THE PERCEPTUAL QUALITY OF MELP SPEECH OVER ERROR TOLERANT IP NETWORKS

Ben Gavula, George Scheets, Keith Teague, and Justin Weber

School of Electrical and Computer Engineering
Oklahoma State University
Stillwater, OK 74078

ABSTRACT

Modifications to IP based packet network protocols are examined that would make the network tolerant of bit errors in packet payloads or headers. These modifications are tested with communication quality MELP voice traffic. As measured by a PESQ score, improvements in the perceptual quality of the speech are noted that are maximized when error checking is disabled for the entire packet.

Index Terms— Vocoders, transport protocols, internet, error analysis, linear predictive coding

1. INTRODUCTION

Modern IP networks are increasingly being used to transmit multimedia traffic such as voice, music, and video. Many of the protocols involved in these transmissions are not optimized for this particular use as often the codec being used was originally designed for bit serial networks. One example is interactive speech communication over IP based wired or wireless networks. Many speech coders were designed with procedures that tolerate the presence of occasional transport network bit errors. Current IP standards ensure that time sensitive packets are transmitted uncorrupted or not at all, with the latter case resulting in significant loss of voice information that the sink codec may have trouble masking. The IP transport system is mismatched to the capabilities of such an error tolerant speech coding protocol. Instead of the network dropping all corrupt packets, were these delivered when possible to the speech codecs at the information sink, the inherent error tolerant capabilities of the codecs should allow the quality of the reproduced speech to be improved.

A transport layer solution to this problem was proposed in [1]. This method provides a checksum over the relevant data while ignoring errors elsewhere in the packet. Furthermore, through testing, packet drop rates for various networks were noted and possible improvements hypothesized for real-time PCM and MPEG applications. A link layer approach was tested in [2] that utilized a proposed 802.11 MAC extension to allow for errored packets to propagate the network. Specifically, speech was tested with the GSM AMR-WB coder and noticeable improvements in perceptual quality were achieved. Improvements in real-time multimedia by a means of error tolerant packet networks were also noted in [3] and [4].

This paper examines modifications that could be made to an IP network to allow for better use of the available bandwidth when transmitting Mixed Excitation Linear Prediction (MELP) speech [5][6] such as might be encountered in low-rate military, government, or telecom applications. A network having the option of keeping corrupted MELP traffic (as opposed to simply dropping an errored packet) would allow for improved voice quality as perceived by an end user.

2. MELP OVER AN ERROR TOLERANT NETWORK

2.1 Test network configuration

For the purposes of this testing, Ethernet was used for the physical and link layers, due to its prevalence in modern networking. Because of the real-time nature of interactive speech transmission, the real-time transport protocol (RTP), which allows the information sink to track dropped or out of order packets, was used in the standard IP/UDP/RTP format. In this testing, at the information sink receiver, an out of order or dropped packet was replaced by an appropriate amount of silence.

One issue associated with MELP coding is that each MELP speech frame is 54 bits in length, which would not end on an octet boundary [5]. Table 1 shows the bit allocation of MELP frames. Individual MELP frames could simply be padded out to the next octet and sent one MELP frame per packet, however, this would be wasteful considering the large number of network header bits. Instead, several MELP frames can be bundled together in a single packet. For the testing presented in this paper, a bundling factor of four MELP frames in each IP packet was used. In addition to keeping latency at a tolerable level, a bundling factor of four yields 216 data bits per packet, ensuring proper octet alignment. Also note that MELP frame boundaries are wholly contained within each packet.

Table 1 - Bit Allocation in a MELP Frame [5]

Parameters	Voiced	Unvoiced
Line Spectral Frequency	25	25
Fourier Magnitudes	8	-
Gain	8	8
Pitch	7	7
Bandpass Voicing	4	-
Aperiodic Flag	1	-
Error Protection	-	13
Sync Bit	1	1

2.2 Error tolerance

The MELP codec was originally designed for use on serial lines and includes built in capabilities for dealing with bit errors in the data stream [5]. Unfortunately, methods for sending speech data over modern IP networks include protocols that are not payload error tolerant. First, UDP has a checksum which covers its own header (and an IP pseudo-header) along with all the data being transported. Additionally, Ethernet also has a check sequence which covers every bit in the entire Ethernet frame. In either case, an error anywhere in the data would result in the entire packet being dropped. To maximize the voice quality at the information sink, MELP should be given the opportunity to process and conceal the occasional bit errors in the received voice frames. For this to occur, the packet transport network should ideally be able to selectively pass errored packets.

For UDP there already exists an elegant alternative in UDP Lite [6]. UDP Lite is very closely related to UDP. The only difference is that UDP Lite replaces the UDP Length field with a Checksum Coverage field. This replacement field specifies how much of the packet (starting from the beginning of the UDP header) is used in the checksum calculation. In fact, if the coverage field is set to the length of the packet, UDP Lite is identical to UDP.

For the link and network layers, there are not obvious ways to allow for error tolerance. At the Data Link layer, the IEEE 802.1 Logical Link Layer "Type" field might be assigned a specific value to alert a switching or end device of an error tolerant packet that should receive non-standard error processing. The same might be accomplished using the IPv4, Type of Service, Protocol, or Padding fields. Once standardized, error-tolerant-aware switches could be gradually inserted into a network in a backwards compatible manner such that a packet requesting error tolerant processing would receive the same at capable switches, and would receive normal processing at older unaware switches. Over time an entire system could be made error-tolerant-aware.

3. SIMULATION AND TESTING

3.1 Testing strategies

Several methods for creating an error tolerant network exist. This paper compares two strategies for accomplishing this task with the currently employed technique of dropping errored packets:

Strategy 1: Checksum all headers and data

This strategy is equivalent to existing Ethernet networks and is used as a baseline test for comparison.

Strategy 2: Checksum only header data

In this strategy, UDP Lite is used with its Checksum Coverage field set to 16. This amounts to covering the full UDP Lite header along with the first 32 bits of the RTP header. The RTP Timestamp and SSRC Identifier fields are not covered, because they are unused in our applications. Additionally, Ethernet is assumed to only calculate a checksum for its own header fields and nothing else. The IP checksum as defined in the IP specification is calculated only over the IP header itself and does not checksum any traffic [7]. In this way, all header fields are protected, but all speech data is allowed to accumulate errors.

Strategy 3: Ignore all checksums

In this strategy, hardware between source and receiver ignores all checksum calculations. This means errors are allowed to occur anywhere in the packet, including header fields. In this case, the packets would be lost or dropped only if "sensitive" parts of the packet are lost. We define sensitive parts to be fields that, upon error, will cause the packet not to be received by the receiving application. We outline four types of reasons why this may occur. Below is a list of these reasons, along with the fields that would cause them to occur:

1. Packet gets routed incorrectly
    - Ethernet Destination
    - IP Destination
2. Packet cannot be decoded correctly by network hardware
    - Ethernet Length
    - IP Version
    - IP Header Length
    - IP Total Length
    - UDP Length
    - RTP Version and Payload Type fields (first two octets)
3. Packets arrive at destination but fail to reach the running speech application
    - Ethernet Source
    - IP Protocol
    - IP Source
    - UDP Source Port
    - UDP Destination Port
4. Application receives the packet data, but cannot use it

- RTP Sequence Number (data appears out-of-order so is dropped)

Note that Strategy 3 has an unwanted side effect of causing additional useless traffic should errors occur in a packet header. Errors in the Ethernet destination address may cause flooding at an Ethernet switch if the errored address is not in the switch look-up table, while errors in the IP destination address will cause packets to be misrouted. Generally, errors in any header will result in an unusable packet occupying network resources.

3.2 Simulation environment

Table 2 shows the strategies along with their number of sensitive and non-sensitive bits when used in the test configuration specified in section 2.1 above. Notice that the number of total bits in each packet is 680. This was purposefully done for comparison purposes, even though more bits may be required to implement the particular strategy in a real network environment.

Table 2 - Packet makeup for various testing strategies

	Description	Sensitive bits	Non-sensitive bits	Total bits
Strategy 1	Drop on all errors	680	0	680
Strategy 2	Checksum only header fields	400	280	680
Strategy 3	Ignore all checksums	240	440	680

The above processes were simulated as follows. MELP frames were generated by a speech application and passed to a simulated stack for transport. Bit errors were generated using a uniform density pseudo-random number generator, thus resulting in a packet that had errors spread randomly throughout. No attempt was made to simulate burst losses, which are better characterized by using burst models at the packet level [8]. If any error occurred in a bit deemed sensitive, the simulated stack dropped the packet, and a blank MELP frame was inserted at the receiving end. Otherwise, the now possibly corrupted MELP frames were passed, with errors, on to the receiving application.

3.3 Speech quality metric

Because of the nature of speech and speech applications, the most important metric is the perceptual quality of the received speech. In this testing, we have used the ITU-T Recommendation P.862 "Perceptual Evaluation of Speech Quality"(PESQ) algorithm to evaluate the performance of each strategy under many different bit error rates. The algorithm compares the original speech samples to the received data samples to yield a metric comparable to perceptual speech quality, In several ITU benchmark experiments, PESQ was found to have a 90% correlation with Mean Opinion Scores [9]. For these tests we used sample speech data which was recorded at slightly better than telephone quality; 16 bit, 8 ksps. Speech consisted of multiple speakers reading a variety of test sentences. Two files were used for testing, one male and one female. Each consisted of about 20 seconds of speech which was made up of various test sentences spoken by various speakers.

4. RESULTS

Testing was performed in the simulation environment described above for each of the three strategies. Additionally, five different target bit error rates were chosen for testing (0.000%, 0.008%, 0.016%, 0.024%, and 0.032%). These bit error rates were chosen to achieve packet loss rates from zero up to a level high enough where received speech quality would be degraded far beyond tolerable levels (around 25% packet loss in the last case). Each of the 15 combinations of strategy and target Bit Error Rate (BER) was tested using 500 independent trials. This same simulation was repeated for both male and female speakers. The theoretical Packet Loss Rate (PLR) can be calculated given the actual BER as $PLR = 1 - (1 - BER)^{SB}$ where SB is the number of sensitive bits in the packet. For instance, given Strategy 2 (having 400 sensitive bits) and a BER of 0.025%, we can expect the PLR to be 9.517%. Figure 1 shows a plot of expected PLR versus BER.

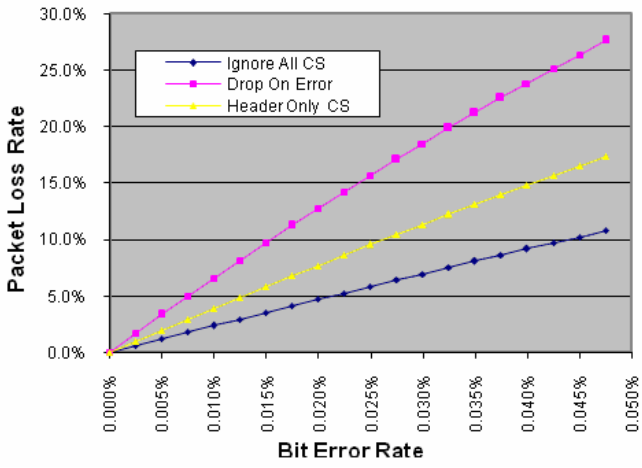

Figure 1 – Packet Loss Rate versus Bit Error Rate Comparison

As stated above, perceptual speech quality was measured here using the PESQ algorithm. Figures 2 and 3 show graphs of PESQ score versus BER for each strategy using male and female speakers respectively. Each point on the graph represents an average of the PESQ scores falling in a

particular experimental BER range. The scores for a particular BER could vary greatly depending on what parts of the data were corrupt, as dropped silence would have little effect on the score, whereas dropping the beginning of a word may make it unintelligible. As noted previously, in this study an out of order or dropped packet was replaced by an appropriate amount of silence.

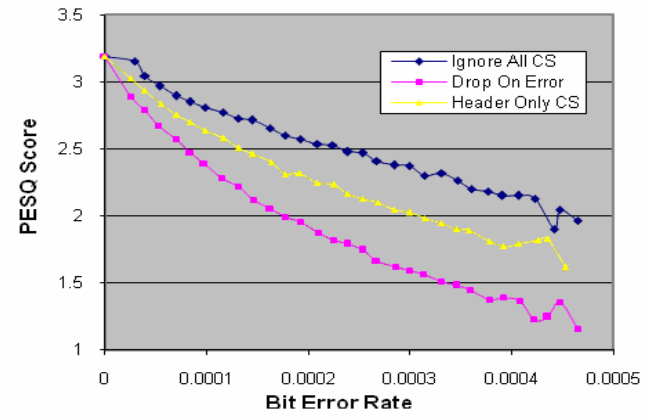

Figure 2 – PESQ Score versus Bit Error Rate Comparison, MELP Male

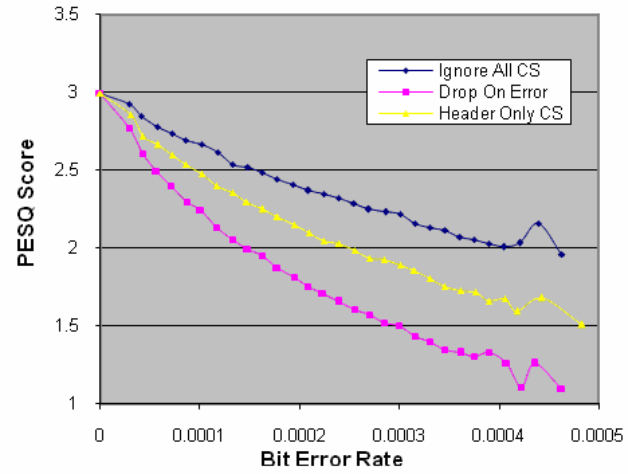

Figure 3 – PESQ Score versus Bit Error Rate Comparison, MELP Female

5. CONCLUSIONS

From the testing two things are immediately apparent. First, there was very noticeable improvement in PESQ score for the same bit error rate by keeping corrupt bits. And second, the more bits left unprotected, the better the PESQ score. One may notice that the graphs become less smooth at very high BERs. This is due to the fact that the highest target BER was only 0.032%. The data points that lie above that target represent the occasional experiment that experienced a higher than average number of bit errors. As the observed experimental BER increases, fewer experiments experienced that particular value and hence the plotted averages are based on a reduced number of sample experiments. Because of this, the variance of these experimental results is high.

Based on these tests it is apparent that some gain is to be had by keeping corrupt voice bits in packetized MELP speech data transmissions. The performance increase is, not surprisingly, minimal at low BER. However, it can be rather significant at high bit error and packet loss rates. This characteristic might make this technique very useful in the wireless realm, especially if the system is suffering degraded performance due to a low received signal strength or deliberate jamming. For this reason, wireless link layer protocols ought to be tested.

In general though, for MELP, and likely for many other voice and video compression algorithms, modifying packet network protocols such that packets could receive non-standard error handling has the potential of perceptually increasing the quality of the product delivered to an end user. Given the increased amount of voice and video flowing over packet networks, a thoughtful examination regarding the use of such techniques is warranted.

6. REFERENCES

[1] L.-A. Larzon, M. Degermark, and S. Pink, "UDP lite for real-time multimedia applications", *Proc. QoS mini-conference of IEEE Int. Conference on Communications*, Vancouver, Canada, June, 1999.

[2] Servetti, A., De Martin, J.C., "Error tolerant MAC extension for speech communications over 802.11 WLANs", *IEEE 61st Vehicular Technology Conference,* June 2005.

[3] Larzon, L.-A., Degermark, M., Pink, S., "Efficient use of wireless bandwidth for multimedia applications," *IEEE International Workshop on Mobile Multimedia Communications, 1999.*

[4] Minaburo, A.C., Toutain, L., et al, "Performance Improvement of Multimedia flows by using UDP-Lite and ROUC compression", *Fifth Intenational. Conference on Information, Com. and Signal Processing,* Dec. 2005.

[5] MIL-STD-3005, "Analog-to-Digital Conversion of Voice by 2,400 BIT/Second Mixed Excitation Linear Prediction (MELP)".

[6] Larzon, L., Degermark, M. et al, "The Lightweight User Datagram Protocol (UDP-Lite)", Internet proposed standard RFC 3828, July 2004.

[7] DARPA Internet Program, "Internet Protocol", Internet proposed standard RFC 791, September 1981.

[8] Jiang, W., Schulzrinne, H., "Modeling of Packet Loss and Delay and Their effect on Real-Time Multimedia Service Quality", *NOSSDAV 2000*, Chapel Hill, NC, June 2000.

[9] ITU-T, Recommendation P.862, "Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs", February 2001.