets from in real-time over the Internet. Real-time transmission imposes an upper bound on the delay between the sender and receiver such that packets that fail to arrive by a particular deadline are considered missing. [17]. Thus, real-time transmission represents an attribute of stream-based transmission for which the original framing and timing of packets must nearly be reproduced upon reception, with little or no distinction for an end user between a stream or packets which has its source on the local system and the same stream coming over a network connection.

For Internet Telephony, voice packets that fail to arrive within a narrowly prescribed window of time, and hence miss their deadline, degrade speech quality. Since IP is inherently unreliable, lost packets due to router discard, network delay, or a combination of both, clearly degrade speech quality during playout unless packet recovery is initiated. However, any form of packet retransmission only increases the end-to-end delay, thereby reducing the conversational dynamics between speaker and listener (assuming echo is adequately controlled). In order to maximize speech quality and minimize end-to-end delay, the first goal in designing an Internet Telephony system is to determine an optimal set of protocols for transmitting voice packets in real-time over the Internet. The following sections present the disadvantages of using TCP and UDP for real-time voice transmission and introduce an improved protocol, the real-time transport protocol, described by RFC 1889 [21], suited for this task.

4.2 Real-Time Applications over TCP

As presented in subsection 3.2.2, TCP is a reliable transport protocol that guarantees orderly and error-free delivery of packets. Although TCP maximizes speech quality, the following subsections describe three reasons for which TCP is actually inappropriate for transmitting delay-sensitive data such as real-time voice.

4.2.1 Packet Retransmission

Whenever a packet is transmitted from sender to receiver, TCP starts a retransmission timer [24]. If the sender fails to receive an acknowledgment (ACK) before the timer expires, the packet is retransmitted up to N number of times where the default value of N is dependent on the TCP vendor. For the Microsoft TCP/IP stack, $N = 5$ and the retransmission timer is initialized to three seconds when a TCP connection is first established [3]. During transmission, the value of the timer is continually adjusted to match the characteristics of the network using smoothed round-trip time (SRTT) calculations. The current timer value is then doubled after each retransmission of that packet [3]. For high rates of packet loss, one retransmission timeout, likely more, expires, thereby forcing the receiver to wait for the missing packet and generate audible gaps during playout.

Although delay is often reduced for packet retransmission over local area networks (LANs), as opposed to a wide area networks (WANs), congestion may arise over WANs due to the presence of routers. Network congestion occurs when packets arriving from a higher capacity network are routed to a lower capacity network or when multiple packets arriving at a node are transmitted through a single output stream [24]. In both cases, packets may arrive either delayed at the receiver due to an increase in processing time or discarded from the network due to a saturation of router memory required for queuing incoming packets. The following subsections present four TCP congestion control mechanisms and the problems they generate during real-time voice transmission.

4.2.2 Slow Start and Congestion Avoidance

The Tahoe, Reno, and Vegas implementations of TCP support two congestion control mechanisms for maintaining a comfortable rate of packet transmission: (1) Slow Start and (2) Congestion Avoidance [24]. The purpose of Slow Start is to gradually increase the rate in which packets are injected into a network. Whenever a TCP session is first established, Slow Start initializes one of the sender's parameters, known as the congestion window, to one packet length. The sender then transmits one packet over the network and waits for an ACK. Every time the sender receives an ACK, the congestion window is increased by one packet length up to the maximum window size advertised by the receiver, thereby exponentially increasing the number of packets sent across the network [24]. When the capacity of the network is reached and a packet loss is first detected, Slow Start ends and Congestion Avoidance is initiated. The purpose of Congestion Avoidance is to decrease the rate in which packets are injected into the network by linearly increasing the size of the congestion window by one divided by the current size of the congestion window, for every ACK received. When the sender finally determines an acceptable level of network congestion through SRTT measurements, Slow Start is again re-initiated [24].

In practice, Slow Start and Congestion Avoidance are independent algorithms that are implemented together to achieve congestion control. The primary difference between the two algorithms is that Slow Start exponentially increases the rate of packet transmission to achieve maximum throughput while Congestion Avoidance linearly increases the rate to decrease the level of network congestion. For Internet Telephony, the problems of using these algorithms include an initial Slow Start delay and a starved receiver during Congestion Avoidance.

During Slow Start, the receiver initially incurs a delay by waiting for the arrival of future packets as the maximum size of the congestion window is gradually reached [17, 18]. This delay increases the overall end-to-end delay for the system and thus reduces the speech quality of the conversation. When a packet loss is then detected and Congestion Avoidance is initiated, the gradual decrease in the congestion window's size leads to another problem.

Since real-time speech requires a natural rate of playout (e.g., 64 kb/s for PCM) that cannot be suddenly decreased without starving the receiver, a sudden decrease in the size of the congestion window results in the receiver generating audible gaps during playout which degrades speech quality [17, 18]. Thus, Slow Start and Congestion Avoidance are inappropriate algorithms for controlling congestion when transmitting real-time voice over the Internet.

4.2.3 Fast Retransmit and Fast Recovery

In addition to Slow Start and Congestion Avoidance, the Reno and Vegas TCP implementations include the Fast Retransmit and Fast Recovery algorithms [24] for treating congestion. Whenever an out-of-order packet is received under these TCP variants, the receiver generates an immediate ACK, known as a duplicate ACK. The purpose of this ACK is to inform the sender that a disordered packet was received and of the sequence number it is expecting. Since the sender does not know whether the ACK was caused by a lost or an out-of-order packet, the sender waits for a small number of duplicate ACKs to be received. If there is just a reordering of packets, it is assumed that only one or two duplicate ACKs will arrive before the reordered packet is processed. However, if three or more duplicate ACKs arrive in a row, this strongly indicates that the packet was lost and the sender immediately retransmits the missing packet without waiting for the retransmission timer to expire [24]. The retransmission of packets after three consecutive ACKs are received is known as the Fast Retransmit algorithm. Next, Congestion Avoidance, not Slow Start, is performed. This is known as Fast Recovery. Although Fast Retransmit and Fast Recovery significantly reduce the delay associated with regular packet retransmission, these algorithms nevertheless increase the end-to-end delay by up to three round-trip times, thereby forcing the receiver to wait for the missing packet and generate audible gaps during playout.

4.3 Real-Time Applications over UDP

As presented in subsection 3.2.2, UDP is an unreliable protocol that fails to provide error-free delivery of packets; lost or corrupted packets are simply discarded and not retransmitted. Out-of-order packets are also discarded at the receiver since UDP does not contain a sequence number field in its header to detect the sequential arrival of incoming packets. Clearly, UDP's lack of packet retransmission and congestion control strategies lead to degradations in speech quality; however, this also enables UDP to minimize the end-to-end delay incurred when sending packets over the Internet.

Since maximizing the QoS of Internet Telephony requires maximizing speech quality and minimizing end-to-end delay, the real-time transport protocol (RTP), specified in RFC 1889 [21], combines the reliability of TCP and speed of UDP. Approved by the IESG as an engineering proposed standard in 1995 [18], RTP has since been used in several of the industry's real-time audio and video applications such as Netscape LiveMedia^{TM} and Mircosoft NetMeeting^{TM} [18]. The advantages that make RTP a near-optimal protocol for real-time voice over the Internet are described in the following section.

4.4 Real-Time Transport Protocol (RTP)

RTP is the Internet standard protocol for the transport of real-time information and consists of two parts: (1) a data part, RTP, and (2) a control part, RTCP [18, 21]. RTP is a thin protocol that provides support for applications with real-time properties including timing reconstruction, loss detection, security, and content identification [21]. RTCP, on the other hand, provides support for real-time group conferencing within an internet. This support includes source identification, quality-of-service feedback, as well as synchronization of different media streams [18]. The following subsections present the advantages of RTP over TCP and UDP for point-to-point Internet Telephony and Internet voice-conferencing.

4.4.1 Point-to-Point Internet Telephony

As illustrated in Figure 4-1, RTP/RTCP is implemented within the application layer of the TCP/IP protocol suite. This is known as known as application-level framing [18]. RTP is therefore not a transport layer protocol per se, but instead, exhibits the important properties of being a transport protocol by providing end-to-end communication, data encapsulation, and demultiplexing. Information which is delivered by the application layer to the RTP protocol is prepended with a 12-byte header that includes the following major identifiers: (1) 7-bit payload type, (2) 32-bit timestamp, (3) 16-bit sequence number, and (4) 32-bit synchronization source [21]. After RTP data encapsulation, the RTP packet is delivered to UDP, then down the rest of the TCP/IP protocol stack. By providing the necessary header information while working on top of UDP, RTP provides two important features for maximizing speech quality and minimizing end-to-end delay.

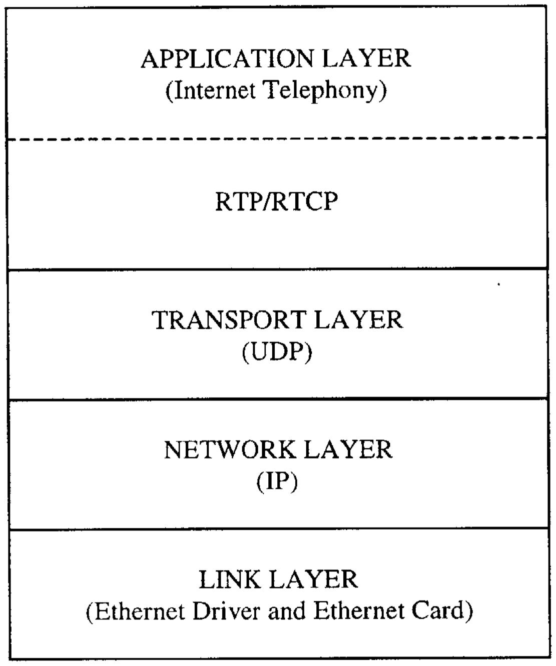

Figure 4-1: RTP/RTCP in TCP/IP Protocol Stack.

First, RTP contains a 16-bit sequence number field in its header to allow the destination system to detect packet loss and to restore packet sequence. Although RTP does not perform error detection, packet recovery, or packet re-ordering, it provides the application layer with 16-bit sequence numbers so that any desired reliability may be added [18, 21]. Thus, RTP mimics the reliability of TCP by providing the application layer with a mechanism to recover missing packets and to sequence disordered ones, thereby maximizing speech quality.

Secondly, RTP works over UDP and thus uses the potential for minimizing end-to-end delay when sending information over the Internet. Since UDP does not perform any packet retransmission or congestion control, RTP, by working over UDP, similarly minimizes the end-to-end delay when delivering packets over the Internet [18, 21]. Combined together, RTP's 16-bit sequence number field and performance over UDP provide two essential features of a near-optimal protocol for maximizing the speech quality and minimizing the end-to-end delay for point-to-point Internet Telephony.

4.4.2 Voice-Conferencing

Multi-Casting [17, 18] is an important feature of RTP that provides an additional advantage over the traditional TCP/IP transport protocols. Unlike TCP and UDP which only support uni-cast, RTP multi-casting allows a client to send a single stream of voice packets to several hosts for voice-conferencing, thereby minimizing the use of bandwidth [17]. The features of RTP that enable multi-casting are described below.

During voice-conferencing, network congestion levels increase or decrease as users enter and leave the conference. To accommodate a new participant who is connected through a low-bandwidth link or to react to indications of network congestion, senders may change the type of speech encoding depending on the conditions of the multi-casting environment. Thus, the RTP header includes a 7-bit payload type to identify the type of speech coding (e.g., 64 kb/s PCM, 32 kb/s ADPCM, or LPC10e) for each packet [19, 21]. Clearly, RTP is more flexible and efficient than either TCP or UDP by allowing different methods of speech encoding depending on the available bandwidth and network conditions of the multi-casting environment.

In addition, the RTP header provides a 32-bit timestamp and 32-bit synchronization source field to ensure multi-casting reliability. For every speech packet that is sent through the network, the value of the timestamp is incremented by the packetization interval, measured in frame lengths, times the sampling rate [19, 21]. At the receiver, RTP packet timestamps are then examined to correctly place incoming speech packets, each from a potentially different user or synchronization source, in the correct timing order such that packets from different sources may be contiguously played out on the loudspeaker. For instance, given one synchronization source sending 20-ms speech packets and another source sending 40-ms speech packets, the values of their timestamps computed and examined at the receiver to determine when each packet from each source should be played. Combined with RTP's 16-bit sequence number, reliability is thus added for multi-casting environments such that packets from different sources are contiguously played in the correct order at the receiver [19, 21]. Thus, given the required mechanisms for multi-casting, RTP is suitable for both point-to-point Internet Telephony and voice-conferencing, where its ability to maximize speech quality and minimize delay make it an improved protocol than either TCP or UDP for real-time voice transmission.

Chapter 5

Speech Coding Fundamentals

The following chapter is one of the main contributions of this thesis and provides a detailed treatment on speech coding. In the following sections, the methods of human speech production and the important properties of voiced and unvoiced speech are first introduced. Next, three types of digital speech coders — waveform coders, voice coders, and hybrid coders — are presented with an emphasis placed on the theoretical model of speech production as applied in voice coding and hybrid coding. Based on the theoretical speech production model, three low bit rate digital speech coding techniques — Linear Predictive Coding, Pitch Predictive Coding, and Vector Quantization — are explored, followed by an examination of the analysis-and-synthesis and analysis-by-synthesis speech coding strategies. This chapter thus provides the necessary fundamentals for understanding the issues of packet recovery optimization in Chapter 6.

5.1 Speech Production

Human speech organs are very complex, possessing both biological and acoustical functions. In this thesis, only the acoustical functions are considered. The speech organs can be divided into the following subsystems: (1) the lungs and trachea, (2) the larynx, and (3) the vocal tract. In electrical terms, the analog of these subsystems are the power supply, the signal generator, and the filtering process respectively [22]. A simple diagram of the human speech system is illustrated in Figure 5-1 [13].

Speech is generated by blowing air from the lungs through the vocal tract. In the adult male, the vocal tract is approximately 17 cm in length with a cross-sectional area that varies

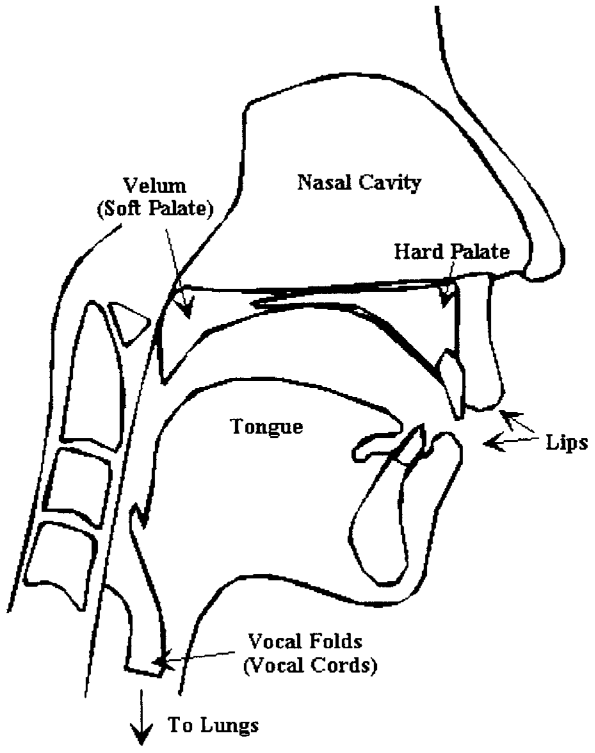

Figure 5-1: The Human Speech System.

from zero to about 20 cm^2 [13]. Situated in the larynx are a set of membranes known as the vocal folds. These membranes allow the area between the vocal folds, known as the glottis, to vary. For normal breathing, the vocal folds remain open, but during speech production, they open and close, generating two broad classifications of speech: (1) voiced speech and (2) unvoiced speech [13].

During voiced speech production, the vocal folds are normally closed. As air is pushed out of the lungs, pressure builds up behind the vocal folds and eventually forces them to open and vibrate. The frequency of vibration is known as the pitch and is determined by the length of the vocal folds and their tension. For normal speakers, the pitch ranges anywhere between 50 Hz to 500 Hz [13]. The effect of opening and closing the glottis during voiced speech is that air passing through the rest of the vocal tract appears as a quasi-periodic pulse train [13]. During unvoiced speech production, the vocal folds are either partially closed or completely open. In this configuration, air freely passes through the rest of vocal tract and appears as random noise [13].

As the glottal excitation travels through the vocal tract, the frequency contents of the signal are filtered by its cavity responses. The nature and characteristics of these filtered frequencies depend on the shape and positioning of the structures forming the vocal tract including the epiglottis, lower jaw, tongue, velum, and hard palate [22]. These filtered signals are then radiated and further modulated at the lips and teeth as shown in Figure 5-1.

5.2 Properties of Speech

Different speech sounds are distinguished by the human ear on the basis of their spectra and how these spectra evolve with time. For voiced speech such as vowels, the vocal tract acts as a resonance cavity where the resonant frequencies, for most humans, are centered at 500 Hz and its odd harmonics [13]. These frequencies produce large peaks in the resulting spectrum, known as formants, and contain almost all the information for the signal. Thus, the vocal tract can be effectively modeled using an all-pole linear system. Characteristics of voiced speech include quasi-periodicity, also known as the fine pitch structure, a low-pass spectrum, strong formant presence up to approximately 4 kHz (above 4 kHz, noise introduced by the turbulent flow of air begins to dominate), and a large dynamic range [13].

Unvoiced speech is generated when air passes through either a partially closed or completely open set of vocal folds. Characteristics of unvoiced speech include random noise-like semblance, a flat or high-pass spectrum, and the lack of strong formants or a fine pitch structure. In addition, the dynamic range for unvoiced speech is typically lower than that of voiced speech [13].

5.3 Digital Speech Coders

Digital speech coders can be broadly classified into the following two types: (1) waveform coders and (2) voice coders. A third type of speech coder, known as a hybrid coder, combines waveform and voice coding principles [10, 11, 22]. The following subsections present a brief description of waveform, voice, and hybrid speech coders.

5.3.1 Waveform Coding

Waveform coding attempts to preserve the time-domain waveform of the original speech by coding the waveform on a sample-by-sample basis. The sampling rate for speech is 8 kHz and every sample generated is represented, or quantized, with a certain number of bits, usually 8 or 16 bits for transmission. Although waveform coders are attractive for coding a wide variety of signals (e.g., music, tones, and voice band data) up to medium bit rates (e.g., 32 kb/s), they exhibit a graceful degradation in the presence of noise and transmission errors [10, 22]. In addition, their high bit rates are a often source of great inefficiency when bandwidth is limited. Examples of waveform coders include 64 kb/s PCM and 32 kb/s Adaptive Differential PCM (ADPCM) [13].

5.3.2 Voice Coding

At the opposite extreme of waveform coding is voice coding or vocoding. Unlike waveform coders, voice coders make no attempt to preserve the time-domain waveform of the original speech. Instead, vocoders employ analysis-and-synthesis techniques based on a theoretical model of speech production to generate an output signal that sounds similar to the input, whether of not the output time-domain waveform matches the original. The theoretical model of speech production assumes that speech is synthesized by exciting an all-pole linear system with a scaled periodic impulse train if the speech is voiced or random noise if the speech is unvoiced [10, 11, 22]. The theoretical model of speech production is illustrated in Figure 5-2.

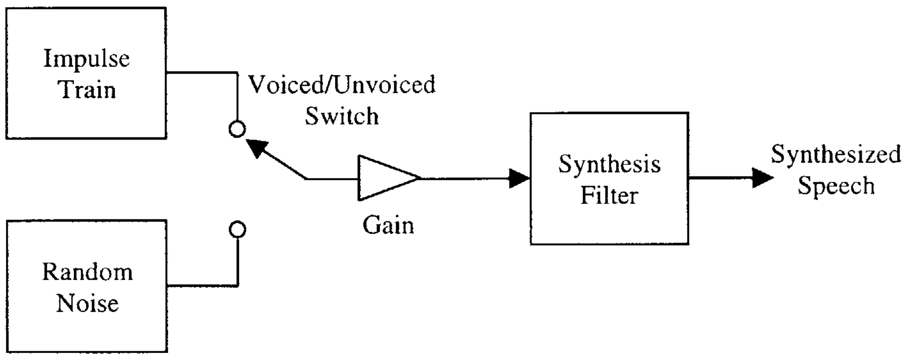

Figure 5-2: Theoretical Speech Production Model.

Based on the theoretical model of speech production, vocoders operate as follows: At the speech encoding system, the input waveform is analyzed to extract a set of parameters from the original speech. These parameters are then digitally coded and transmitted to the receiver. At the speech decoding system, the parameters are decoded to generate the excitation and synthesis filter where the output of the filter is the synthesized speech [10, 22]. As shown in Figure 5-2, if the original speech is voiced, the synthesis filter is excited by a periodic impulse train, where the distance between the pulses equals the pitch period. If the original speech is unvoiced, the synthesis filter is excited by random noise.

Since vocoders attempt to produce a signal that sounds similar to the original, highly intelligible speech is generated even at very low bit rates. However, as no attempt is made to preserve the original time-domain waveform, the synthesized speech often sounds unnatural or synthetic. Thus, vocoders are normally used when bit rate is of the utmost importance [10, 22]. Examples of vocoders include those employing the following techniques: (1) Linear Predictive Coding, (2) channel coding, (3) formant coding, and (4) phase coding [13]. This thesis focuses on Linear Predictive Coding techniques.

5.3.3 Hybrid Coding

As previously described, waveform coders attempt to preserve the original signals's time-domain waveform and are thus capable of providing very high quality speech at medium bit rates. However, they are inefficient and error-prone for coding speech at low bit rates. Vocoders, on the other hand, are often used for low bit rate coding and attempt to synthesize speech that sounds like the original waveform. Although the synthesized speech is highly intelligible, it also tends to sound very synthetic. Hybrid coders combine the high quality potential of waveform coding with the compression efficiency of vocoding [10, 22].

Since the excitation for the theoretical model of speech production is classified as either voiced or unvoiced, synthetic quality speech often results as no intermediate voicing states are permitted. Hybrid coders attempt to solve this problem by generating a time-varying signal ranging from either pulse-like to noise-like in characteristic to excite the synthesis filter [10, 22]. Thus, the main difference among different hybrid coders is the method of excitation which include Self-Excitation (SE), Regular-Pulse Excitation (RPE), Multi-Pulse Excitation (MPE), and Codebook-Excitation (CE). Combined with the principles of Linear Predictive Coding and analysis-by-synthesis techniques, hybrid coders generate better synthesized speech quality at low bit rates than traditional AAS vocoders. Examples of hybrid coders include 13 kb/s RPE, 9.6 kb/s MPE, and 4.8 kb/s CELP.

5.4 Low Bit Rate Speech Coding Techniques

Linear Predictive Coding (LPC) and Pitch Predictive Coding (PPC) are two of the most powerful techniques for processing speech at low bit rates. Each of these techniques takes advantage of the correlations between speech samples as well as the periodic similarities present in the time-domain waveform. After a speech signal is sampled, its parameters are then quantized with a finite number of bits. Vector Quantization (VQ) is often used to digitally represent these parameters [27]. The following subsections describe the LPC, PPC, and VQ techniques.

5.4.1 Linear Predictive Coding (LPC)

Close examination of speech signals reveals certain correlations between neighboring speech samples. These short-term correlations are caused by resonances in the vocal tract and appear every 125 μs at an 8 kHz sampling frequency [22]. Exploitation of these redundancies effectively allows for transmission of speech at low bit rates. LPC exploits the redundancies between speech samples by predicting the output sequence based on a linear combination of past inputs and or past outputs. In mathematical terms, this can be expressed as the following difference equation,

$$y(x) = \sum_{j=1}^{p} \alpha_j y(n-j) + \sum_{k=0}^{q} \xi_k x(n-k) \tag{5.1}$$

where $x(n)$ and $y(n)$ are the input and output respectively, and α_j and ξ_k are known as the LPC predictor coefficients. Assuming $\xi_0 = 1$ and from elementary Z-transforms, equation (5.1) can be expressed as the following transfer function,

$$H(z) = \frac{Y(z)}{H(z)} = \frac{1 + \sum_{k=1}^{q} \xi_k z^{-k}}{1 - \sum_{j=1}^{p} \alpha_j z^{-j}} \tag{5.2}$$

Since the vocal tract is the source of resonant frequencies and can thus be modeled as an all-pole linear filter, most LPC-based speech coders assume an all-pole or autoregressive (AR) model for equation (5.2) [22]. In this configuration, $\xi_k = 0$ for $k \geq 1$, and equation (5.2) reduces to the following LPC synthesis equation,

$$H(z) = \frac{Y(z)}{H(z)} = \frac{1}{1 - \sum_{j=1}^{p} \alpha_j z^{-j}} \tag{5.3}$$

LPC Analysis

In order to solve for the LPC predictor coefficients, α_j, in equation (5.3), a block of input speech samples, known as a frame, is first passed through an LPC analysis filter, $A_l(z)$. Since the short-term correlations between neighboring speech samples do not change very rapidly, speech is considered stationary over short, finite periods of time and analyzed in short time frames. Speech frames vary in size but usually lie between 10 ms and 40 ms [22]. The transfer function for the LPC analysis filter is expressed as the following,

$$A_l(z) = \sum_{j=1}^{p} \alpha_j z^{-j} \tag{5.4}$$

When the original speech frame, $x(n)$, is passed through $A_l(z)$, the difference between the input and output results in an error or residual signal, $e(n)$,

$$e(n) = x(n) - \sum_{j=1}^{p} \alpha_j z^{-j} \tag{5.5}$$

Since the goal of LPC analysis is to determine p coefficients for the LPC synthesis filter, the optimal coefficients are those that minimize $e(n)$. To obtain these coefficients, the Least-Squares approach or Lattice method may be used, as described in [10, 11, 22].

5.4.2 Pitch Predictive Coding (PPC)

In addition to short-term correlations, the quasi-periodicity of voiced speech exhibits pulse-like spikes at regular intervals. These redundancies are known as long-term correlations and are caused by the periodic vibrations of the vocal cords where the rate of opening and closing of the glottis is known as the harmonic frequency or pitch [22]. Since pitch pulses tend to take the same shape at regular intervals, PPC exploits these redundancies by predicting the output sequence based on a linear combination of past inputs and or past outputs. This is described by the following pitch synthesis equation,

$$P(z) = \frac{1}{1 - \sum_{k=-i}^{i} \beta_k z^{-(M+k)}} \tag{5.6}$$

where $x(n)$ and $y(n)$ are the input and output respectively, M is the pitch period or pitch lag, and β_k represents the pitch predictor coefficients relating to the degree of signal periodicity or correlation between distant pitch pulses. It must be noted that since the modeling of the glottis is only based on voiced speech, PPC is inefficient for unvoiced speech where there exists little correlation between neighboring samples.

Accurate estimation of the pitch period, M, is essential for good pitch prediction. The most common pitch period estimation algorithms include the Average Magnitude Difference Function (AMDF) and the Autocorrelation Function (ACF) as described in [10, 11, 22]. The main principles behind these pitch detection algorithms is to find the pitch by comparing the similarity between the original signal and its shifted version. If the shifted distance is equal to the pitch, then the two signal waveforms have the greatest similarity.

Pitch Analysis

After determining the pitch lag, M, using either AMDF or ACF, a pitch analysis filter, A_p, is then used to determine the pitch predictor coefficients. The transfer function for the pitch analysis filter is expressed as the following,

$$A_p(z) = \sum_{k=i}^{-i} \beta_k z^{-(M+k)} \tag{5.7}$$

where given the optimal M, the optimal pitch predictor coefficients are determined by solving a set of simultaneous linear equations in the unknowns, β_k.

In performing pitch analysis, two separate procedures can be applied: (1) Open-Loop pitch analysis and (2) Closed-Loop pitch analysis [10, 11, 22]. Open-Loop analysis is typically applied in AAS-LPC coding schemes, while Closed-Loop is typically applied in ABS-LPC coding schemes. Each of these procedures is presented in sections 5.5 and 5.6 respectively.

5.4.3 Vector Quantization (VQ)

The conversion of analog signals to digital information requires sampling and quantization. Sampling is the discrete representation of analog signals along the time axis while quantization is the discrete representation of analog signals along the amplitude axis.

Waveform coding uses scalar quantization in which continuous values are mapped to the nearest level from a finite number of levels. Although scalar quantization is simple, robust to errors, and offers linear correlation between speech samples, it is expensive in terms of capacity and does not consider vector dimensionality or non-linear dependencies between samples [10, 11, 22]. From Shannon's Rate Distortion Theorem [4] which states that better performance is always achieved by coding vectors instead of scalars, VQ was thus introduced in LPC speech coding. Among the benefits of VQ include transmission rates of about 1 bit per parameter or less which is exactly where the performance of scalar quantization sharply degrades [22]. This section introduces the principles of VQ.

VQ is formulated as follows: Assume $\mathbf{x}$ is an N-dimensional vector whose components are real-valued, continuous amplitude random variables. Thus, $\mathbf{x}$ is expressed as follows,

$$\mathbf{x} = [x_1, x_2, x_3, ..., x_N]^T \tag{5.8}$$

Using VQ, $\mathbf{x}$ is mapped onto a real-valued, discrete amplitude, N-dimensional vector, $\mathbf{y}$, which is expressed as follows,

$$\mathbf{y} = [y_1, y_2, y_3, ..., y_N]^T \tag{5.9}$$

Thus, $\mathbf{y}$ is the vector quantized version of $\mathbf{x}$, or in mathematical notation,

$$\mathbf{y} = Q(\mathbf{x}) \tag{5.10}$$

where Q is the quantization operator.

Typically, $\mathbf{y}$ takes on one of a finite set of vectors from an L-level codebook, where L is the number of entries in the codebook. The codebook, Y, is expressed as the following,

$$\mathbf{Y} = [\mathbf{y_1}, \mathbf{y_2}, \mathbf{y_3}, ..., \mathbf{y_L}] \tag{5.11}$$

Designing the L-level codebook requires partitioning the N-dimensional space into L

cells, $C = [C_i,\ 1 \leq i \leq L]$, and associates for each cell, a vector $\mathbf{y_i} = [y_1, y_2, ..., y_N]^T$, usually known as the centroid or code-vector. Techniques for constructing the L-level codebook include the Linde-Buzo-Gray algorithm, Vector Predictive Quantization, and Finite-State VQ, as described in [10, 11, 22].

The size of an L-level codebook is usually a power of two and the distortion rate, R, of the vector quantizer is expressed in bits per vector as follows,

$$R = \log_2 L \tag{5.12}$$

Thus, the number of bits, r, required to code each sample in the code-vector is expressed in bits per sample as follows,

$$r = \frac{R}{N} \tag{5.13}$$

Clearly, from equation (5.13), a fractional number of bits per sample can be achieved using VQ. Thus, VQ is more efficient than scalar quantization by reducing the number of bits required for transmission.

5.5 Analysis-and-Synthesis (AAS) Coding of Speech

In traditional vocoders, the original speech is represented with a set of parameters that is based on the theoretical model of speech production. Since speech can be described in terms of its formants, pitch, and gain, these parameters are extracted from the original speech, digitally coded, and transmitted to the receiver. The process of extracting parameters from the original speech is known as analysis and includes both LPC and PPC analysis as described in subsections 5.4.1 and 5.4.2 respectively. At the decoder, synthesis applies the extracted parameters to the theoretical model of speech production to generate the synthesized speech. Combined together, AAS coding of speech has generated high quality coders with bit rates between 8 kb/s and 16 kb/s [10, 11, 22]. Figure 5-3 illustrates an analysis-and-synthesis speech coding system. A treatment on the operation of AAS speech coding is presented below.

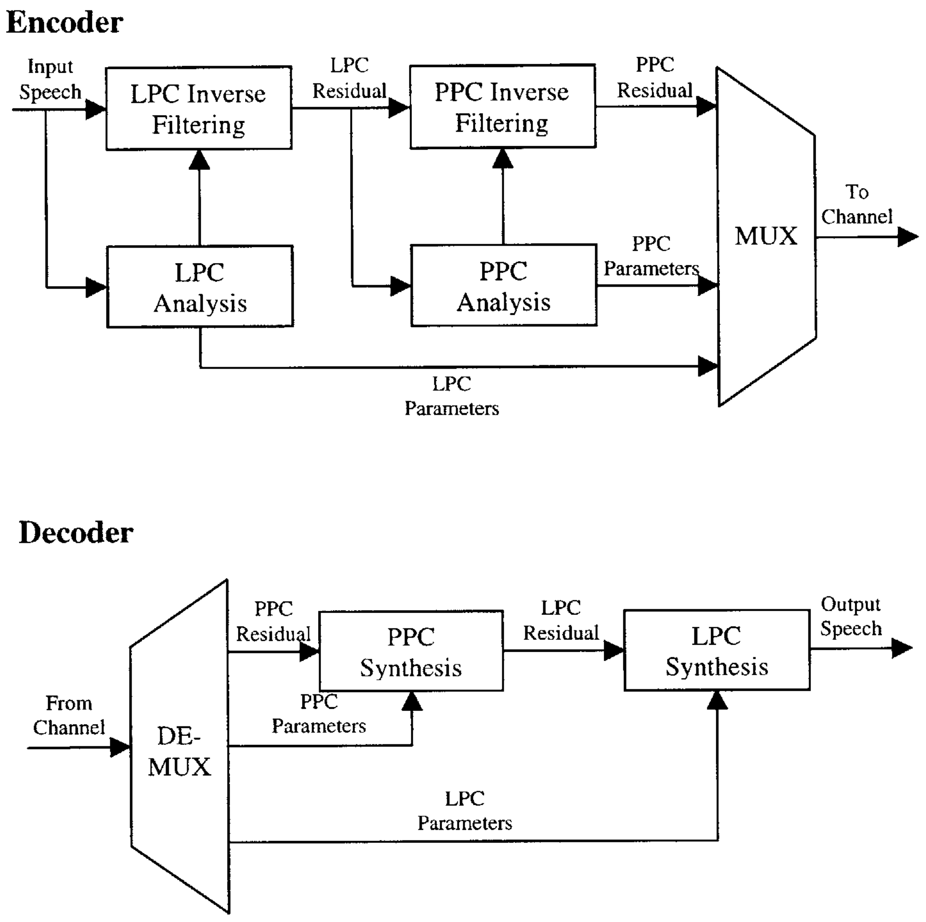

Figure 5-3: Analysis-and-Synthesis Speech Coding System.

AAS Encoder

At the AAS encoder, an input frame is first passed through the LPC analysis filter, $A_l(z)$, in order to determine p optimal LPC predictor coefficients as described in subsection 5.4.1. After the LPC coefficients are determined, they are represented in the form of Line Spectral Pair (LSP) parameters, quantized, and transmitted to the decoder. The purpose of representing LPC coefficients in the form of LSPs is because LSP parameters have a well-behaved dynamic range and stability preservation properties that allow efficient encoding of spectral information [10, 11, 22].

Inverse LPC Filtering After LPC analysis, the input frame is passed through an inverse LPC synthesis filter, $H^{-1}(z)$, which generates an LPC residual signal. If no pitch prediction is performed, the LPC residual is simply quantized and transmitted to the decoder. Otherwise, pitch prediction is performed on the LPC residual. The transfer function for the inverse LPC filter is expressed as follows,

$$H^{-1}(z) = 1 - \sum_{j=1}^{p} \alpha_j z^{-j} \tag{5.14}$$

where the LPC coefficients, α_j, are determined through LPC analysis. The purpose of inverse filtering is twofold. First, by removing the short-term correlations between speech samples, the inverse filter generates an LPC residual with clearly visible long-term correlations. This enables better pitch prediction [13]. Secondly, for speech coders that only perform LPC and no pitch prediction, the advantage of quantizing the LPC residual for transmission, as opposed to the original signal, is a reduction in bit rate. For instance, waveform coders such as PCM or ADPCM require many bits for encoding given the original waveform's large dynamic range. However, by passing the input through an inverse LPC filter to remove the short-term correlations, a residual signal is generated with a smaller dynamic range which thus requires fewer bits to encode.

Assuming no pitch prediction is performed, the LPC residual is then passed through the pitch analysis filter, $A_p(z)$, to determine the optimal pitch lag and predictor coefficients as described in subsection 5.4.2. AAS coders typically employ an open-loop procedure for pitch analysis. In Open-Loop pitch analysis, the pitch parameters are determined by finding the values of M and β_k that minimize the mean-squared error between the original LPC residual and predicted LPC residual [10, 11, 22]. After the optimal pitch lag and predictor coefficients are determined, they are quantized and transmitted to the decoder.

Inverse Pitch Filtering Like short-term analysis which requires passing the original signal through an inverse LPC filter, long-term analysis involves passing the LPC residual through an inverse pitch synthesis filter, $P^{-1}(z)$, to remove the long-term correlations between pitch pulses. The transfer function for the inverse pitch filter, $P^{-1}(z)$ is expressed as follows,

$$P^{-1}(z) = 1 - \sum_{k=-1}^{i} \beta_k z^{-(M+k)} \tag{5.15}$$

The output of $P^{-1}(z)$ is a pitch or PPC residual signal whose short-term and long-term correlations have each been respectively removed by inverse LPC filtering and inverse pitch filtering. Along with the LSP and pitch parameters, the pitch residual is quantized and transmitted to the decoder. Clearly, by removing the long-term correlations from the LPC residual, the dynamic range of the resulting pitch residual is further reduced which leads to an even greater reduction in the number of bits required to encode the signal.

AAS Decoder

At the AAS decoder, the PPC parameters and LPC parameters are decoded to generate the pitch synthesis filter, $P(z)$, and LPC synthesis filter, $H(z)$, respectively. The linear combination of these filters thus represents the synthesis filter used in the theoretical model of speech production. The decoded pitch residual is first passed through $P(z)$ to generate the LPC residual which is then input to $H(z)$ to generate the synthesized speech.

5.6 Analysis-by-Synthesis (ABS) Coding of Speech

The AAS-LPC techniques presented thus far assume a theoretical model of speech production in which the excitation is either a periodic impulse train for voiced speech or random noise for unvoiced speech. Although AAS-LPC strategies lead to high quality speech coders between 8 kb/s and 16 kb/s, the simplistic nature of the excitation results in synthetic-sounding speech for coding at 8 kb/s and below [11].

The are two reasons for this degradation in speech quality. First, the encoded speech is not analyzed to see if the coding procedure is operating efficiently. This leads to little control over the distortions of the synthesized speech. Secondly, in adaptive coding schemes, errors that accumulate from previous frames are not considered in the current frame of analysis; hence, these errors may propagate into following frames without any form or resetting [11].

One of the key issues in designing high quality low bit rate speech coders is to represent the residual signal or excitation as efficiently as possible. One method proposed by Atal [10, 11, 22], is known as analysis-by-synthesis LPC (ABS-LPC) coding. In this strategy, attempts are made to quantize the residual signal which minimizes the error between the original and locally synthesized speech, using a closed-loop optimization procedure. Since the theoretical model of speech production requires a set of parmaters for synthesis, ABS-LPC performs both analysis and synthesis at the encoder to yield the optimal set of parameters for matching the original signal. Figure 5-4 illustrates an ABS-LPC speech coding system. The following subsections describe the ABS-LPC algorithm and three important blocks of an ABS-LPC system: (1) the time-varying filter, (2) the perceptual weighting filter, and (3) the excitation signal.

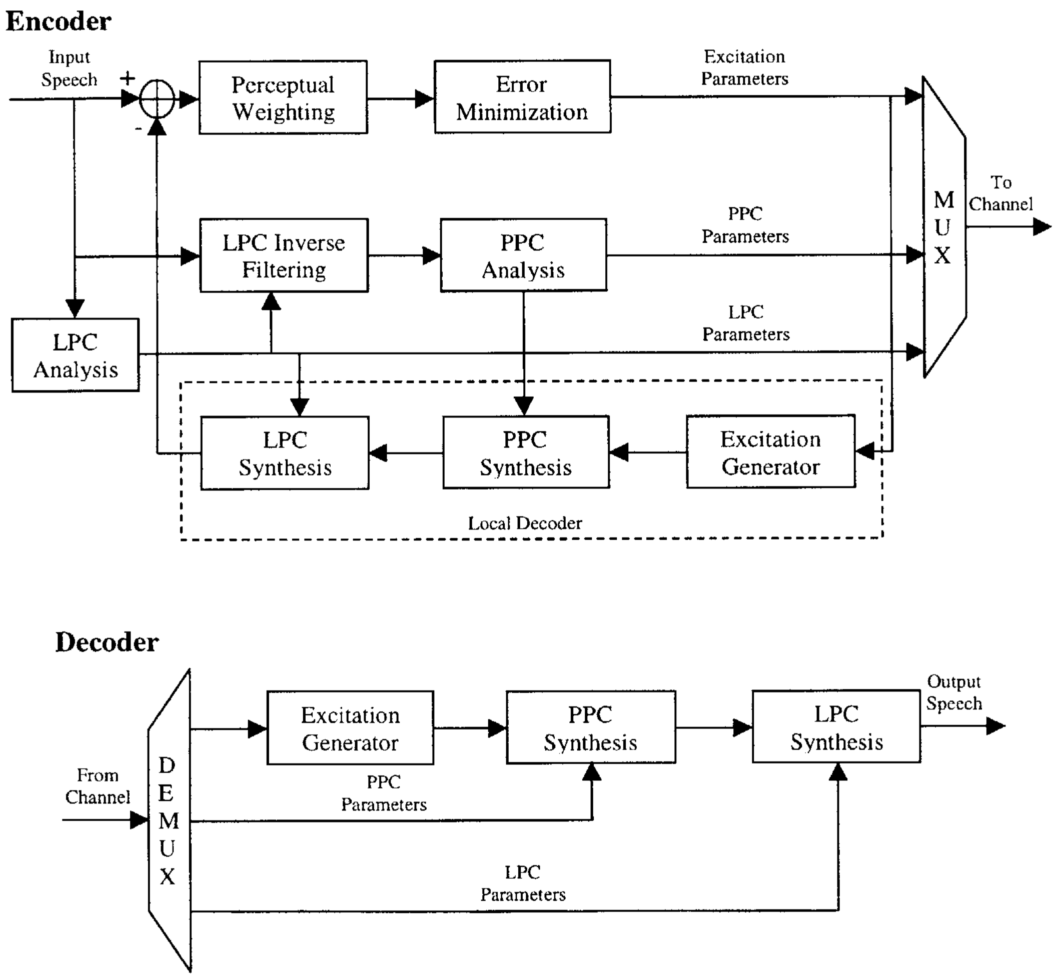

Figure 5-4: Analysis-by-Synthesis Speech Coding System.

5.6.1 The ABS-LPC Algorithm

The basic algorithm for an ABS-LPC system is described as follows [10, 11]:

1. Initialize the contents of the LPC and pitch synthesis filters to pre-determined values.
2. Buffer a frame of speech samples and perform LPC analysis for that frame.
3. Based on the LPC coefficients computed in step (2), calculate the first unquantized LPC residual by passing the original signal through the inverse LPC filter.
4. As the LPC residual frame is usually too large to efficiently determine the excitation, divide the frame into an integer number of subframes.
5. For each subframe, performing the following:
   (a) Calculate the pitch parameters using the LPC residual from inverse LPC filtering.
   (b) Generate a local pitch and LPC synthesis filter (combined to form a cascaded filter) based on the pitch parameters computed in step (5a) and LPC parameters computed in step (2).
   (c) Use the cascaded filter to determine an optimal excitation that minimizes the error between the locally synthesized speech and original speech.
6. At the receiver, generate the synthesized speech by passing the optimal secondary excitation through the cascaded filter with all the initial memory contents of the two filters, left over the from the previous synthesis frame, restored.
7. Repeat steps (2) through (6) for subsequent frames.

Although ABS-LPC speech coding employs a very similar model of speech production used by traditional vocoders, there is one major difference between them. In vocoders, the excitation signal is represented by either a periodic impulse train for voiced speech or random noise for unvoiced speech. In ABS-LPC schemes however, this categorization is not explicit and the excitation signal can range from pulse-like to noise-like in characteristic. Thus, the ability for ABS-LPC coders to support intermediate voicing states results in better synthesized speech quality [11].

5.6.2 Time-Varying Filter

In Figure 5-4, the time-varying filter is represented by the cascade of the LPC synthesis filter, $H(z)$, and pitch synthesis filter, $P(z)$. Since these filters are recursive in nature, each contains memory in its working buffer from analysis of the previous subframe. Inclusion and preservation of this filter memory is very important since it reflects the past history of the analysis and includes any errors incurred in previous subframes. Thus, the encoder and decoder each generate a copy of the synthetic speech in order to update the memory contents of the time-varying filter so that replica conditions are contained in both filter memories [10, 11].

As described in subsection 5.4.2, ABS-LPC schemes often use a closed-loop procedure for pitch estimation. In Closed-Loop pitch analysis, instead of minimizing the pitch residual (i.e., the error between the LPC residual and predicted LPC residual), the error between the original and locally synthesized speech is minimized [10, 11, 22]. This is achieved by passing the predicted LPC residual through a local LPC synthesis filter and comparing the synthesized speech with the original speech. Thus, the optimal pitch parameters, M and β_k, are those which minimize the mean-squared error between the locally synthesized and original speech.

Closed-Loop pitch analysis achieves better performance over Open-Loop pitch analysis since it considers the joint performance of the vocal tract and glottal excitation. In addition, any errors generated during the quantization of the LPC and pitch parameters are considered when the speech is synthesized at the decoder [11].

5.6.3 Perceptual Weighting Filter

In Figure 5-4, the error minimization block minimizes the error between the original and locally synthesized speech according to a suitable error criterion. Although this optimization procedure commonly minimizes the mean-squared error, this offers only adequate performance and degrades at lower bit rates. To take the role of human perception into account, which is necessary for improved speech quality at lower bit rates, ABS-LPC schemes introduce a perceptual weighting filter before error minimization [10, 11, 22].

It has been shown that the human ear is less sensitive to noise in higher energy regions of the spectrum, such as the formants, than in lower energy regions, such as the valleys

[22]. This enables perception of some speech signals to be masked away by higher energy signals. Since the error between the original and synthesized speech tends to have a flat spectrum, when the energy of this residual exceeds that of the speech spectrum, especially in the valleys, this becomes annoying to the ear. To overcome this problem, the residual is passed through a linear perceptual weighting filter to increase its energy in the formant regions and to decrease its energy in the valley regions [22]. The transfer function, $W(z)$, for this perceptual weighting filter is expressed as follows,

$$W(z) = \frac{1 - \sum_{j=1}^{p} \alpha_j \gamma^j z^{-j}}{1 - \sum_{j=1}^{p} \alpha_j \delta^j z^{-j}} \tag{5.16}$$

where α_j are the LPC predictor coefficients, and γ and δ are fractions between 0 and 1 that control the increase of the residual in the formant regions. The values of these weighting factors are determined by suitable listening tests [22]. Overall, the effect of the perceptual weighting filter is to broaden the bandwidth of the formants, thereby deemphasizing the frequency regions corresponding to the formants and emphasizing the frequency regions corresponding to the valleys.

5.6.4 Excitation Signal

The excitation represents the input to the ABS-LPC system and is the most important block in Figure 5-4. This is because the excitation source contains any residual structures that are not represented by the spectral model of the time-varying filters. These include pitch dependent structures with significant correlations that are not covered by pitch analysis or any random structures that cannot be efficiently modeled by deterministic methods [10]. The excitation can thus range from pure pulses to random noise and is expressed as follows,

$$\mathbf{U}_i = \mathbf{g}_i \mathbf{X}_i \tag{5.17}$$

where $\mathbf{U}_i$ is an L-dimensional i^{th} excitation vector, $\mathbf{X}_i$ is an $M \times L$ matrix representing the shape of the excitation, and $\mathbf{g}_i$ is the M-dimensional gain or scale vector associated with shape, $\mathbf{X}_i$. By appropriately selecting various shape vectors that reflect the underlying statistics of the excitation, the excitation can appear in any form. These shapes include

Multi-Pulse, Regular-Pulse, Codebook-Excitation, and Self-Excitation. Since the ITU-T G.723.1 dual rate speech coder, which is used for packet recovery optimization in Chapter 6, employs Multi-Pulse and Codebook-Excitation, a introduction to these two excitation types is presented below.

Codebook-Excitation

In Codebook-Excited LPC (CELP), the excitation vector is chosen from a set of pre-stored collection of C possible stochastic sequences with an associated scaling or gain vector. In mathematical terms, this requires dividing an L-dimensional code-vector, $\mathbf{c}_k$, into M parts, each of length L/M, and computing the optimal gains, $g_{k1}, ..., g_{kM}$, for for each index k, $1 \leq k \leq C$. Thus, for Codebook-Excitation, the i^{th} excitation, $\mathbf{U}_i$, is expressed as follows,

$$\mathbf{U}_i = \mathbf{g}_i \mathbf{X}_i, \quad i = 1, ..., C \tag{5.18}$$

where

$$\mathbf{g}_i = [g^j_{i1}, ...g^j_{iM}], \quad j = 1, ..., L/M \tag{5.19}$$

$$\mathbf{X}_i = \begin{cases} c^k_j & \text{for } j = 0, 1, ..., L-1 \\ 0 & \text{otherwise} \end{cases} \tag{5.20}$$

Since the C-level codebook is of finite dimension, it must be populated with representative excitation vectors. Among the possible codebook types include Gaussian, Sparse or Center Clipped, Binary, Ternary, Overlapped, and Fractals as described in [22].

In ABS-LPC techniques, each of the C code-vectors is passed through the time-varying filter, and the code-vector that results in the smallest residual between the original and synthesized speech is the desired code-vector. Since the codebook is present at both the encoder and decoder, only an index value matching the desired code-vector is required for transmission. Thus, less than 1 bit per sample is achievable for transmission [10].

Multi-Pulse Excitation

Multi-Pulse LPC (MPLPC) was the first of the ABS-LPC coding schemes. In MPLPC, the rigorous division between voiced and unvoiced classes is avoided by making no prior assumptions about the nature of the excitation signal. In MPLPC, rather than selecting an optimal stochastic sequence from a codebook and transmitting its index, the excitation is specified by a small set of pulses with different amplitudes located at non-uniformly spaced intervals [10, 11, 22]. MPLPC involves determining the pulse positions and amplitudes of the excitation that result in the minimum residual signal and transmitting the pulse positions and amplitudes to the decoder. Thus, the only a priori information is the number of pulses required per analysis block. The representation for MPLPC can be defined for a given set of M-pulse locations, n_i, with the $M \times L$ matrix, $\mathbf{X}_M$, as follows,

$$\mathbf{U}_M = \mathbf{g}_M \mathbf{X}_M \tag{5.21}$$

$$\mathbf{g}_M = [g_{M1}, \ldots g_{nM}] \tag{5.22}$$

$$\mathbf{X}_M = \begin{cases} 1 & \text{for } j = n_{i+1}, \quad 0 \le i \le M-1 \\ 0 & \text{otherwise,} \qquad 0 \le j < L \end{cases} \tag{5.23}$$

From the above formulation, MPLPC can actually be viewed as a CELP system with a very large codebook, the size of which is determined by the number of pulses and the number of bits used to quantize the pulse positions and pulse amplitudes [10].

Chapter 6

Packet Recovery Optimization

As presented in Chapter 4, RTP was determined to be the optimal protocol for maximizing speech quality and minimizing end-to-end delay. One of the important features of RTP is the inclusion of sequence numbers which allows the application layer to detect frame erasures and to perform lost-frame recovery if necessary. Given this mechanism for reliability and by applying the speech coding concepts described in Chapter 5, one of this thesis' main contributions is presented in this chapter.

First, the performance of G.723.1 error concealment is assessed for randomly distributed packet loss rates up to 15%. Based on a set of qualitative results, an improved lost-frame recovery technique is then developed in this thesis to eliminate the problems associated with G.723.1 error concealment. This technique combines three original algorithms — linear interpolation, selective energy attenuation, and energy tapering — and is shown to achieve up to a 20% increase in speech quality over G.723.1 error concealment and to be preferred by 66% of the tested population. The following sections describe the conception and operation of the improved lost-frame recovery technique and the methods and results from the informal listening tests.

6.1 ITU-T G.723.1 Dual Rate Speech Coder

The G.723.1 dual rate speech coder operates at 5.3 kb/s and 6.3 kb/s and was ratified by the ITU in 1996 for application in low bit rate videophone systems. It has since been used, in part, to encode voice for multimedia services over packet-switching networks, including IP, ATM, and Frame Relay, as well as over mobile communications networks. With a mean opinion score of 3.98 out of 5 [26], the coder's near toll quality is highly successful for voice over LANs where packet loss is minimal. However, over WANs, global area networks (GANs), and mobile communications networks, congestion can be severe, and packet loss often leads to degradations in speech quality if left untreated. To maintain an acceptable degree of speech quality, the decoder must implement a method of error concealment to mask missing packets. The following sections describe the operation of the G.723.1 encoder and decoder, as well as the G.723.1 error concealment strategy.

6.1.1 Encoder Operation

The G.723.1 dual rate speech coder encodes 16-bit linear PCM speech, sampled at a rate of 8 kHz, using ABS-LPC coding as described in Chapter 5. The excitation for the high rate coder is Multi-pulse Maximum Likelihood Quantization (MP-MLQ), while the excitation for the low rate coder is Algebraic Code-excited Linear Prediction (ACELP). The encoder operates on a 30-ms frame size, equivalent to a frame length of 240 samples, and divides every frame into four subframes of 60 samples each [9].

For every subframe, a tenth-order LPC filter is computed; the LPC coefficients for every last subframe are converted into LSP parameters and quantized using Predictive Split Vector Quantization (PSVQ). Based on the LPC coefficients, a short-term perceptual weighting filter is generated and the Open-Loop pitch period is estimated. From this point on, processing occurs on a subframe by subframe basis [9].

For every subframe, a harmonic noise shaping filter is constructed from the Open-Loop pitch period and combined with the LPC synthesis and perceptual weighting filters to generate an impulse response. The Open-Loop pitch and impulse response are then used to compute a fifth-order, Closed-Loop pitch predictor, or adaptive codebook contribution. Finally, the excitation signal, or fixed codebook contribution, is approximated using either MP-MLQ for the high-rate coder or ACELP for the low-rate coder. The resulting bit-stream sent from encoder to decoder consists of the LSP parameters, adaptive codebook pitch lags, fixed and adaptive codebook gains, pulse positions, pulse signs, and the grid index [9]. This thesis assumes operation of the G.723.1 high-rate 6.3 kb/s coder unless stated otherwise.

6.1.2 Decoder Operation

At the G.723.1 decoder illustrated in Figure 6-1, the LSP parameters are decoded and the LPC synthesis filter is generated. For every subframe, the fixed and adaptive codebook contributions are added together and the resulting LPC residual is input to a pitch postfilter where the output of pitch postfiltering is sent to the LPC synthesis filter. The output of LPC synthesis is then input to a formant postfilter and gain scaling unit to generate the synthesized speech. In the case of indicated frame erasures, an error concealment strategy is provided [9].

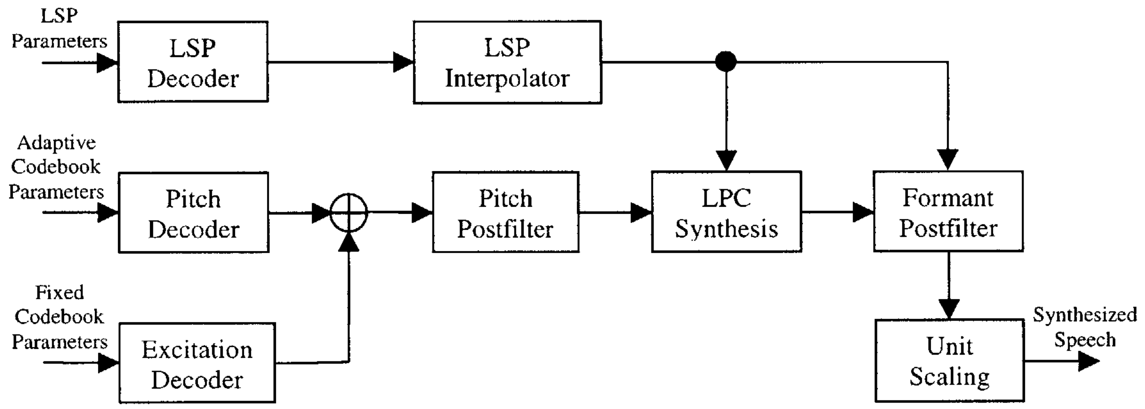

Figure 6-1: G.723.1 Decoder Operation.

6.1.3 G.723.1 Error Concealment

In the presence packet of loss, the G.723.1 dual rate speech coder provides an error concealment strategy to recover missing packets. This strategy consists of two steps: (1) LSP vector recovery and (2) excitation recovery [9]. Each of these steps is described below.

LSP Vector Recovery

The missing frame's LSP vector is recovered by applying a fixed linear predictor to the previously decoded LSP vector [9].

Excitation Recovery

The missing frame's excitation is recovered using only the recent information available at the decoder. This is achieved by first determining the previous frame's voiced/unvoiced classifier using a cross-correlation maximization function and then testing the prediction gain for the best vector. If the gain is more than 0.58 dB, the frame is declared voiced; otherwise, the frame is declared unvoiced. The classifier returns a value of 0 if the previous frame is unvoiced or the estimated pitch lag if the previous frame is voiced [9].

For unvoiced speech, the missing frame's excitation is generated using a uniform random number generator and scaled by the average of the gains for subframes 2 and 3 of the previous frame. For voiced speech, the previous frame is attenuated by 2.5 dB and regenerated with a periodic excitation having a period equal to the estimated pitch lag. If packet losses continue for the next two frames, the regenerated excitation is attenuated by an additional 2.5 dB for each frame; however, after three interpolated frames, the output is completely muted [9].

6.2 G.723.1 Error Concealment Simulations

To assess the performance of G.723.1 error concealment, five simulations were performed in which a 3200-frame speech segment was injected with five different rates of packet loss and recovered using G.723.1 error concealment. Informal listening tests were then conducted by nine participants who were asked to qualitatively identify distortions present in the G.723.1 error concealed segments. The following subsections describe the simulation environments, methods, and results.

6.2.1 Environments

Each simulation environment consisted of a network and one of the following levels of congestion: (1) Low, (2) Low-Middle, (3) Middle, (4) Middle-High, and (5) High. Each congestion level corresponded to a rate of fixed packet loss, FPL, which represented the percentage of packets to experience discard. For low congestion, $FPL = 1\%$. For low-middle congestion, $FPL = 3\%$. For middle, middle-high, and high congestion levels, the fixed packet loss rates were 6%, 10%, and 15% respectively.

6.2.2 Methods

First, a 3200-frame speech segment was transmitted across each congested network. The 3200-frame segment consisted of 24 male-spoken English sentences and was encoded using G.723.1 voice compression. To simulate the corresponding rate of packet loss for each network, a random number generator was applied to the 3200-frame segment to randomly select ($\frac{FPL}{100} \times 3200$) packets to discard. Table 6.1 summarizes the distribution of missing packets simulated for each rate.

Consecutive Losses	Congestion Level				
	Low	Low-Middle	Middle	Middle-High	High
1	32	92	168	275	362
2	0	2	9	18	45
3	0	0	2	3	5
4	0	0	0	0	2
5	0	0	0	0	1
Total Losses	32	92	192	275	480

Table 6.1: Distribution of Fixed Packet Loss.

After simulating packet loss for the 3200-frame segment, G.723.1 error concealment was applied on the corrupted signal to generate a G.723.1 error concealed segment. This process was performed for all rates of packet loss resulting in five G.723.1 error concealed speech segments. Informal listening tests were then conducted by nine participants (six male and three female) to assess the performance of G.723.1 error concealment for all packet loss rates.

Each participant was brought into a listening room, equipped with a pair of headphones, and played the entire duration of each G.723.1 error concealed segment. After each speech segment playout, the listener was asked to qualitatively identify any distortions that led to discomfort and quality degradations in the synthesized speech.

6.2.3 Results

Based on the unified results from the nine listeners, it was discovered that the following problems persisted at all rates and became increasingly severe as packet loss increased: (1) unnatural-sounding speech, (2) metallic-sounding artifacts, (3) high-energy spikes, and (4) choppy speech. Each of these problems and their causes is described below.

Unnatural-Sounding Speech

First, many parts of the synthesized speech were found to sound unnatural or synthetic. The robotic-sounding quality of the synthesized speech is attributed to LSP vector recovery based on fixed prediction, as described in subsection 6.1.3. Since the missing frame's LSP vector is recovered by applying a fixed predictor to the previous frame's LSP vector, spectral changes across previous and recovered frames are not as accurate and as smooth as if interpolation-based recovery, for instance, were applied instead. Thus, failure to generate smooth spectral changes across missing frames results in a more synthetic-sounding quality, which leads to decreased speech quality during higher rates of packet loss.

Metallic-Sounding Artifacts

Another problem resulting from G.723.1 error concealment was occurrence of high-frequency, metallic-sounding artifacts. These artifacts, which primarily occur in unvoiced regions of the synthesized speech, are caused by inaccurate voicing estimation of the previous frame during excitation recovery, as described in subsection 6.1.3. Since missing, unvoiced frames may be incorrectly classified as voiced, transition into these frames generates high-frequency glitches or metallic-sounding artifacts when the incorrectly estimated pitch lag is used in excitation recovery. As packet loss rates increase, this problem intensifies as a greater occurrence of metallic-sounding artifacts decreases speech quality.

High-Energy Spikes

High-Energy spikes were also heard in the synthesized speech. These high-energy spikes, which are extremely uncomfortable for the ear, are caused either by poor prediction of the missing frame's gain or LSP parameters, the latter of which is used in the formant postfilter and LPC synthesis filter, as shown in Figure 6-1. As packet loss rates increase, the number of high-energy spikes also increases, leading to greater listener discomfort and additional speech distortion.

Choppy Speech

Finally, choppy speech, resulting from partial or complete muting of the output was also evident. As described in subsection 6.1.3, partial output muting occurs when consecutive missing packets are increasingly attenuated by 2.5 dB, thereby leading to a sharp signal energy decrease over multiple lost packets that results in the synthesized speech to sound choppy to the listener. Complete output muting occurs when G.723.1 error concealment recovers only three consecutive missing packets and mutes subsequent missing packets. This also leads to choppy speech as segments of complete silence are generated in the synthesized speech. Since there is a greater probability that more than three consecutive packets may be lost in a network, especially during higher levels of congestion, this results in even more instances of choppy speech that further degrades speech quality.

6.3 An Improved Lost-Frame Recovery (ILFR) Technique

To eliminate the problems associated with G.723.1 error concealment, an improved lost-frame recovery (ILFR) technique was developed in this thesis. Figure 6-2 illustrates a block diagram of the ILFR technique implemented at the G.723.1 decoder.

As shown in Figure 6-2, ILFR combines a linear interpolation (LI), selective energy attenuation (SEA), and energy tapering (ET) algorithm, each of which is responsible for eliminating a particular problem associated with G.723.1 error concealment. In addition, the following 30-ms buffers are introduced at the decoder: (1) Future Buffer, (2) Ready Buffer, and (3) Copy Buffer. Before describing this technique, it is first necessary to define the following terms as applied to ILFR:

1. **previous frame**, refers to the last good frame that was processed by the decoder and is stored in the Copy Buffer.

2. **current frame**, refers to a good or missing frame that is currently being processed by the decoder and is stored in the Ready Buffer.

3. **future frame**, refers to a good or missing frame immediately following the current frame and is stored in the Future Buffer.

Two important issues must be stressed at this point. First, although packets can arrive both disordered and missing at the receiver, the following discussion assumes no out-of-order packets so that the performance of ILFR, in the presence of packet loss alone, can be singularly assessed. In Chapter 7, a frame-resequencing technique is introduced at the decoder to resequence disordered packets at the receiver. Secondly, since ILFR exists at

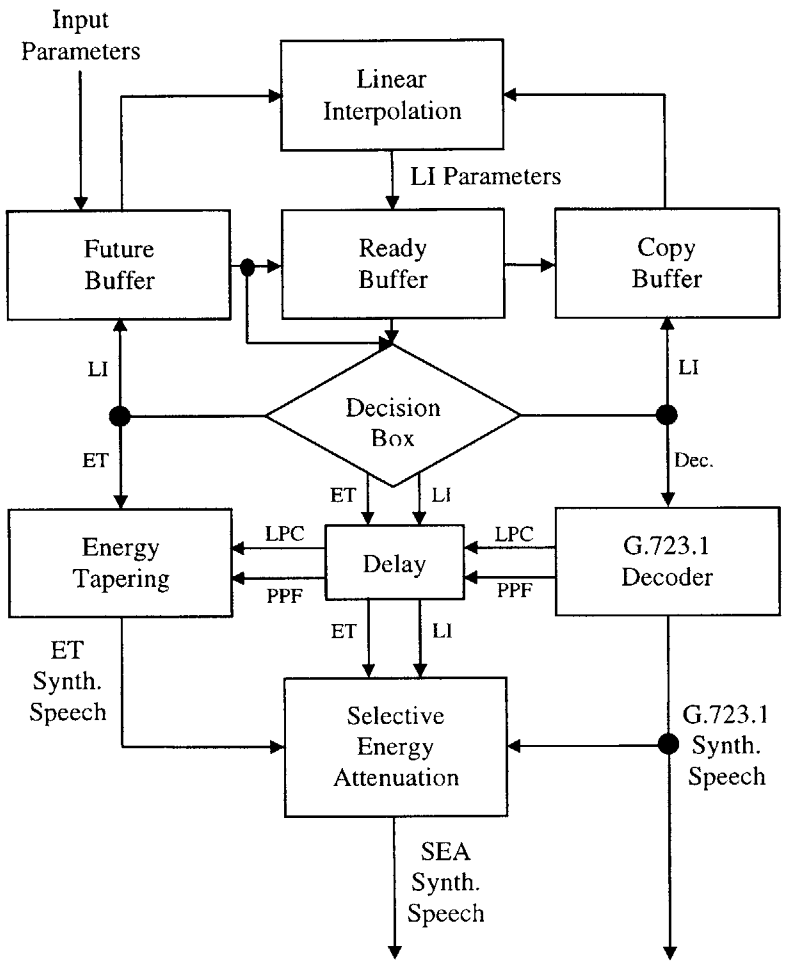

Figure 6-2: Improved Lost-Frame Recovery Technique.

the application layer, the information it receives from RTP is sequentially-numbered speech frames. The following subsections describe the three algorithms of the ILFR technique.

6.3.1 Linear Interpolation

Linear interpolation of the speech model parameters was developed in this thesis to smooth spectral changes across a single frame erasure (i.e. a missing frame in between two good speech frames) and hence, generate more natural-sounding speech while eliminating metallic-sounding artifacts from the synthesized speech. Linear interpolation is a multi-step procedure that operates as follows:

1. Store the current good frame to be processed in the Ready Buffer and the future frame of the encoded speech sequence, either good or missing, in the Future Buffer; create a copy of the current frame's speech model parameters and store this copy in the Copy Buffer.

2. Determine the status of the future frame, either good or missing, by examining the future frame's sequence number:

   (a) If the future frame's sequence number is one greater than the current frame's sequence number, the future frame is good; no linear interpolation is necessary; the linear interpolation flag is reset to 0.

   (b) If the future frame's sequence number is not one greater than the current frame's sequence number, the future frame is missing; linear interpolation might be necessary; the linear interpolation flag is temporarily set to 1.

3. Decode and synthesize the current frame; create a copy of the current frame's LPC synthesis filter and pitch postfiltered excitation.

4. Copy the contents of the Future Buffer into the Ready Buffer; store the next frame in the encoded speech sequence, either good or missing, in the Future Buffer.

5. Check the value of the linear interpolation flag:

   (a) If the flag is set to 0, jump to step (2).

   (b) If the flag is set to 1, jump to step (6).

6. Determine the status of the future frame:

   (a) If the future frame is good, linear interpolation is applied; the linear interpolation flag remains set to 1; jump to step (7).

   (b) If the future frame is missing, energy tapering is applied; the energy tapering flag is set to 1 and the linear interpolation flag is reset to 0.

   (Note that the energy tapering algorithm is applied only for multiple frame losses and is described in section 6.3.3.)

7. Perform LSP recovery by averaging the tenth-order LSP vectors from the previous and future good frames, stored in the Copy and Future Buffers respectively, to obtain the interpolated tenth-order LSP vector for the current frame.

8. Perform excitation recovery as follows:

   (a) Average the four fixed codebook gains from each of the four subframes of the previous frame to yield $\overline{\Gamma}_{n-1}$.

   (b) Average the four fixed codebook gains from each of the four subframes of the future frame to yield $\overline{\Gamma}_{n+1}$.

   (c) Interpolate the fixed codebook gains, Γ_{nj}, $0 \leq j \leq 3$, for each of the four subframes of the missing frame using the following weighting formula,

$$\Gamma_{nj} = \left(\frac{4-j}{5}\right)\overline{\Gamma}_{n-1} + \left(\frac{1+j}{5}\right)\overline{\Gamma}_{n+1} \tag{6.1}$$

   Assume all other speech model parameters from the previous frame.

9. Perform pitch lag and predictor gain estimations for the previous frame, stored in the Copy Buffer, using the identical technique described for G.723.1 error concealment, as described in subsection 6.1.3.

10. Determine the voicing classification for the previous frame:

    (a) If the predictor gain is less than α dB, declare the previous frame unvoiced; generate the excitation signal for the current frame by using a random number generator scaled by the previously calculated interpolated fixed codebook gain in step (8c).

    (b) If the predictor gain is greater than α dB and the estimated pitch lag exceeds a threshold value P_{thresh}, declare the previous frame voiced; generate the excitation signal for the current frame by first attenuating the previous frame's excitation by γ dB for every two subframes, and then regenerating this excitation with a period equal to the estimated pitch lag.

    (c) Otherwise, if the predictor gain is greater than α dB and the estimated pitch lag is less than P_{thresh}, declare the current frame unvoiced and recover the excitation as described in (10a).

11. Decode and synthesize the interpolated frame.

12. Copy the contents of the Future Buffer into the Ready Buffer; store the next frame in the encoded speech sequence, either good or missing, in the Future Buffer; jump back to step (2).

6.3.2 Selective Energy Attenuation

Selective energy attenuation was developed in this thesis to eliminate high-energy spikes in the synthesized speech. These high-energy spikes, which are extremely uncomfortable for the ear, are caused either by poor prediction of the missing frame's gain or LSP parameters, the latter of which is used in the form of LPC coefficients during the formant postfiltering and LPC synthesis stages. To provide more accurate estimates for the missing frame's LSP parameters, linear interpolation was developed to recover the missing frame's LSP vector. To solve the gain problem and eliminate high-energy spikes, a selective energy attenuation algorithm was designed which operates as follows: After formant postfiltering, the signal energy for every synthesized subframe is checked against a threshold energy, S_{thresh}. If the signal energy for any one of the four subframes exceeds S_{thresh}, then the signal energies for all remaining subframes are attenuated to an acceptable energy level, S_{max}.

6.3.3 Energy Tapering

Finally, an energy tapering algorithm was designed in this thesis to eliminate the effects of choppy speech evident from G.723.1 error concealment. As described in subsection 6.2.3, choppy speech results when the signal energy of the output is increasingly attenuated by 2.5 dB for consecutive missing frames, thereby generating regions of partial output muting, or when the output is completely muted after three missing frames are recovered. In both cases, segments of low signal energy or complete silence are generated, resulting in choppy synthesized speech that decreases the overall speech quality. To eliminate this problem, a multi-step energy tapering algorithm was developed in this thesis. This energy tapering algorithm operates as follows:

1. Store the current good frame to be processed in the Ready Buffer and the future frame of the encoded speech sequence, either good or missing, in the Future Buffer; create a

copy of the current frame's speech model parameters and store this copy in the Copy Buffer.

2. Determine the status of the future frame, either good or missing, by examining the future frame's sequence number:
   (a) If the future frame's sequence number is one greater than the current frame's sequence number, the future packet is good; no linear interpolation is necessary; the linear interpolation flag is reset to 0.
   (b) If the future frame's sequence number if not one greater than the current frame's sequence number, the future frame is missing; linear interpolation might be necessary; the linear interpolation flag is temporarily set to 1.
3. Decode and synthesize the current frame; create a copy of the current frame's LPC synthesis filter and pitch postfiltered excitation.
4. Copy the contents of the Future Buffer into the Ready Buffer; store the next frame in the encoded speech sequence, either good or missing, in the Future Buffer.
5. Check the value of the linear interpolation flag:
   (a) If the flag in set to 0, jump to step (2).
   (b) If the flag is set to 1, jump to step (6).
6. Determine the status of the future frame:
   (a) If the future frame is good, linear interpolation is applied as described in 6.3.1.
   (b) If the future frame is missing, energy tapering is applied; the energy tapering flag is set to 1 and the linear interpolation flag is reset to 0; jump to step (7).
7. Attenuate the copy of the previous frame's pitch postfiltered excitation signal, from step (4), by ($\delta \times$ value of energy tapering flag) dB.
8. Synthesize the current frame using the copy of the previous frame's LPC synthesis filter, from step (3), and the attenuated excitation from step (7).
9. Copy the contents of the Future Buffer into the Ready Buffer; store the next frame in the encoded speech sequence, either good or missing, in the Future Buffer.

10. Determine the status of the future frame:

    (a) If the future frame is good, perform final energy tapering; increment the value of the energy tapering flag by 1; synthesize the current frame using steps (7) and (8); jump to step (11).

    (b) If the future frame is missing, continue energy tapering; increment the value of the energy tapering by 1; synthesize the current frame using steps (7) to (9); jump to step (10).

11. Copy the contents of the Future Buffer into the Ready Buffer; store the next frame in the encoded speech sequence, either good or missing, in the Future Buffer; jump back to step (2).

6.4 Advantages of Improved Lost-Frame Recovery

The individual advantages of linear interpolation, selective energy attenuation, and energy tapering theoretically enable ILFR to generate better speech quality over G.723.1 error concealment. In this section, these advantages are described; in section 6.5, a comparison of their performances based on informal listening tests is presented.

6.4.1 Advantages of Linear Interpolation

There are three important advantages of linear interpolation which allow it to achieve better speech quality over G.723.1 error concealment: (1) improved LSP vector recovery, (2) improved fixed codebook gain recovery, and (3) improved voicing estimation. These advantages are described below.

Improved LSP Recovery

The first advantage of linear interpolation over G.723.1 error concealment is improved LSP recovery. Since linear interpolation determines the missing frame's LSP parameters based on the previous and future speech frames, a better estimate is generated for the missing frame's LSP vector. This results in smoother and more accurate spectral changes across a missing frame than if fixed LSP prediction were simply used, as in G.723.1 error concealment. Thus, more natural-sounding speech is synthesized.

Improved Gain Recovery

The second advantage of linear interpolation over G.723.1 error concealment is improved gain recovery. As described in subsection 6.3.1, linear interpolation generates the missing frame's fixed codebook gains by weighting the gains between the previous and future frames. Thus, a better estimate is provided for the missing frame's gain, as opposed to simply averaging the gains from subframes 2 and 3 of the previous frame, as in G.723.1 error concealment. This interpolated gain, which is then applied for recovering unvoiced frames, results in more comfortable-sounding gain transitions across frame erasures.

Improved Voicing Estimation

The third advantage of linear interpolation over G.723.1 error concealment is improved voicing estimation during excitation recovery. As described in subsection 6.3.1, linear interpolation relies on both the predictor gain and estimated pitch lag for recovering the excitation signal, as opposed to the predictor gain alone. Specifically, a speech frame whose predictor gain is greater than α dB is also compared against a threshold pitch lag, P_{thresh}, to determine its voicing classification. Since unvoiced frames are primarily composed of high-frequency spectra, frames that have lower estimated pitch lags and hence higher estimated pitch frequencies, have a higher probability of being unvoiced. Thus, for missing frames whose predictor gain exceeds α dB, those whose pitch lag falls below P_{thresh} are declared unvoiced, while those frames whose estimated pitch lag exceeds P_{thresh} are declared voiced. By selectively determining a frame's voicing classification based on both the predictor gain and estimated pitch lag, this algorithm effectively masks away all occurrences of high-frequency, metallic-sounding artifacts occurring in the synthesized speech.

6.4.2 Advantages of Selective Energy Attenuation

The main advantage of selective energy attenuation over G.723.1 error concealment is the implementation of a gain-checking criterion that efficiently detects high-energy spikes. Specifically, the subframe gains for the pitch postfiltered signal are checked against a threshold gain, S_{thresh}, and attenuated to S_{max} if the threshold is exceeded. Combined with linear interpolation, this algorithm effectively eliminates all instance of high-energy spikes from the synthesized speech without introducing noticeable output degradation.

Figure 6-3 illustrates an uncorrupted G.723.1 speech segment. Figure 6-4 illustrates the same segment with a high-energy spike, due to G.723.1 error concealment; Figure 6-5 illustrates elimination of the high-energy spike due to selective energy attenuation. It should also be noted that the slight output muting in the segment shown in Figure 6-4 is eliminated by energy tapering, as shown in Figure 6-5.

6.4.3 Advantages of Energy Tapering

There are two important advantages of energy tapering over G.723.1 error concealment. The first improvement, choppy speech minimization, results in better synthesized speech quality, while the second improvement, complexity minimization, decreases end-to-end delay. Each of these advantages is described below.

Choppy Speech Minimization

The primary advantage of energy tapering that allows it to achieve better speech quality over G.723.1 error concealment is that the synthesized speech's signal energy is never completely muted over multiple frame erasures. Rather, the signal energy is gradually reduced or tapered by ($\delta \times$ value of energy tapering flag) dB over consecutive frame losses, thereby eliminating any choppy speech due to complete output muting.

Figure 6-7 illustrates the presence of extreme output muting in the uncorrupted speech segment of Figure 6-6, due to G.723.1 error concealment. Figure 6-8 illustrates elimination of complete output muting due to energy tapering.

Complexity Minimization

In addition to improving the synthesized speech quality, energy tapering requires a relatively smaller amount of computation time than G.723.1 error concealment. Since energy tapering synthesizes a missing frame by attenuating the previous frame's pitch postfiltered gain and sending this signal through a copy of its LPC synthesis filter, the total complexity and algorithmic delay is less than that compared to performing G.723.1 error concealment and decoding. Thus, end-to-end delay are minimized.

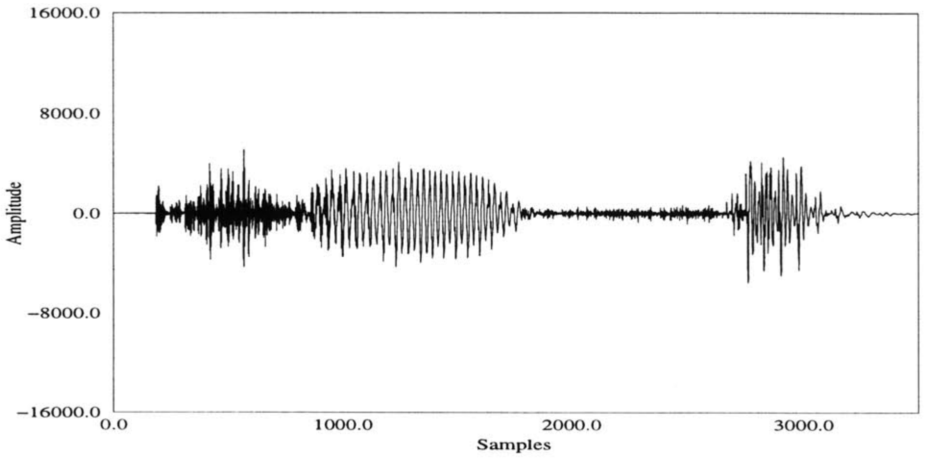

Figure 6-3: Uncorrupted G.723.1 Compressed Speech Segment.

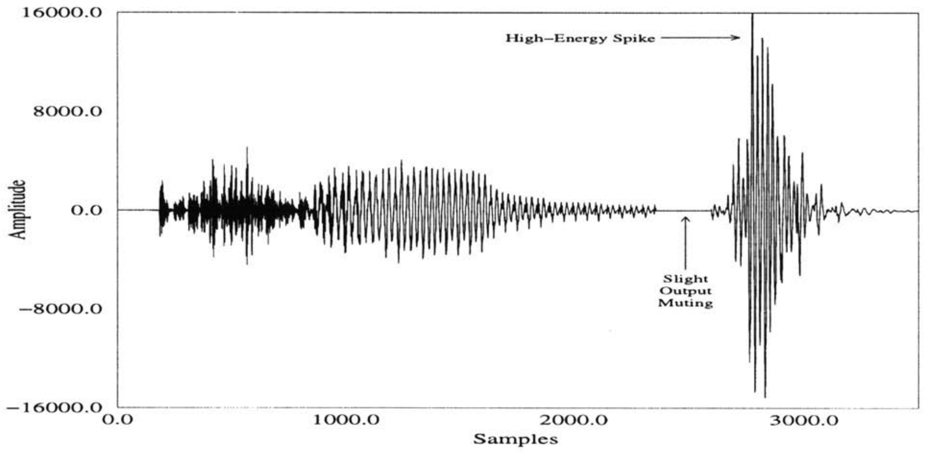

Figure 6-4: High-Energy Spike from G.723.1 Error Concealment.

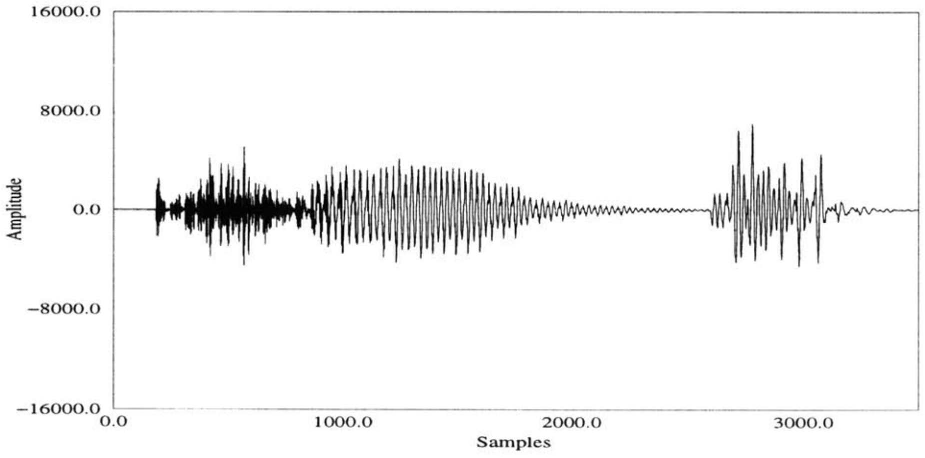

Figure 6-5: Elimination of High-Energy Spike.

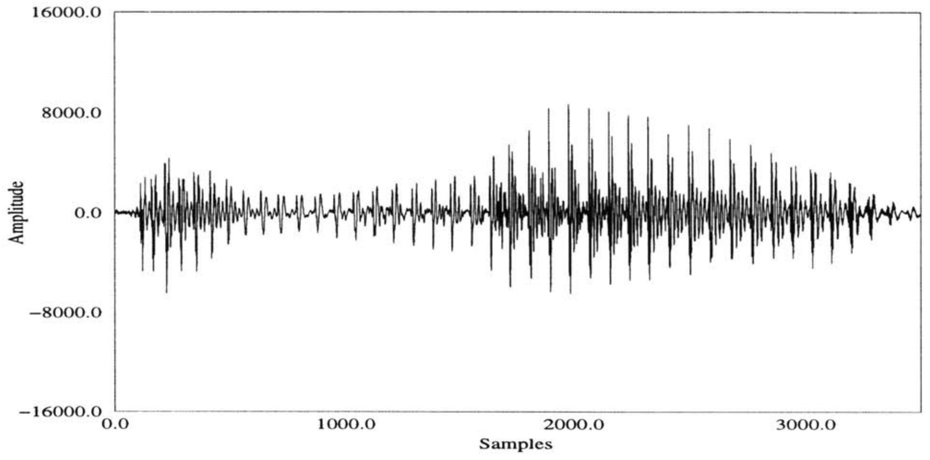

Figure 6-6: Uncorrupted G.723.1 Compressed Speech Segment.

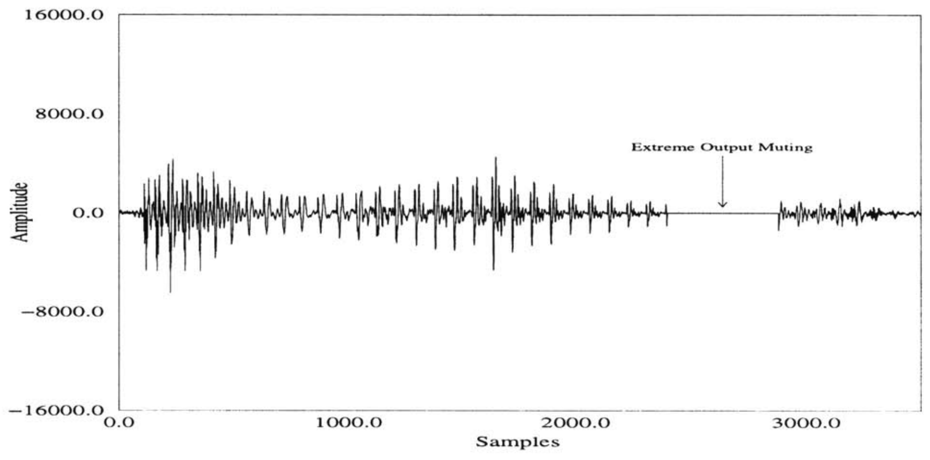

Figure 6-7: Choppy Speech from G.723.1 Error Concealment.

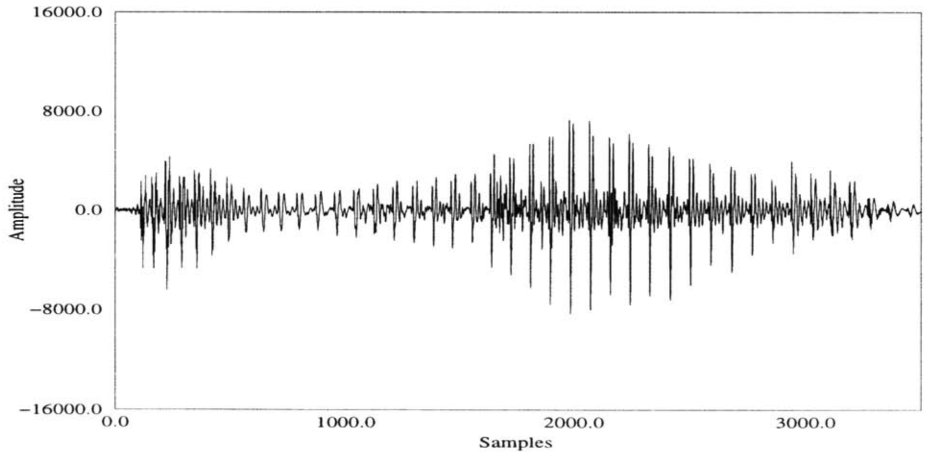

Figure 6-8: Elimination of Choppy Speech.

6.5 G.723.1 vs. Improved Lost-Frame Recovery Simulations

Although the advantages of ILFR theoretically enable it to achieve better speech quality over G.723.1 error concealment, subjective listener evaluations were necessary to truly determine the performances of both methods relative to one another. Five comparison simulations were thus performed in which an 800-frame speech segment was injected with five different rates of packet loss and recovered using both G.723.1 error concealment and ILFR. Informal listening tests were then conducted by the same participants in the first set of simulations to determine whether or not ILFR could subjectively outperform G.723.1 error concealment in the presence of packet loss and to what degree this was achievable. The following subsections describe the simulation environments, methods, and results.

6.5.1 Environments

Each simulation environment consisted of a congested network with the same rate of fixed packet loss as in the first set of simulations. For low congestion, the fixed packet loss rate was 1%. For low-middle congestion, the fixed packet loss rate was 3%. For middle, middle-high, and high congestion levels, the fixed packet loss rates were 6%, 10%, and 15% respectively.

6.5.2 Methods

First, an 800-frame speech segment consisting of six male-spoken English sentences was transmitted across each congested network. An 800-frame segment instead of a 3200-frame segment was used for these simulations since the informal listening tests would require listeners to remember how a previous synthesized segment sounded for comparing against the current synthesized segment. Hence, shorter speech segments were used so that participants could better remember the speech quality of the previous speech segment.

After randomly distributed packet loss was simulated for the 800-frame segment as described in subsection 6.2.2, G.723.1 error concealment and ILFR were separately applied on the corrupted signal to generate two types of synthesized segments: (1) a G.723.1 error concealed segment and (2) an ILFR segment. This process was performed for all rates of packet loss resulting in five G.723.1 error concealed segments and five ILFR segments. Informal listening tests were then conducted by the nine participants who were asked to subjectively score the speech quality of the two segment types.

Each listener was first played an uncorrupted G.723.1 sentence pair, with a de facto speech quality score of $\frac{50}{50}$. Then, the G.723.1 error concealed version followed by the ILFR version were played, and the listener was asked to subjectively score their speech qualities relative to the uncorrupted G.723.1 sentence pair, with 1 being the lowest possible score and 50 being the highest possible score. This process was performed for all ten synthesized speech segments simulated at all packet loss rates.

6.5.3 Results

After generating the speech quality scores for the G.723.1 error concealed and ILFR segments, the scores from all participants were averaged to determine a mean speech quality (MSQ) score at every rate of packet loss. The MSQ scores were then normalized out of 3.98 to determine their corresponding mean opinion scores (MOS) on a scale of 1 to 5. This normalization stems from the fact that the MOS for the G.723.1 dual rate coder is 3.98 out of 5, in the presence of zero packet loss. The mean opinion scores generated for G.723.1 error concealment and ILFR are illustrated in Figure 6-9.

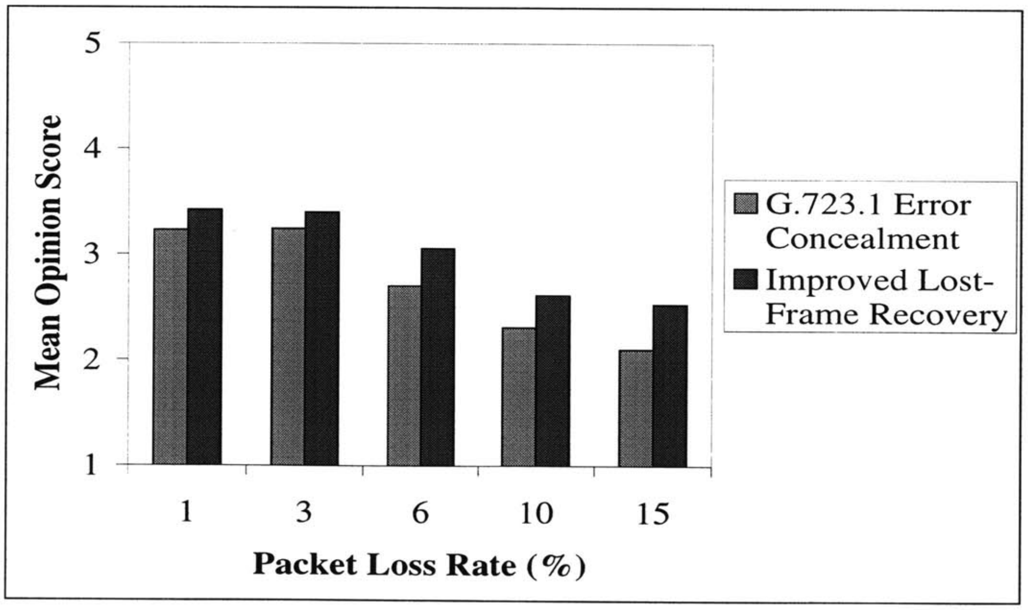

Figure 6-9: MOS of G.723.1 and Improved Lost-Frame Recovery.

As shown in Table 6.2, ILFR clearly increased the synthesized speech quality over G.723.1 error concealment by at least 5% and up to 20% for all rates of packet loss. When the participants were each asked to comment on the performance of ILFR and G.723.1 error concealment, their unified conclusions were that G.723.1 error concealment generated unnatural-sounding speech, metallic-sounding artifacts, high-energy spikes, and choppy synthesized speech while ILFR effectively eliminated these problems.

Technique	Congestion Level				
	Low	Low-Middle	Middle	Middle-High	High
G.723.1	3.22	3.24	2.70	2.30	2.10
ILFR	3.41	3.40	3.05	2.61	2.53
MOS Gain	1.06	1.05	1.13	1.14	1.20

Table 6.2: MOS Gain of Improved Lost-Frame Recovery over G.723.1.

Each participant was finally asked to provide an overall preference between the speech quality of two techniques. The results in Table 6.3 show that 66% of the participants thought that the speech quality due to ILFR was better than G.723.1 error concealment, while 33% thought that it was only slightly better. None of the participants felt that quality of ILFR was equal to or less than that provided by G.723.1 error concealment.

Overall Preference	Number of Participants
ILFR is Better than G.723.1	6
ILFR is Slightly Better than G.723.1	3
ILFR is Equal to G.723.1	0
ILFR is Slightly Worse than G.723.1	0
ILFR is Worse than G.723.1	0

Table 6.3: Overall Preference of Improved Lost-Frame Recovery and G.723.1.

Chapter 7

Packet Size Optimization

Chapter 6 presented a lost-frame recovery technique that improved upon the speech quality of G.723.1 error concealment. One of the technique's important features was the introduction of a Future Buffer that allowed effective linear interpolation and energy tapering to be performed at the decoder. Although this technique increased the speech quality over G.723.1 error concealment by up to 20% and was preferred by 66% of the tested population, the simulations only assumed packet loss and no out-of-order packet arrivals at the receiver. Since Internet congestion generates both missing and disordered packets, two of the thesis' main contributions are presented in this chapter.

First, an original frame-resequencing technique is presented that correctly sequences disordered voice packets before lost-frame recovery or decoding is initiated. Combined with ILFR, the two techniques are then jointly optimized using divide-and-conquer and the results from Shannon's Channel Coding Theorem to determine the optimal packet and Future Buffer sizes for transmitting speech packets across three multi-hop terrestrial networks.

7.1 A Frame-Resequencing Technique

Since network congestion results in both missing and disordered packets at the receiver, the decoder must implement a technique to resequence any out-of-order packets before lost-frame recovery or decoding is applied. To correctly sort incoming packets, a frame-resequencing technique was developed and its combined implementation with ILFR is illustrated in Figure 7-1.

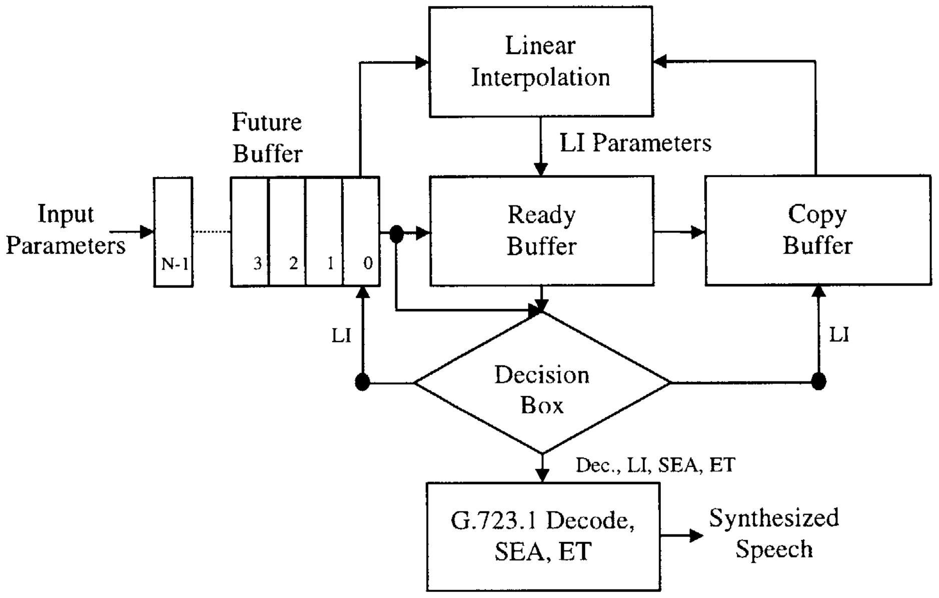

Figure 7-1: Frame-Resequencing and Improved Lost-Frame Recovery Technique.

As shown in Figure 7-1, one of the features of frame-resequencing is that the length of the Future Buffer is increased from one slot to N slots such that up to N arriving speech frames are stored and their RTP sequence numbers examined for resequencing. Specifically, the following four types of speech frames can arrive in slot $N-1$ of the Future Buffer:

1. **zero-delayed frame**, refers to an incoming good frame whose sequence number is one greater than the index number of slot $N-2$ of the Future Buffer. Zero-delayed frames do not need to be resequenced before decoding.

2. **delayed frame**, refers to an incoming good frame whose sequence number is greater than or equal to the index number of slot 0 of the Future Buffer minus 1. Delayed frames need to be correctly resequenced before decoding.

3. **severely-delayed frame**, refers to an incoming good frame whose sequence number is less than the index number of slot 0 of the Future Buffer minus 1. Since severely-delayed frames are essentially missing at the receiver, they cannot be resequenced. Thus, ILFR must be applied to recover these frames.

4. **infinitely-delayed frame**, refers to a frame which has simply been discarded some-

where within the network. Like severely-delayed frames, infinitely-delayed frames are missing at the receiver and cannot be resequenced; ILFR must be applied.

The frame-resequencing technique consists of two major components: (1) the slider and (2) the shuffler. The slider accepts incoming frames into the Future Buffer and slides them one-by-one along the Future Buffer until they reach the Ready Buffer. The shuffler continuously examines the sequence numbers of the frames and sorts them accordingly to generate the correct playout sequence. When combined together, the slider and shuffler enable incoming frames to be properly sequenced before decoding or lost-frame recovery is initiated. The frame-resequencing technique works as follows:

1. Increment all indices of the N-size Future Buffer by one.
2. Slide the contents in rightmost slot of Future Buffer, FB_0, to the Ready Buffer; slide the contents in Future Buffer slots [FB_1,...,FB_{N-1}] to [FB_0,...,FB_{N-2}] respectively; increment the index of leftmost slot in Future Buffer, FB_{N-1}, by one.
3. Store the future frame from the encoded sequence in FB_{N-1}.
4. Determine whether frame-resequencing is necessary:
   (a) If the sequence number of the future frame in FB_{N-1} matches the index of FB_{N-1}, no resequencing is necessary; jump to step (6).
   (b) If the sequence number of the future frame in FB_{N-1} does not match the index of FB_{N-1}, perform one of the following:
      i. If the sequence number of the future frame in FB_{N-1} is greater than or equal to the index of $FB_0 - 1$, the future frame is out of order and resequencing is necessary; set resequencing flag to 1.
      ii. If the sequence number of the future frame in FB_{N-1} is less than the index of FB_0, the future frame is missing; linear interpolation or energy tapering is necessary; reset resequencing flag to 0.
5. Check value of resequencing flag:
   (a) If the resequencing flag is set to 1, perform one of the following:

        i. If the sequence number of future frame in FB_{N-1} is greater than or equal to index of FB_0, reshuffle Future Buffer by placing future frame in FB_{N-1} in correct location in Future Buffer; jump to step (6).
        ii. If the sequence number of future frame in FB_{N-1} is equal to index of FB_0 minus 1, place future frame in FB_{N-1} in Ready Buffer; jump to step (6).
    (b) If the resequencing flag is set to 0, perform ILFR as described in Chapter 6; jump back to step (1).

6. Synthesize and decode the current frame in Ready Buffer.

7. Jump back to step (1).

As previously stated, the Internet's packet-switching framework results in both packet loss and network delay, requiring the receiver to implement a combined technique to resequence disordered frames and perform lost-frame recovery. Since increasing the number of network hops and congestion level increases the frequency of packet loss and network delay, two important questions arise as to how these techniques should be implemented at the decoder to best maximizes the speech quality and minimize the end-to-end delay. These questions are carefully considered in the following section by examining two variables inherent in both techniques: (1) the packet size and (2) the Future Buffer size.

7.2 Varying the Packet Size

The first question to be asked in best implementing the two techniques is the following: Given a network with a particular level of congestion, how will the transmission of different-sized voice packets affect speech quality and end-to-end delay? The advantages and disadvantages of sending different-sized packets are presented in the following subsections.

7.2.1 Smaller Packets

The two advantages of sending smaller packets through a network include more effective recovery from lost frames and lower end-to-end delay. Since linear interpolation recovers the missing LSP and gain parameters based on the previous and future speech frames, the amount of degradation introduced in the synthesized speech is minimized when smaller packets are interpolated. Similarly, since energy tapering involves repeating and scaling information from the previous good frame over consecutive frame erasures, the total amount of degradation is again minimized if smaller packets and hence less energy tapering is applied.

The second advantage of sending smaller packets is a decrease in end-to-end delay, τ_{ee}, as shown in equation (2.1). Since the required delays for buffering speech frames at the encoder and decoder, τ_{enc} and τ_{dec} respectively, are a function of the number of speech frames per packet, smaller packet sizes clearly result in a lower end-to-end delay.

Effective Bandwidth

Along with the advantages of smaller packet sizes is one major disadvantage — effective bandwidth reduction. The effective bandwidth, β_{eff}, is the transmission rate required for sending a packet's useful information or payload through a network and is expressed as follows,

$$\beta_{eff} = \left(\frac{PL}{PL + H}\right) \times \beta_{orig} \tag{7.1}$$

which reduces to,

$$\beta_{eff} = \left(1 - \frac{H}{H + PL}\right) \times \beta_{orig} \tag{7.2}$$

where β_{orig} is the original network bandwidth and PL and H are the size of the payload and header in bits. Since the size of a packet is equal to $(PL + H)$ bits, it is clear from equation (7.2) that as the size of the payload is decreased, a proportionally greater amount of packet overhead is introduced into the network thereby decreasing β_{eff}.

Another way to express β_{eff} is in terms of the total number of hops, H_{tot}, within a network,

$$\beta_{eff} = \frac{PL}{H_{tot} \times \tau_{hop}} \tag{7.3}$$

where τ_{hop} is the processing delay for a single hop. For instance, τ_{hop} for a digital-to-digital transit exchange is approximately 0.45 ms according to the ITU-T Recommendation G.114 [8]. From equation (7.3) it is also clear that as the size of the payload is decreased or as the number of network hops is increased, β_{eff} is reduced.

From the preceding discussion, the effective bandwidth of a network decreases when smaller packets are transmitted across or when the number of network hops increases. As presented in subsection 7.2.3 on Shannon's Channel Coding Theorem, this reduction in β_{eff} leads to a degradation in speech quality.

7.2.2 Larger Packets

Similar to the discussion on smaller speech packets are both advantages and disadvantages of larger-sized packets. From equation (7.2), the primary advantage of sending larger speech packets is an increase in β_{eff} since a smaller amount of packet overhead is introduced into the network. In addition, by decreasing H_{tot} in equation (7.3), β_{eff} increases.

However, the clear disadvantages of transmitting larger packets are increased end-to-end delay and less effective lost-frame recovery. Specifically, end-to-end delay increases since more frames must be buffered at the encoder and decoder which increases the delays of the packetization and de-packetization processes.

Less effective lost-frame recovery is also achieved because linear interpolation and energy tapering must be applied over larger segments of lost information, thereby introducing more distortion into the synthesized speech. Linear interpolation of the LSP and gain parameters is performed for an m-sized packet (i.e., a packet containing m 30-ms speech frames) as described below.

LSP Interpolation for an m-Sized Packet

Assuming the n^{th} packet in an encoded speech sequence is missing, the m LSP vectors are interpolated using $\mathbf{\Lambda}_{n-1}$, the LSP vector from the last frame of the $(n-1)^{th}$ packet and $\mathbf{\Lambda}_{n+1}$, the LSP vector from the first frame of the $(n+1)^{th}$ packet. The missing LSP vector, $\mathbf{\Lambda}_j^n$, $0 \leq j \leq (m-1)$, is recovered by applying the following weighting formula,

$$\mathbf{\Lambda}_j^n = \left(\frac{m-j}{m+1}\right)\mathbf{\Lambda}_{n-1} + \left(\frac{1+j}{m+1}\right)\mathbf{\Lambda}_{n+1} \tag{7.4}$$

Gain Interpolation for an m-Sized Packet

Since the fixed codebook gains are calculated on a subframe by subframe basis, the missing gains, Γ_{kj}^n, for every subframe k, $0 \leq k \leq 3$, of every frame j, $0 \leq j \leq (m-1)$, are interpolated using $\overline{\Gamma}_{n-1}$, the averaged fixed codebook gain from the last frame of the $(n-1)^{th}$

packet and $\overline{\Gamma}_{n+1}$,, the averaged fixed codebook gain from the first frame of the $(n+1)^{th}$ packet. The following formula describes the weighting function to determine λ^n_{kj},

$$\Gamma^n_{kj} = \left(\frac{4m - 4j + k}{4m+1}\right)\overline{\Gamma}_{n-1} + \left(\frac{4j - k + 1}{4m+1}\right)\overline{\Gamma}_{n+1} \tag{7.5}$$

7.2.3 Shannon's Channel Coding Theorem

As described in subsections 7.2.1 and 7.2.2, smaller packets or an increase in the number of network hops results in a greater reduction of β_{eff} necessary for transmitting information across a network. Although injecting packets of any size always decreases the network's original bandwidth, and even more so when PL is decreased and H_{tot} is increased, as long as the effective bandwidth does not fall below the rate of information transmission, then the distortion between the output and input can be minimized. This is the heart of Shannon's Channel Coding Theorem [4] and is stated as follows:

All rates below capacity C are achievable. Specifically, for every rate $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with maximum probability of error, $P_e^{(n)} \to 0$.
Conversely, any sequence of $(2^{nR}, n)$ codes with $P_e^{(n)} \to 0$ must have $R \leq C$.

Thus, given the rate of information transmission, R, between sender and receiver, the effective bandwidth or channel capacity, C, must be greater than or equal to R in order to minimize the probability of receiving incorrect information. The probability of receiving incorrect information or probability of error, $P_e^{(n)}$, is expressed as follows,

$$P_e^{(n)} \geq 1 - \frac{C}{R} - \frac{1}{nR} \tag{7.6}$$

where n is the number of bits per codeword. From equation (7.6), it is clear that for $R > C$, the probability of error becomes exponentially unbounded from 0 for sufficiently large n, making it impossible to achieve an arbitrarily low probability of error at rates above capacity. This inequality is graphically illustrated in Figure 7-2. As a simple example, if one were to send G.723.1 compressed speech at 6.3 kb/s over a network with $\beta_{eff} = 4.8$ kb/s, the error introduced at the decoder would grow unbounded, thereby generating continuously degraded and unintelligible synthesized speech. Thus, in order to be able to control the distortion at the receiver, it is required that the rate of transmission is less than β_{eff}.

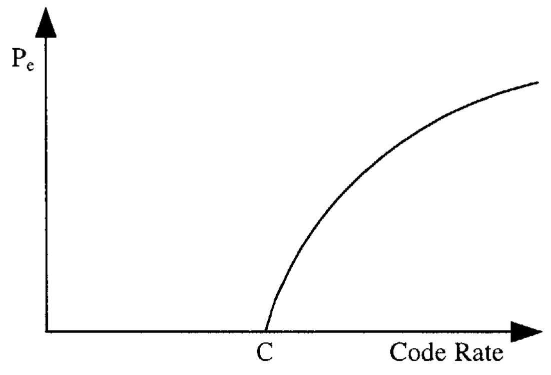

Figure 7-2: Lower Bound on the Probability of Error.

7.3 Varying the Future Buffer Size

The second question to be asked in determining the best implementation of the frame-resequencing and lost-frame recovery techniques is the following: Given a network with a particular level of congestion, how will the arrival of packets in different-sized Future Buffers affect speech quality and end-to-end delay? The advantages and disadvantages of using different-sized Future Buffers are presented in the following subsections.

7.3.1 Smaller Future Buffers

The primary advantage of using a smaller-sized Future Buffer is a decrease in end-to-end delay. With the introduction of a Future Buffer, the total end-to-end delay, τ_{ee}, is now expressed as follows,

$$\tau_{ee} = \tau_{enc} + \tau_{dec} + \tau_{buf} + \tau_{trans} \tag{7.7}$$

The extra delay introduced by the Future Buffer, τ_{buf}, is equivalent to the length of the buffer in packets, which is a function of the number of speech frames. For instance, if the length of the Future Buffer is n packets long and every packet consists of m 30-ms speech frames, then τ_{buf} is $(30 \times m \times n)$ ms. Clearly, if the size of the Future buffer is decreased, τ_{ee} is also decreased, thereby improving the conversational dynamics and speech quality of the system.

The disadvantage of using smaller-sized Future Buffers is clear when considering the original reason for extending its length beyond one slot. As described in section 6.3, the length of the Future Buffer required for lost-frame recovery is one frame length or 30 ms, because only the future frame needs to be stored and its sequence number examined in order to perform linear interpolation and energy tapering. However, since frames can arrive also disordered at the receiver, the length of the Future Buffer must be extended in order to successfully resequence any out-of-order frames beyond a single frame delay. By decreasing the length of the Future Buffer, fewer disordered packets can be correctly resequenced, thus degrading the synthesized speech quality as the need for lost-frame recovery is increased.

7.3.2 Larger Future Buffers

The advantage of using a larger-sized Future Buffer is that a higher probability of correctly resequencing disordered packets is possible, thereby decreasing the necessity for lost-frame recovery and increasing the synthesized speech quality. However, the disadvantage of using a larger-sized Future Buffer, as shown in equation (7.7), is an increase in τ_{ee} and thus a decrease in conversational dynamics and system speech quality.

7.4 Determining the Optimal Packet and Future Buffer Sizes

Having examined the advantages and disadvantages of selecting different-sized packets and Future Buffers, the following question can now be presented: Given a particular network with a certain congestion level, what are the optimal packet size, P_{opt}, and Future Buffer size, B_{opt}, for maximizing speech quality and minimizing end-to-end delay? This question requires the joint optimization of two variables — the packet size and Future Buffer size — where the length of the Future Buffer depends on packet size. Thus, to simplify the complexity and recursive nature of this problem, the following divide-and-conquer strategy was applied to determine P_{opt} and B_{opt}:

1. Determine the mean opinion scores of the synthesized speech as a function of packet loss alone and zero end-to-end delay. Denote this as the zero-delay MOS (ZDL MOS) set.

2. Determine the mean opinion scores of the synthesized speech as a function of end-to-end delay alone and zero packet loss. Denote this as the zero-packet-loss MOS (ZPL MOS) set.

3. Average the ZDL and ZPL MOS sets for all packet and Future Buffer sizes to generate a set of combined mean opinion scores. Denote this as the combined MOS (CMB MOS) set.

4. Select the next network for which to find P_{opt} and B_{opt}:

   (a) Apply the results from Shannon's Channel Coding Theorem to eliminate all possible packet sizes and Future Buffer sizes which are invalid for transmission.

   (b) Select the next congestion level for which to find P_{opt} and B_{opt}:

      i. Using the CMB MOS set, select the highest combined mean opinion score from the allowable packet and Future Buffer sizes as determined in step (4a). P_{opt} and B_{opt} are represented by this mean opinion score.

      ii. Return to step (4b) until P_{opt} and B_{opt} have been found for all congestion levels.

   (c) Return to step (4) until P_{opt} and B_{opt} have been found for all networks.

The importance of the divide-and-conquer strategy is breaking the complexity of the original problem into two simpler problems by first determining the ZDL MOS set of the synthesized speech in step (1) and then determining the ZPL MOS set of the synthesized speech in step (2). Thus, the impact of packet loss on speech quality and impact of end-to-end delay on speech quality are separately determined. The results from each subproblem are then averaged in step (3) to generate the CMB MOS set upon which the results from Shannon's Channel Coding Theorem are applied for determining P_{opt} and B_{opt}. The following sections present the simulation methods and results obtained from determining the ZDL MOS, ZPL MOS, and CMB MOS sets.

7.5 Zero-Delay MOS (ZDL MOS) Simulations

Since the quality of service for Internet Telephony depends on both maximizing speech quality and minimizing end-to-end delay, the purpose of the ZDL MOS simulations is to subjectively evaluate the speech quality as a function of packet loss alone. To determine the ZDL MOS set, an 800-frame speech segment was injected with five rates of of randomly distributed packet loss and recovered using combined frame-resequencing and ILFR. For every rate simulated, four packet sizes and up to ten Future Buffer sizes were tested. Informal listening tests were then conducted by nine participants who subjectively scored the speech qualities of the recovered synthesized segments. The following subsections describe the simulation environments, methods, and results.

7.5.1 Environments

To determine the ZDL MOS set, three simulations were performed wherein each simulation environment consisted of a one multi-hop network and five levels of congestion. A description of the multi-hop networks, the five congestion levels, and an important constraint placed on τ_{ee} that limited the possible set of packet and Future Buffer sizes within a given network is presented below.

Multi-Hop Packet-Switching Networks

A packet-switching network (PSN) consists of a sender and receiver interconnected with at least two digital local packet-switching exchanges (PSE) and any number of intermediate hops. The purpose of the source PSE is to store an incoming packet from the sender, inspect its destination address, and forward the packet to an intermediate PSE based on the information contained in the PSE's routing directory. As each packet is received, stored, and inspected at each hop, it is then forwarded to the next available hop within the network. At the destination PSE, determined by the destination address within the packet, the packet is finally passed to the receiver. Figure 7-3 illustrates a typical packet-switching network.

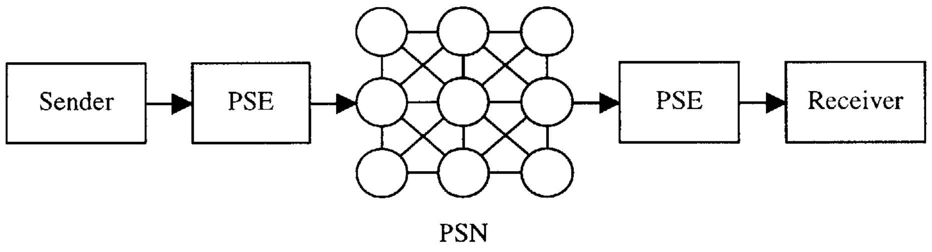

Figure 7-3: Packet-Switching Network.

Three 5000 km, point-to-point, all digital networks were considered in the ZDL MOS simulations. From equation (2.2), the total transmission delay, τ_{trans}, between sender and receiver is approximately 20 ms. Similarly, the total processing delay, τ_{proc}, for each network is calculated as follows,

$$\tau_{proc} = 2\tau_{pse} + H_{tot} \times \tau_{hop} \tag{7.8}$$

where τ_{pse} and τ_{hop} are the PSE and hop delays respectively and H_{tot} is the total number of hops within the network. According to the ITU-T Recommendation G.114, the delay for a digital PSE is approximately 0.825 ms while the delay for a digital hop is approximately 0.45 ms [8].

Table 7.1 summarizes the three networks and their corresponding processing delays considered in the ZDL MOS simulations. The subscript in the "Network" column indicates the total number of hops within that network. Thus, NW_{50} represents a 50-hop PSN, while NW_{100} and NW_{150} represent 100-hop and 150-hop PSNs respectively.

Network	τ_{proc} (ms)
NW_{50}	24.15
NW_{100}	46.65
NW_{150}	69.15

Table 7.1: Multi-Hop Networks and Their Processing Delays.

Congestion Levels

Five congestion levels were then associated with each PSN. Similar to the congestion levels presented in Chapter 6, the following congestion levels were assumed: (1) Low, (2) Low-Middle, (3) Middle, (4) Middle-High, and (5) High. Unlike the simulations presented in Chapter 6 which only considered fixed packet loss, the ZDL MOS simulations considered both fixed packet loss as well as packet delay. Thus, each congestion level corresponded to a rate of potential packet loss (PPL) which was the sum of the following two quantities: (1) fixed packet loss (FPL) and (2) Gaussian packet delay (GPD). For low congestion, $PPL = 1\%$. For low-middle congestion, $PPL = 3\%$. For middle, middle-high, and high congestion levels, the potential packet loss rates were 6%, 10%, and 15% respectively.

As described section 6.5, FPL refers to the percentage of packets which are randomly discarded at hops within the network, and thus arrive as infinitely-delayed or missing frames at the receiver. GPD, on the other hand, refers to the percentage of packets which are randomly selected for delay and may thus arrive as either zero-delayed, delayed, or severely-delayed frames at the receiver. Specifically, the amount of packet delay was determined by the following positive-valued, zero-mean, Gaussian PDF with variance σ^2,

$$F_x(x_0) = \frac{2}{\sqrt{2\pi}\,\sigma} e^{-\frac{x_0^2}{2\sigma^2}} \tag{7.9}$$

where x, $0 \leq x \leq \infty$, is a random variable representing the delay for a GPD packet in packet lengths. Unlike FPL packets, GPD packets have the opportunity of being correctly resequenced within the Future Buffer before decoding or lost-frame recovery is initiated. Thus, the upper bound on the total rate of lost packets is PPL which corresponds to the state in which none of the GDP packets are correctly resequenced; the lower bound of the total rate of lost packets is FPL, which corresponds to the state in which all of the GDP packets are correctly resequenced.

For increasing levels of congestion, the rates of FPL, GPD, and the standard deviation of the Gaussian PDF, σ, were each increased. The reason for increasing the rates of FPL and GPD was based on the assumption that as congestion levels rise, the number of packets which are discarded and experience network delay increases as well. Similarly, the standard deviation of $F_x(x_0)$ was increased under the assumption that with rising levels of congestion, the amount of delay or interarrival time between delayed packets also increases. Table 7.2 summarizes the rates of FPL, GPD, PPL, and σ used to simulate potential packet loss at the five congestion levels.

Congestion Level	FPL (%)	GPD (%)	PPL (%)	σ
Low	0	1	1	1
Low-Middle	1	2	3	2
Middle	2	4	6	3
Middle-High	3	7	10	4
High	4	11	15	5

Table 7.2: Distribution of Fixed and Potential Packet Loss.

Constraint on τ_{ee}

Finally, there existed one important constraint that placed a limitation on the size of the packet and Future Buffer. This constraint was the maximum recommended value of τ_{ee} according to the ITU-T Recommendation G.114. In the Recommendation G.114, the ITU-

T recommends a maximum one-way transmission delay of 400 ms for connections with echo adequately controlled, provided that Administrations are aware of the transmission time impact on the transmission quality of user applications [8].

Given this upper bound on end-to-end delay, the set of possible packet sizes, P_{set}, and their corresponding Future Buffer sizes, B_{set}, were determined from equations (2.2) and (7.7) for the three 5000 km packet-switching networks. As summarized in Table 7.3, four packets sizes in P_{set} and ten Future Buffer sizes in B_{set} were considered in the ZDL MOS simulations.

Packet Size (frames)	Future Buffer Size (frames)	τ_{ee} (ms)
1	1	110
1	2	130
1	3	170
1	4	200
1	5	230
1	6	260
1	7	290
1	8	320
1	9	350
1	10	380
2	2	200
2	4	260
2	6	320
2	8	380
3	3	320
3	6	380
4	4	380

Table 7.3: Packet Sizes and Their Future Buffer Sizes.

7.5.2 Methods

First, the 800-frame speech segment was transmitted across each congested network. Like the simulations in Chapter 6, the 800-frame segment consisted of 6 male-spoken English sentences and was encoded using G.723.1 voice compression. To simulate the different potential packet loss rates, a random number generator was applied to the 800-frame segment to randomly select ($\frac{FPL}{100} \times 800$) packets to discard. Next, the random number generator was applied to the corrupted segment to randomly select ($\frac{GPD}{100} \times 800$) packets to delay. The amount of delay was determined by applying $F_x(x_0)$, expressed as equation (7.9), to each of the GDP packets at that congestion level.

After simulating potential packet loss, frame-resequencing and ILFR were jointly applied on the corrupted signal to generate the synthesized speech segment. This process was performed at all rates of potential packet loss for the four packet sizes in P_{set} and the ten Future Buffer sizes in B_{set}, resulting in 85 recovered synthesized segments, 17 at each congestion level. Informal listening tests were then conducted by the nine participants who were asked to subjectively score the speech quality of each recovered segment.

Each listener was first played an uncorrupted G.723.1 synthesized segment, with a de facto speech quality score of $\frac{50}{50}$. Then, the recovered segment was played and the listener was asked to subjectively score its speech quality relative to the uncorrupted segment, with 1 being the lowest possible score and 50 being the highest possible score. This process was performed for all 85 recovered segments, 17 for each potential packet loss rate.

7.5.3 Results

After generating the speech quality scores for the frame-resequenced, ILFR segments, the scores from all participants were averaged to determine a mean speech quality (MSQ) score at every potential packet loss rate for all packet sizes in P_{set} and Future Buffer sizes in B_{set}. Since the MOS for the G.723.1 dual rate speech coder is 3.98 out of 5 in the presence of zero packet loss, the MSQ scores were then normalized out of 3.98 to determine their corresponding mean opinion scores on a scale of 1 to 5. Figures 7-4 through 7-8 illustrate the ZDL MOS set for the low, low-middle, middle, middle-high, and high congestion levels using the four packet sizes from P_{set} and the ten Future Buffer sizes from B_{set}.

7.6 Zero-Packet-Loss MOS (ZPL MOS) Simulations

According to the divide-and-conquer strategy, the purpose of determining the ZPL MOS set is to subjectively evaluate the quality of synthesized speech as a function of end-to-end delay alone. This is important since maximizing the quality of service for Internet Telephony requires minimizing the delay in addition to maximizing the quality of synthesized speech. Although the necessary facilities were not available to simulate end-to-end delay experiments to this effect, the test documented in the ITU-T Recommendation G.114, described in Chapter 2, provided the necessary results.

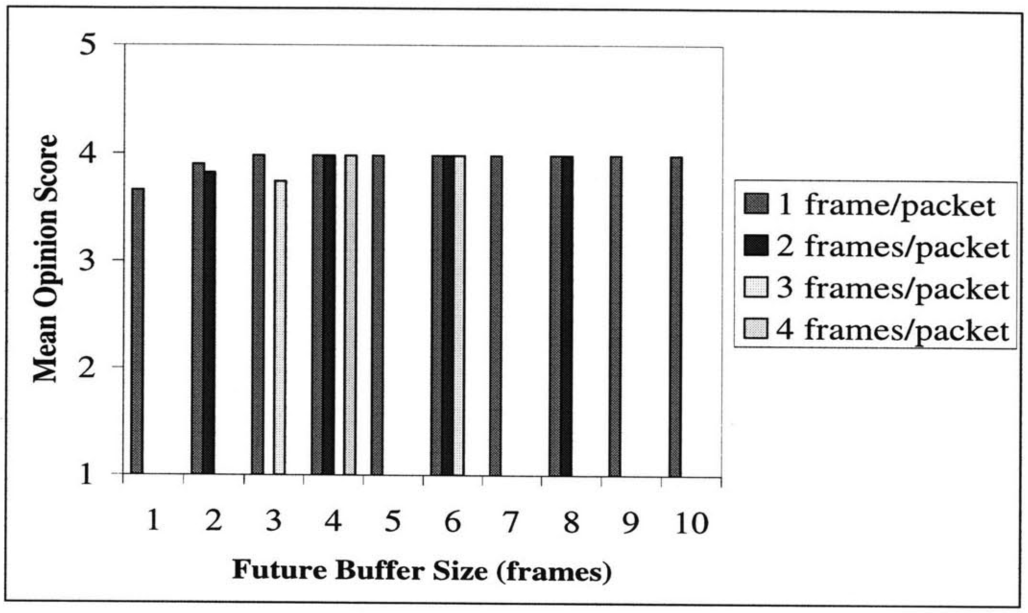

Figure 7-4: Zero-Delay MOS for Low Congestion.

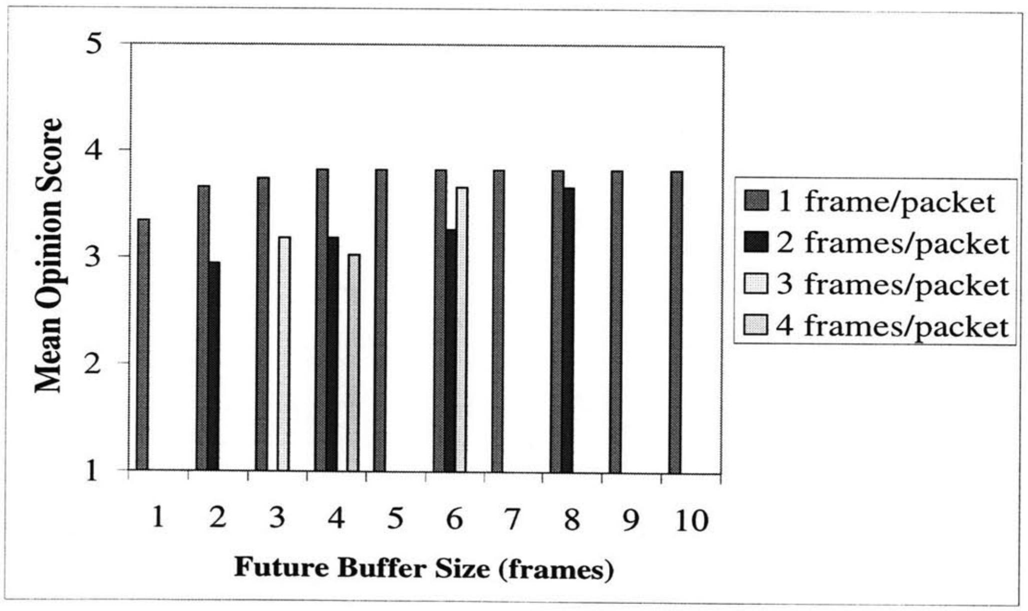

Figure 7-5: Zero-Delay MOS for Low-Middle Congestion.

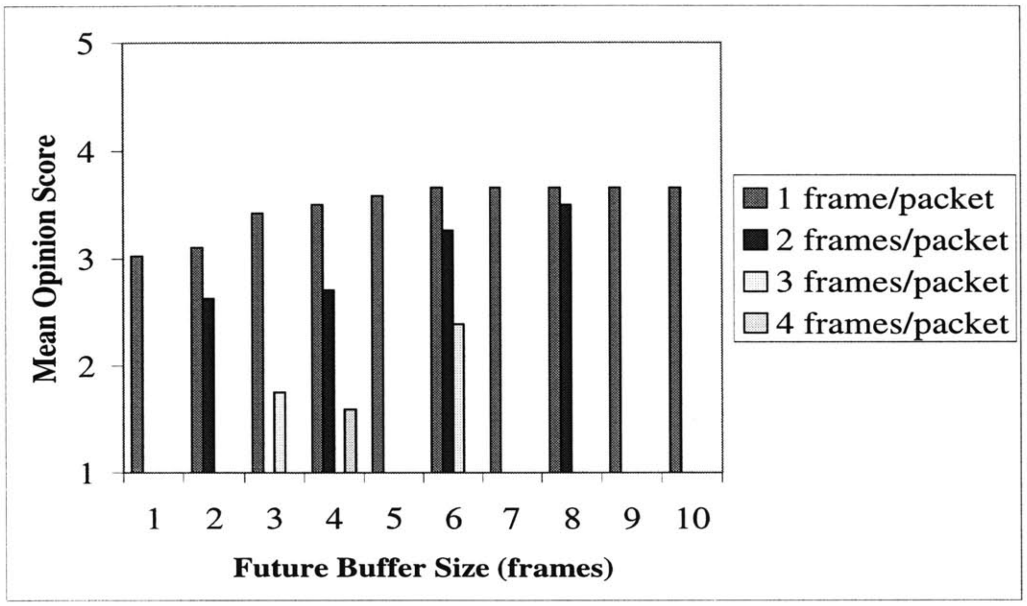

Figure 7-6: Zero-Delay MOS for Middle Congestion.

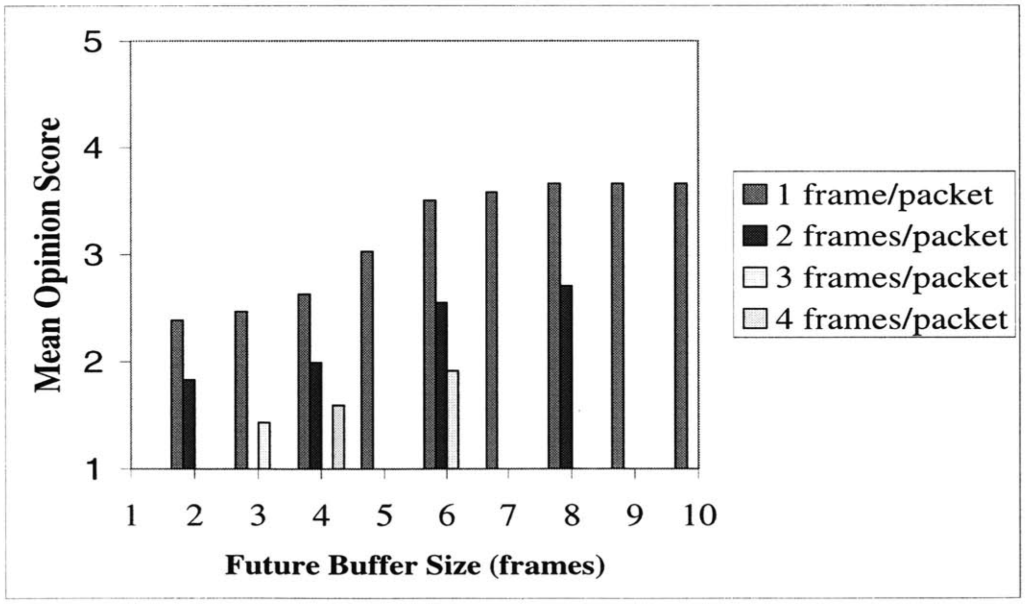

Figure 7-7: Zero-Delay MOS for Middle-High Congestion.

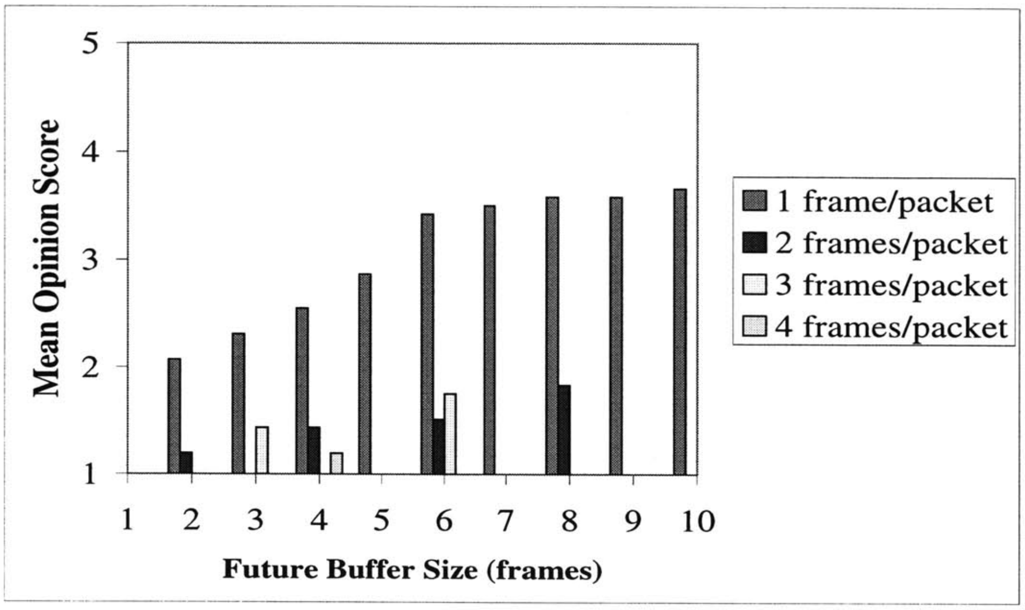

Figure 7-8: Zero-Delay MOS for High Congestion.

Figure 2-2 illustrated decaying mean opinion scores for echo-free telephone connections in which increasing values of one-way delay were inserted into telephone conversations. Given the toll quality of speech coding (i.e., 64 kb/s PCM) used in these tests, the MOS for the G.723.1 dual rate speech coder was approximated by normalizing the MOS curve in Figure 2-2 to the near toll quality of the G.723.1 coder. This is a valid approximation since both speech coders are of high quality; thus, they can be assumed to exhibit similar degradations in MOS as delay is increased. Figure 7-9 illustrates an approximation for the MOS of the G.723.1 dual rate speech as a function of end-to-end delay.

Since the packet and Future Buffer sizes are each functions of end-to-end delay, equation (7.7) was used to calculated τ_{ee} for the 5000 km digital networks implemented with each of the four packet sizes in P_{set} and ten Future Buffer sizes from B_{set}. For all values in P_{set} and B_{set}, the ZPL MOS set was then extrapolated from Figure 7-9 to yield Figure 7-10.

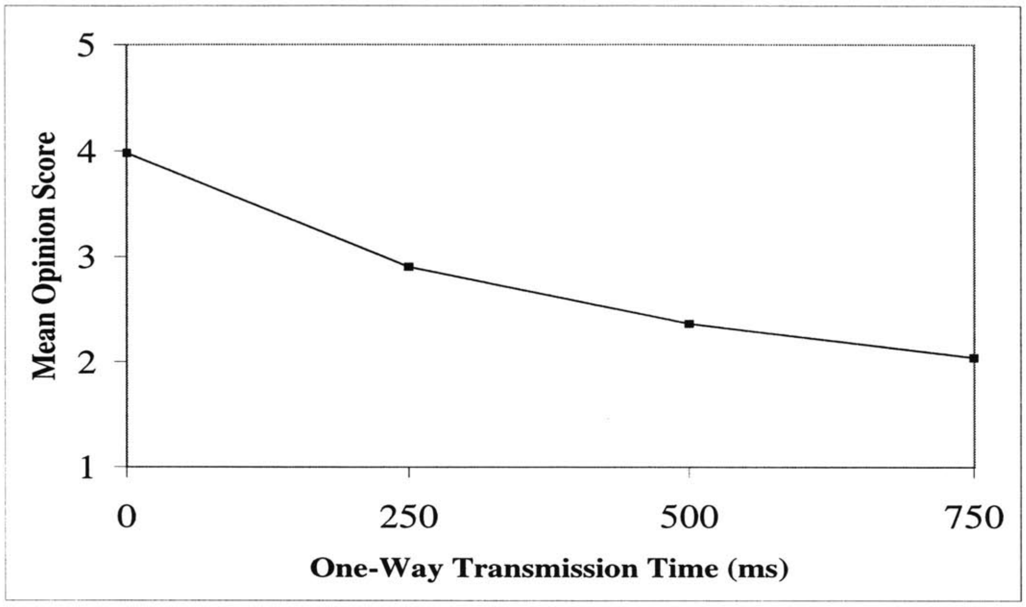

Figure 7-9: G.723.1 MOS for Four Delay Conditions.

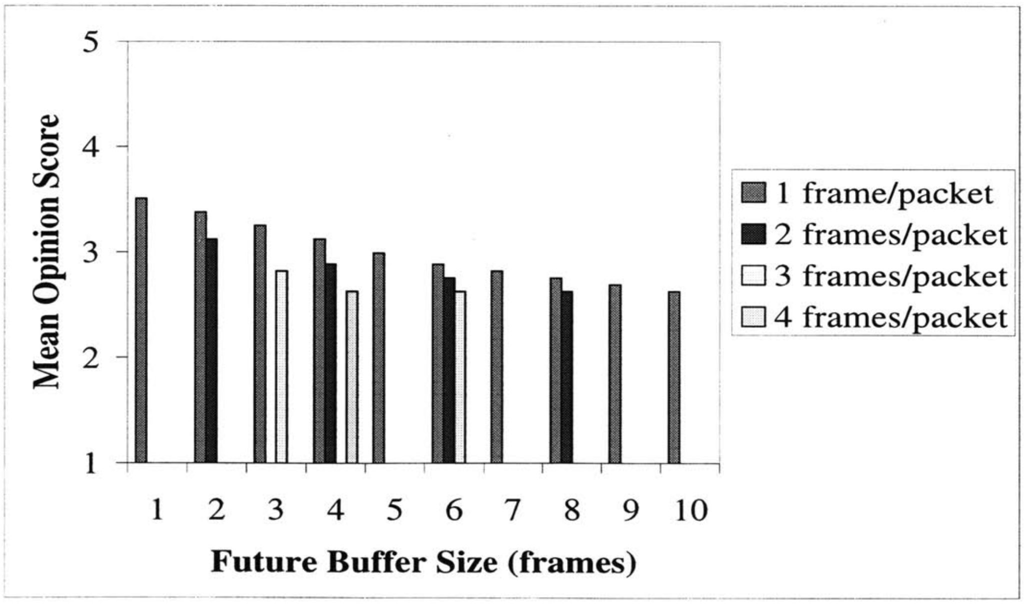

Figure 7-10: G.723.1 Zero-Packet-Loss MOS

7.7 Combined MOS (CMB MOS) Simulations

After determining the ZDL MOS and ZPL MOS sets as described in the divide-and-conquer strategy, the next step was to average the results from both MOS sets to generate the CMB MOS set. The results from the CMB MOS set thus represented the mean opinion scores for the synthesized speech in the presence of both packet loss and end-to-end delay. Tables 7.4 through 7.7 summarize the CMB MOS sets for each of the five congestion levels.

After determining the CMB MOS set, it was then possible to determine the optimal packet and Future Buffer sizes for transmitting the 800-frame speech segment over any of the three networks at any potential packet loss rate. As summarized in the divide-and-conquer strategy, the next step in determining P_{opt} and B_{opt} was to apply the results from Shannon's Channel Coding Theorem to eliminate any invalid packet and Future Buffer sizes from P_{set} and B_{set}. The following subsections describe the application of Shannon's Channel Coding Theorem and present the optimal packet and Future Buffer sizes for transmitting G.723.1 compressed voice across all multi-hop networks for all levels of congestion.

7.7.1 50-Hop Network

The total processing delay incurred by the 50-hop digital network is approximately 24.15 ms as shown in Table 7.1. Using equation (7.3), the effective bandwidth for each of the four packet sizes in P_{set} was calculated for the 50-hop network.

As shown in Table 7.9, β_{eff} is greater than the required speech coder rate of 6.3 kb/s for any packet size in P_{set}. Thus, an arbitrarily low probability of error is achievable for any packet size according to Shannon's Channel Coding Theorem; all packet and Future Buffer sizes were thus considered valid for transmission across the 50-hop network. The optimal packet and Future Buffer sizes were determined by finding the maximum combined mean opinion scores in Tables 7.4 through 7.8. Table 7.10 summarizes P_{opt} and B_{opt} for each congestion level within the 50-hop network.

7.7.2 100-Hop Network

For the 100-hop network, the total processing delay incurred is approximately 46.65 ms as shown in Table 7.1. Again, β_{eff} was calculated for each of the four packet sizes in P_{set}.

As shown in Table 7.11, only packet sizes greater than or equal to 2 frames per packet

Packet Size (frames)	Future Buffer Size (frames)	Combined MOS
1	1	3.66
1	2	3.90
1	3	3.98
1	4	3.98
1	5	3.98
1	6	3.98
1	7	3.98
1	8	3.98
1	9	3.98
1	10	3.98
2	2	3.82
2	4	3.98
2	6	3.98
2	8	3.98
3	3	3.82
3	6	3.98
4	4	3.98

Table 7.4: Combined MOS for Low Congestion.

Packet Size (frames)	Future Buffer Size (frames)	Combined MOS
1	1	3.42
1	2	3.52
1	3	3.49
1	4	3.47
1	5	3.41
1	6	3.35
1	7	3.32
1	8	3.29
1	9	3.25
1	10	3.22
2	2	3.03
2	4	3.03
2	6	3.00
2	8	3.02
3	3	3.00
3	6	3.02
4	4	2.82

Table 7.5: Combined MOS for Low-Middle Congestion.

Packet Size (frames)	Future Buffer Size (frames)	Combined MOS
1	1	3.27
1	2	3.24
1	3	3.34
1	4	3.31
1	5	3.29
1	6	3.27
1	7	3.24
1	8	3.21
1	9	3.18
1	10	3.14
2	2	2.87
2	4	2.79
2	6	3.01
2	8	3.06
3	3	2.28
3	6	2.51
4	4	2.11

Table 7.6: Combined MOS for Middle Congestion.

Packet Size (frames)	Future Buffer Size (frames)	Combined MOS
1	1	2.87
1	2	2.88
1	3	2.85
1	4	2.87
1	5	3.01
1	6	3.19
1	7	3.20
1	8	3.21
1	9	3.18
1	10	3.14
1	2	2.48
2	4	2.44
2	6	2.65
2	8	2.67
3	3	2.13
3	6	2.27
4	4	2.11

Table 7.7: Combined MOS for Middle-High Congestion.

Packet Size (frames)	Future Buffer Size (frames)	Combined MOS
1	1	2.63
1	2	2.72
1	3	2.78
1	4	2.83
1	5	2.93
1	6	3.15
1	7	3.16
1	8	3.17
1	9	3.18
1	10	3.14
2	2	2.16
2	4	2.16
2	6	2.13
2	8	2.11
3	3	2.24
3	6	2.29
4	4	1.90

Table 7.8: Combined MOS for High Congestion.

generate an effective bandwidth that is greater than the speech coder rate of 6.3 kb/s. For a packet size of 1 frame per packet, $\beta_{eff} = 4.05$ kb/s and an arbitrarily low probability of error cannot achieved according to Shannon's Channel Coding Theorem. Thus, a packet size of 1 frame per packet was considered invalid for transmission across the 100-hop network and was eliminated from P_{set} when determining the optimal packet and Future size. Of the three remaining packet sizes and seven remaining Future Buffer sizes, P_{opt} and B_{opt} were again determined by finding the maximum combined mean opinion scores in Tables 7.4 through 7.8. Table 7.12 summarizes the P_{opt} and B_{opt} for each congestion level within the 100-hop network.

7.7.3 150-Hop Network

For the 150-hop network, the total processing delay incurred is approximately 69.15 ms as shown in Table 7.1. Again, β_{eff} was calculated for the four packet sizes in P_{set}.

As shown in Table 7.13, only packet sizes greater than or equal to 3 frames per packet generate an effective bandwidth which is greater than the speech coder rate of 6.3 kb/s. For packet sizes of 1 or 2 frames per packet, $\beta_{eff} = 2.73$ kb/s and $\beta_{eff} = 5.47$ kb/s respectively; thus, an arbitrarily low probability of error cannot be achieved according to

Packet Size (frames/packet)	Payload Size (bits)	β_{eff} (kb/s)
1	189	7.83
2	378	15.65
3	567	23.48
4	756	31.30

Table 7.9: 50-Hop Network: Effective Bandwidths for Four Packet Sizes.

Congestion Level	P_{opt} (frames)	B_{opt} (frames)
Low	1	2
Low-Middle	1	2
Middle	1	3
Middle-High	1	8
High	1	9

Table 7.10: 50-Hop Network: Optimal Packet and Future Buffer Sizes.

Packet Size (frames/packet)	Payload Size (bits)	β_{eff} (kb/s)
1	189	4.05
2	378	8.10
3	567	12.15
4	756	16.21

Table 7.11: 100-Hop Network: Effective Bandwidths for Four Packet Sizes.

Congestion Level	P_{opt} (frames)	B_{opt} (frames)
Low	2	4
Low-Middle	2	4
Middle	2	8
Middle-High	2	8
High	3	6

Table 7.12: 100-Hop Network: Optimal Packet and Future Buffer Sizes.

Shannon's Channel Coding Theorem. Packet sizes of 1 and 2 frames per packet were both considered invalid for transmission across the 150-hop network and eliminated from P_{set} when determining the optimal packet and Future Buffer size. Of the two remaining packet sizes and three remaining Future Buffer sizes, P_{opt} and B_{opt} were determined by finding the maximum combined mean opinion scores from Tables 7.4 through 7.8. Table 7.14 summarizes the P_{opt} and B_{opt} for each congestion level within the 150-hop network.

Packet Size (frames/packet)	Payload Size (bits)	β_{eff} (kb/s)
1	189	2.73
2	378	5.47
3	567	8.20
4	756	10.93

Table 7.13: 150-Hop Network: Effective Bandwidths for Four Packet Sizes.

Congestion Level	P_{opt} (frames)	B_{opt} (frames)
Low	3 or 4	6 or 4
Low-Middle	3	6
Middle	3	6
Middle-High	3	6
High	3	6

Table 7.14: 150-Hop Network: Optimal Packet and Future Buffer Sizes.

7.7.4 Summary

In sum, the results from packet size optimization show two monotonically increasing relationships. The first relationship is the following: As the number of network hops increases, larger packet sizes are required for transmission in order to maximize speech quality and minimize end-to-end delay. This result is attributed to the fact that as number of network hops increases, thereby decreasing the effective bandwidth, only packet sizes whose β_{eff} exceed the coder rate are able to achieve an arbitrarily low probability of error according to Shannon's Channel Coding Theorem.

The second monotonically increasing relationship is the following: As the level of congestion increases within a network, larger Future Buffers are required to maximize speech quality and minimize end-to-end delay. This result is attributed to the fact that the ZDL MOS set degrades more sharply as packet loss is increased, as opposed to the ZPL MOS set which degrades less sharply as delay is increased. Thus, since the perception of speech quality is more sensitive to increased packet loss than it is to increased delay, larger Future Buffers can be used to correctly resequence out-or-order frames, and thus minimize the degradation due to ILFR, despite an increase in end-to-end delay.

Chapter 8

Conclusion and Recommendations

8.1 Summary

This thesis provided a detailed treatment on TCP/IP protocols and low bit rate digital speech coding techniques. Based on their principles, three optimization methods were then developed to maximize the speech quality and minimize the end-to-end delay for an Internet Telephony system. These design methods and their results are summarized as follows:

- Through protocol optimization, RTP was introduced and evaluated as a near-optimal protocol, by combining the reliability of TCP and speech of UDP, for transmitting voice packets over the Internet. RTP's multi-casting capability was shown to be an important advantage over TCP and UDP by allowing voice compression with different speech coders incident upon the available bandwidth and level of congestion within a network.

- Through packet recovery optimization, an original lost-frame recovery technique combining three algorithms — linear interpolation, selective energy attenuation, and energy tapering — was developed to treat five rates of randomly distributed packet loss. The advantages of ILFR included eliminating unnatural-sounding speech, metallic-sounding artifacts, high-energy spikes, and choppy synthesized speech otherwise generated by G.723.1 error concealment. Based on informal listening tests, ILFR was shown to achieve up up to a 20% increase in speech quality over G.723.1 error concealment and to be preferred by 66% of the tested population.

- Since voice packets experience both loss and delay over the Internet, an original frame-resequencing technique was then implemented with ILFR at the decoder. Based on these two techniques, packet size optimization was performed to determine the optimal packet size, P_{opt}, and Future Buffer size, B_{opt}, for transmission of G.723.1 voice packets over three multi-hop networks, each simulated under five rates of randomly distributed packet loss. By employing a divide-and-conquer strategy, the zero-delay, zero-packet-loss, and combined MOS sets for all allowable packet and Future Buffer sizes were first determined. The results from Shannon's Channel Coding Theorem were then applied to every simulated network and combined MOS set to determine P_{opt} and B_{opt}, leading to the following monotonically increasing relationships: First, as the number of hops is increased within a network, speech quality is maximized and delay is minimized by sending larger speech packets. Secondly, as level of network congestion is increased, the system's quality of service is again maximized by employing larger Future Buffers at the decoder.

8.2 Future Work

The results presented in Chapters 6 and 7 indicate that improvements can be made during the packet recovery and packet size optimization stages of this thesis. Based upon these observations, the following suggestions are made for future work which can lead to significant improvements in the frame-resequencing and ILFR techniques as well interesting applications based on their results:

- For all informal listening tests conducted in this thesis, a larger number of participants could be used to better assess the overall performances between G.723.1 error concealment and ILFR. Clearly, the results from a larger sample space of participants would generate more accurate mean opinion scores for the two techniques and thus provide a better measure of their relative performances.

- Different speech segment playouts are also necessary to better assess the quality of ILFR. Since the informal listening tests conducted in this thesis only used male-spoken speech segments, testing female and children-spoken speech segments are also necessary in order to improve the performance of the lost-frame recovery technique. As the harmonic frequency for females and children are higher than that of males on the average [13], this would result in a slight decrease in the estimated pitch lag threshold, P_{thresh}, used in the linear interpolation algorithm. By combining the informal listening test results based on male, female, and children speakers, a better value of P_{thresh} could thus be implemented for linear interpolation and lead to an improved technique that would be more applicable over a wider range of speakers.

- Another suggestion for future work involves experimenting with different methods of simulating packet loss and network delay. As presented in Chapters 6 and 7, the simulations performed in this thesis only applied a random number generator to the encoded speech stream to randomly select packets for discard and delay. In order to improve the robustness of the frame-resequencing and ILFR techniques, different channel simulations are necessary. For instance, one could initially determine the performance of the two techniques given a channel simulated with Gaussian and Poisson distributions of packet loss. Next, actual Internet traffic and packet loss statistics could be gathered from different Internet Service Providers (ISPs) to develop a model that accurately represents the Internet's dynamic congestion levels. By using these alternative approaches to simulate packet loss and network delay, the performance of the two techniques could be better assessed, and thus altered as necessary, in order to best maximize the system's quality of service over a wider range of network conditions.

- To truly assess the performance of frame-resequencing and ILFR, it is necessary to develop a real-time implementation of the proposed Internet Telephony system. Although RTP is not currently available within the public domain, the following uni-cast implementation could work equally well as a preliminary real-time application: The client and server systems, each with their own IP addresses, would be equipped with a microphone, loudspeaker, and SoundBlaster^{TM} card. First, software would be developed to perform analog-to-digital conversion and digital-to-analog conversion between the microphone and sound card and sound card and loudspeaker respectively. Using an Internet programming software development kit such as Microsft's Winsock^{TM} Application Programming Interface, software would then be developed to establish a UDP/IP connection between the client and server to send and receive sequentially-numbered speech packets. In essence, an RTP protocol environment would be simulated by prepending sequence numbers to the speech packets and working on top of

UDP. For simplicity, this real-time implementation could be first implemented over a LAN to test communication establishment. Next, packet loss could be simulated to test the effectiveness of the frame-resequencing and ILFR techniques. Finally, the system could be implemented over a WAN to assess the system's performance with the true congestion levels of the Internet.

- Based on the results from packet size optimization, an interesting application worth investigating is the development of a client and server that could dynamically change the packet size and Future Buffer size depending on the level of congestion detected within the Internet. For instance, 24-hour Internet traffic statistics could be collected from different ISPs and partitioned into five levels, each of which would correspond to the time of day during which the Internet experiences high levels of congestion, low levels of congestion, and intermediate levels of congestion. Given the Internet traffic statistics for a 24-hour day, the partitioning may show high congestion levels during the start, middle, and end of a typical business day (e.g., 8:30 A.M., 12:00 P.M. and 5:30 P.M.) when individuals may be most likely to use the Internet, and low congestion levels during the early hours of the morning (e.g., 2:00 A.M. to 6:00 A.M.) when individuals may be least likely to use the Internet. The intermediate congestion levels would be similarly determined. After partitioning the 24-hour Internet traffic statistics into five levels of congestion, the results from packet size optimization could then be applied to develop an Internet Telephony system that would continuously track the time of day and dynamically switch the packet and Future Buffer sizes for transmission, depending on the level of congestion detected at that time. Hence, larger packet and Future Buffer sizes may be used before the start and end of a typical business day while smaller packet and Future sizes may be used during the early morning hours. In this way, speech quality is maximized as the most efficient use of network bandwidth is achieved.

Bibliography

[1] A. Anderson. TCP/IP Networks. Downloaded from: http://www.theochem.uni-dusseldorf.de/network-guide/node7.html, March 1996.

[2] Cisco Systems, Inc. Voice over ATM in Wide-Area Networks. *White Papers*, 1996.

[3] Comsat Laboratories Systems Studies Group. Throughput Measurements for TCP/IP (Microsoft and SUN TCP/IP Stack) over Satellite Link., July 1997.

[4] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley & Sons, West Sussex, England, 1991.

[5] B. Garbee. Introduction to TCP/IP. Downloaded from: http://www.rheinneckar.-de/ranger/tcptutor.html, 1996.

[6] R. Gareiss. Voice over the Internet. *Data Communications*, September 1996.

[7] H. Gilbert. Planning TCP/IP. Downloaded from: http://pclt.cis.yale.edu/pclt-/winworld/plantcp.html, April 1995.

[8] ITU-T Study Group 12. *ITU-T Recommendation G.114: One-Way Transmission Time*, February 1996.

[9] ITU-T Study Group 15. *ITU-T Recommendation G.723.1: Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s*, March 1996.

[10] A. M. Kondoz. *Digital Speech: Coding for Low Bit Rate Communication Systems.* John Wiley & Sons, West Sussex, England, 1994.

[11] K. Y. Lee. *Analysis-by-Synthesis Linear Predictive Coding.* PhD thesis, University of Surrey, Guildford, Surrey, March 1990.

[12] D. E. McDysan and D. L. Spohn. *ATM: Theory and Application.* McGraw-Hill, New York, NY, 1995.

[13] C. McElroy. Speech coding. Downloaded from: http://wwwdsp.ucd.ie/speech/tutorial-/speech_coding/, November 1995.

[14] P. Nummi. TCP/IP Quick Primer. Downloaded from: http://home.ican.net/pjnummi-/systems_1.html, July 1996.

[15] D. Roberts. *Developing for the Internet with Winsock.* The Coriolis Group, Scottsdale, AZ, 1995.

[16] K. Savetz and A. Sears. How Can I Use the Internet as a Telephone? Downloaded from: http://www.northcoast.com/savetz/voice-faq.html, July 1996.

[17] H. Schulzrinne. Internet Services: From Electronic Mail to Real-Time Multimedia. In *Informatik Aktuell Series*, pages 21–34, Chemnitz, Germany, February 1995. KIVS'95 (Kommunikation in Verteilten Systemen).

[18] H. Schulzrinne. RTP: Overview. Downloaded from: http:/www.cs.columbia.edu/hgs-/rtp/toc.html, May 1997.

[19] H. Schulzrinne. *Internet Draft: Issues in Designing a Transport Protocol for Audio and Video Conferences and other Multiparticipant Real-Time Applications.* Audio/Video Transport Working Group within the Internet Engineering Task Force, May 1994.

[20] H. Schulzrinne. Internet Telephony: Towards the Integrated Services Internet. In *IEEE Workshop on Internet Telephony*, Utrecht, The Netherlands, February 1996.

[21] H. Schulzrinne. *RFC 1889: RTP: A Transport Protocol for Real-Time Applications.* Audio/Video Transport Working Group within the Internet Engineering Task Force, February 1996.

[22] R. Soheili. *Analysis-by-Synthesis Coding of Speech Signals at 8 kb/s and Below.* PhD thesis, University of Surrey, Guildford, Surrey, March 1993.

[23] W. Stallings. *Data and Computer Communications.* Prentice-Hall, Upper Saddle River, NJ, 5th edition, 1997.

[24] W. R. Stevens. *TCP/IP Illustrated*, volume 1. Addison-Wesley, Reading, MA, 1994.

[25] M. Taylor. Voice over ATM: A Sound Assessment. *Data Communications*, December 1996.

[26] A. R. Thryft. Voice over IP Looms for Intranets in '98. *Electronic Engineering Times*, 967:79, 102, August 1997.

[27] S. Yeldener. Comparison of Analysis and/by Synthesis Approaches in Low Bit Rate Speech Coding. Master's thesis, University of Surrey, Guildford, Surrey, September 1989.