Forward Error Correction for Multimedia and Teleimmersion Data Streams

Ray Fang, Dan Schonfeld, Rashid Ansari, Jason Leigh
(ray@evl.uic.edu)
Electronic Visualization Laboratory, University of Illinois at Chicago
February 1, 2000

Abstract

Experimental results suggest that the quality of transmitting multimedia and Teleimmersion data streams over the Internet is affected by high packet loss rates. This makes it important to design mechanisms that minimize packet loss rate. Thus, error control is important in this case. In this report, an error control mechanism using forward error correction (FEC) is implemented and evaluated.

1. Introduction

Two different approaches can be used to deal with the transmission error in the networks. One is Automatic Repeat Request (ARQ), and another one is Forward Error Correction (FEC). Most of common protocols (HDLC, TCP/IP, etc) are using ARQ to ask for retransmission of the lost data packets. However, in the case of distributing real-time multimedia data, the ARQ mechanism will result in considerable delay and jitter which are not allowed in such applications. While the traditional FEC methods mainly focus on the correction of bit errors, on high-speed networks, especially on fiber networks, bit errors rarely occur. For an example, on fiber networks, the Bit Error Rate (BER) is only 10^{-9}. The main data loss comes from whole packet loss in the ATM switch queue buffer[1], or in the end-site device's buffer.
In this report, a FEC method is introduced to recover from packet loss with minimum overhead for multimedia data transmission.

2. A Simple FEC Scheme for Packet Loss

2.1 Motivation

For long distance networks like international networks, latencies are high (on the order of hundreds of ms)[3]. This can severely impact real-time interactive applications such as Teleimmersion or remote operation. Hence a scheme is needed to transmit data reliably over long distances without requiring the acknowledgement typically used in protocols such as TCP. FEC provides a promising solution to the problem in that errors are corrected at the end point without the need to wait for the retransmission of a packet.

The traditional reason for choosing ARQ as the main error correction used by many reliable protocols is that the FEC may introduce considerable computational overhead, and will also increase the bandwidth requirements. Thus, it is important to choose an FEC method that can achieve loss recovery while minimizing computational overhead. The

most suitable FEC scheme will depend on the nature of the data being transmitted[2]. This is further elaborated in Section 2.3.

2.2 Generating FEC Redundancy for Packets

There are several guidelines for generating FEC redundancy for real-time environments:

1. Do not use very complex mathematic operations to generate the redundancy. Make sure the computational time is less than the retransmission time.
2. Use the adjacent packets to generate the redundancy. Using packets far away from each other (For example, generating from packet 1, 10, and 20) will result in more delay, an increase in the requirements for the buffer both at the sender and receiver, and an increase in the complexity of buffer management.

Based on the above considerations, the FEC scheme provided in this report is as follows:

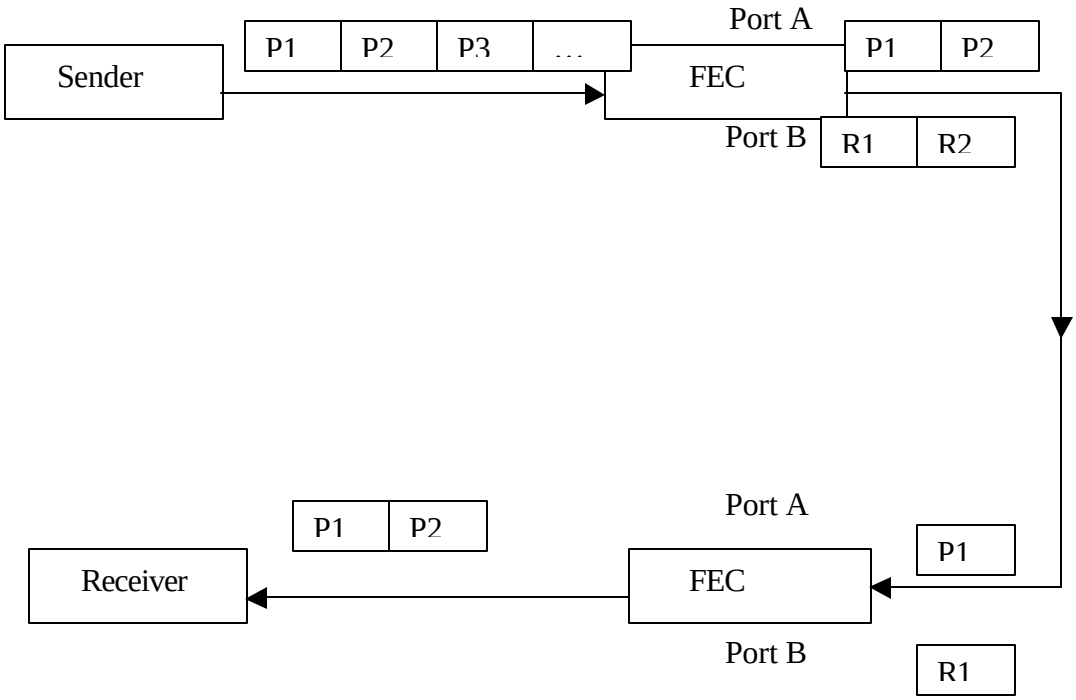


Fig1. The FEC scheme

1. Create the redundant packets based on the adjacent packets.
2. The data packets are sent through port A by using UDP protocol, while the redundancy packets are sent through port B by using UDP protocol.
3. In the receiver site, two receiver buffers (buffer A and buffer B) are prepared according to the port A and port B, respectively.
4. In the good conditions, there is no packet loss. The redundant packets in the buffer B will be ignored. Thus, in this case, the scheme is similar with the normal UDP except the very small overhead of processing redundant packets.

5. In the bad conditions, packet loss occurs. The scheme will try to go to the buffer B to look for appropriate redundant packets. If succeed, the lost packet will be put back in the buffer A with the right order. If failed, the lost packet cannot be reconstructed. The real packet loss occurs.

2.3 FEC Types

The data to be transmitted can be divided into three priorities:

Priority 1: The data that may not be lost during the transmission.
Priority 2: The data can be lost, but the loss will definitely affect the quality of service.
Priority 3: The data whose loss will slightly affect the quality of service.

According to the requirement of data importance, three types of FEC are given as follows:

- Type 1 (for Priority 1): Use higher redundancy- for example, produce one redundant packet for each one transmitted, or one for every two, or two for every three.
- Type 2 (for Priority 2): Use lower level of redundancy- for example, produce one redundancy packet for every three, or one for every four, etc..
- Type 3 (for Priority 3): Perform no FEC.

All of the operations are binary addition (+) and multiplication (•).
Let matrix G be coding generator. Encoding is carried out by evaluating

$$V = U \bullet G$$

Where V is the output code word and U is the information word. For an example, G is defined as

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$U = [P1 \;\; P2 \;\; P3 \;\; P4]$$

Where P1,P2,P3 and P4 are data packets.

Thus, the output will be

$$V = [P1 \;\; P2 \;\; P3 \;\; P4 \;\; P1+P2+P4]$$

Where the first four elements are the original data packets and the last one is the FEC redundancy packets R = P1+P2+P4.
Another issue which needs to be considered is the distance between those data packets in the operation. Normally, the adjacent packets will be chosen. For an example, the redundant packet R can be achieved by R = P1 + P2 + P3. Thus, the one-packet loss can be recovered (e.g., P2 = P1 + P3 + R). However, in the network which has high burst errors, the adjacent packets may be lost (e.g., P1, P2 are missing). The two-packet loss

can not be recovered. In this case, the distance will be increased somehow. For an example:

The redundant packet R = P1 + P2 + P3 will be replaced with:

R1 = P1 + P8 + P16 R2 = P2 + P9 + P17

Thus, P1, P2 can be recovered by applying
P1 = R1 + P8 + P16 P2 = R2 + P9 + P17

However, increasing the distance between packets will definitely increase the time delay in the real time transmission. Thus, different FEC scheme will be chosen by the negotiation between users and server before transmitting data.

3. Experiments

3.1 Experimental Environments

The experiments are done over the link between SARA and EVL, which is from Europe to America. The sender and receiver are SGIs running with IRIX 6.5 or above. The machines' details are as follows:

Zbox (zbox.evl.uic.edu):
Processors: 14 Processors (150MHZ IP19)
RAM: 512MB
OS: IRIX 6.5

Unite (unite.sara.nl):
Processors: Cray Origin2000 128 Processor (250MHZ IP27)
RAM: 57344 MB
OS: IRIX 6.5.4

The traceroute between Zbox and Unite is:

```
1 eecsevl.gw.uic.edu(131.193.48.1) 1 ms(ttl=64!)  1 ms (ttl=64!)  1 ms(ttl=64!)
2 eecs.gw.uic.edu (131.193.32.1)  1 ms 1 ms 1 ms
3 batm-16.gw.uic.edu (128.248.120.16)  2 ms 1 ms 1 ms
4 BR1.NewYork.surf.net (145.41.0.37)  17 ms 17 ms 18 ms
5 BR2.Newyork.surf.net (145.41.7.106)  17 ms 18 ms 17 ms
6 BR2.Amsterdam.surf.net (145.41.7.109)  95 ms 95 ms 96 ms
7 BR7.Amsterdam.surf.net (145.41.7.145)  92 ms (ttl=250!)  96 ms(ttl=250!)  92 ms
(ttl=250!)
8 sara-r3.rtr.sara.nl (145.41.10.42)  95 ms 95 ms 96 ms
9 rsm-sara-r1.rtr.sara.nl (145.100.5.12)  100 ms 97 ms 96ms 10 unite.sara.nl
(145.100.19.2)  98 ms 95 ms 98 ms
```

3.2 Experiment 1

Data packets with the same size will be transmitted by in three different ways: TCP, UDP, and FEC over UDP. In FEC over UDP, port 6000 and 6001 are used to send data packets and redundancy packets, respectively. During the transmission, no other data are

sent over the network. The packet sending frequency is approximately 2~3 ms. A redundant packet is created from every 3 data packets.

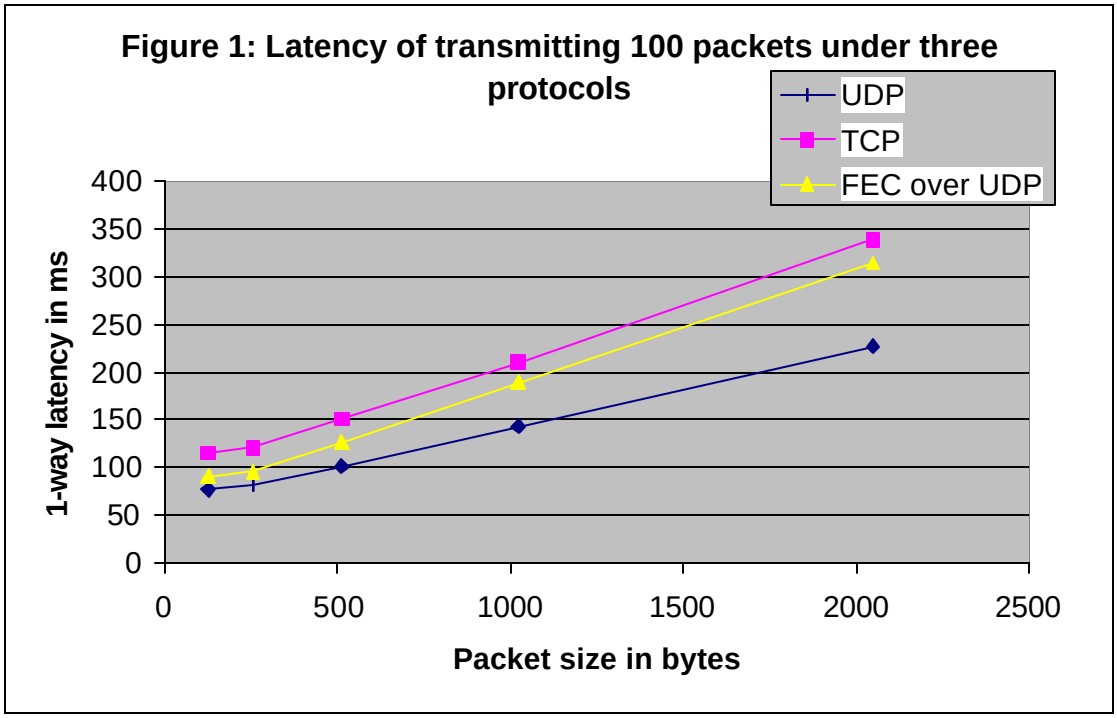
The results are as follows:

	UDP	TCP	FEC over UDP
Packet Size = 128B	77.0ms	115ms	90.3ms
256B	81.7ms	121ms	95.3ms
512B	101.0ms	150.8ms	126.0ms
1024B	143.0ms	210ms	189.0ms
2048B	227.3ms	339ms	314.3ms

Table 1. The latency of transmitting 100 packets of varying packet sizes over UDP, TCP and FEC over UDP. This data is plotted in Figure 1.

	Min (ms)	Max (ms)	Mean (ms)	Standard Dev.
UDP	121.0	155.1	143.5	4.09
TCP	181.9	239.0	211.5	10.95
FEC over UDP	165.2	210.0	190.0	6.81

Table 2. Statistical data for transmitting 100 packets (packet size = 1024B)



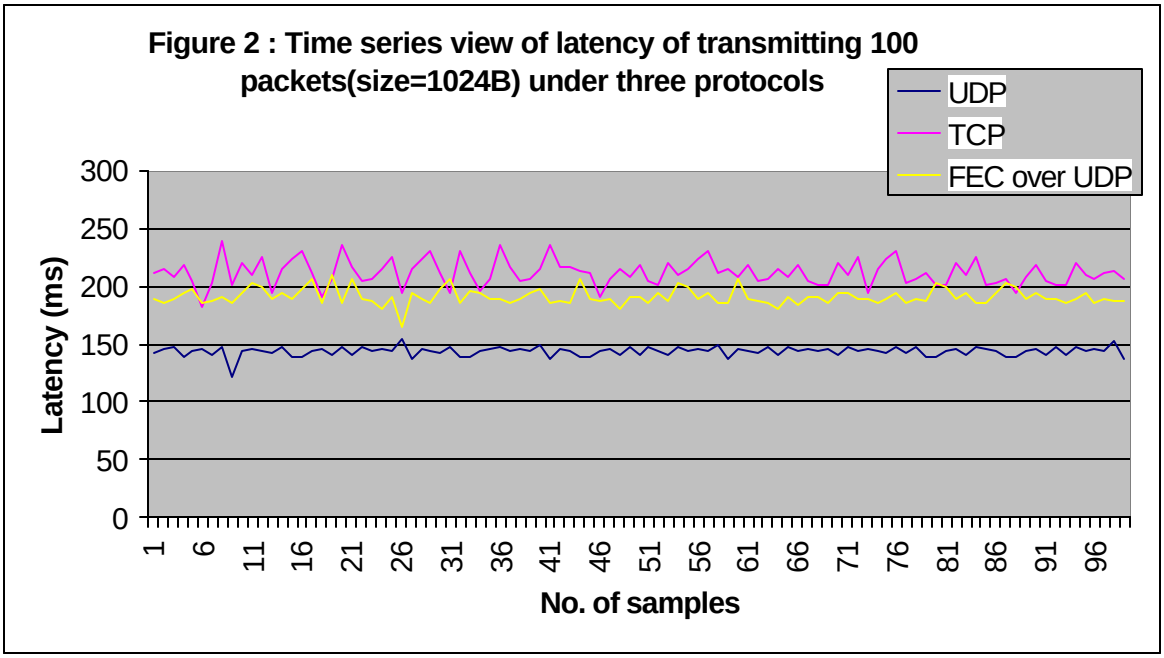


Figure 2 : Time series view of latency of transmitting 100 packets(size=1024B) under three protocols
UDP
TCP
FEC over UDP
Latency (ms)
300
250
200
150
100
50
0
1
6
11
16
21
26
31
36
41
46
51
56
61
66
71
76
81
86
91
96
No. of samples

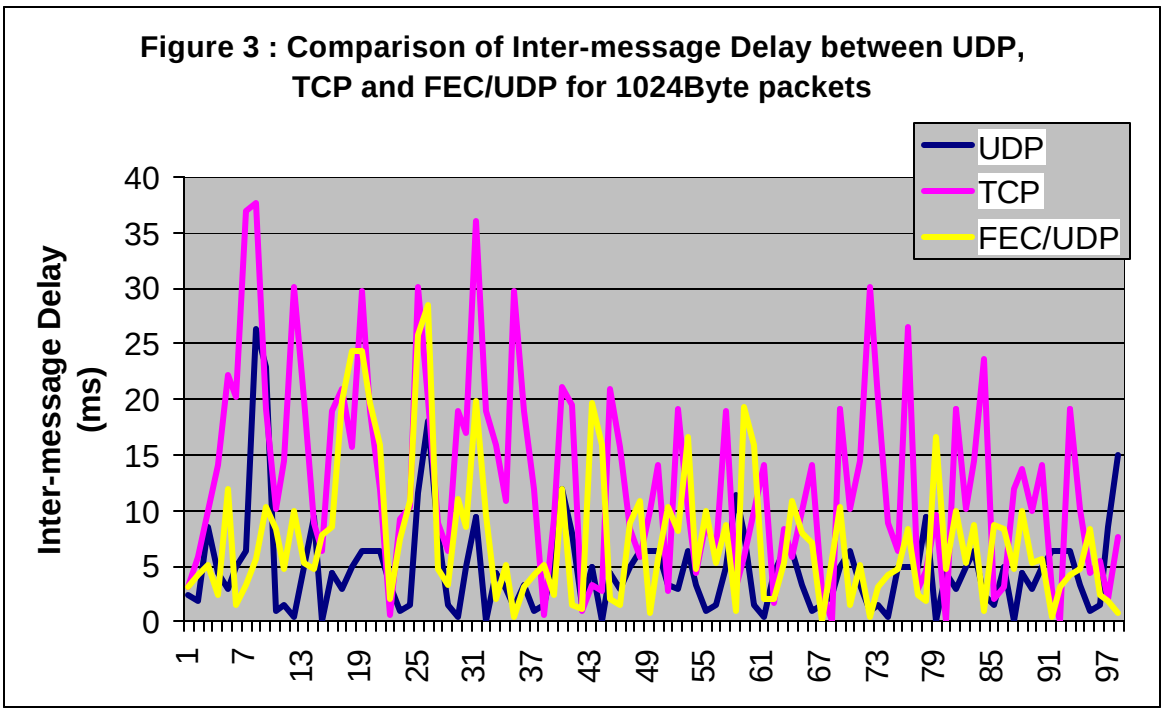


Figure 3 : Comparison of Inter-message Delay between UDP, TCP and FEC/UDP for 1024Byte packets
UDP
TCP
FEC/UDP
Inter-message Delay (ms)
40
35
30
25
20
15
10
5
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97

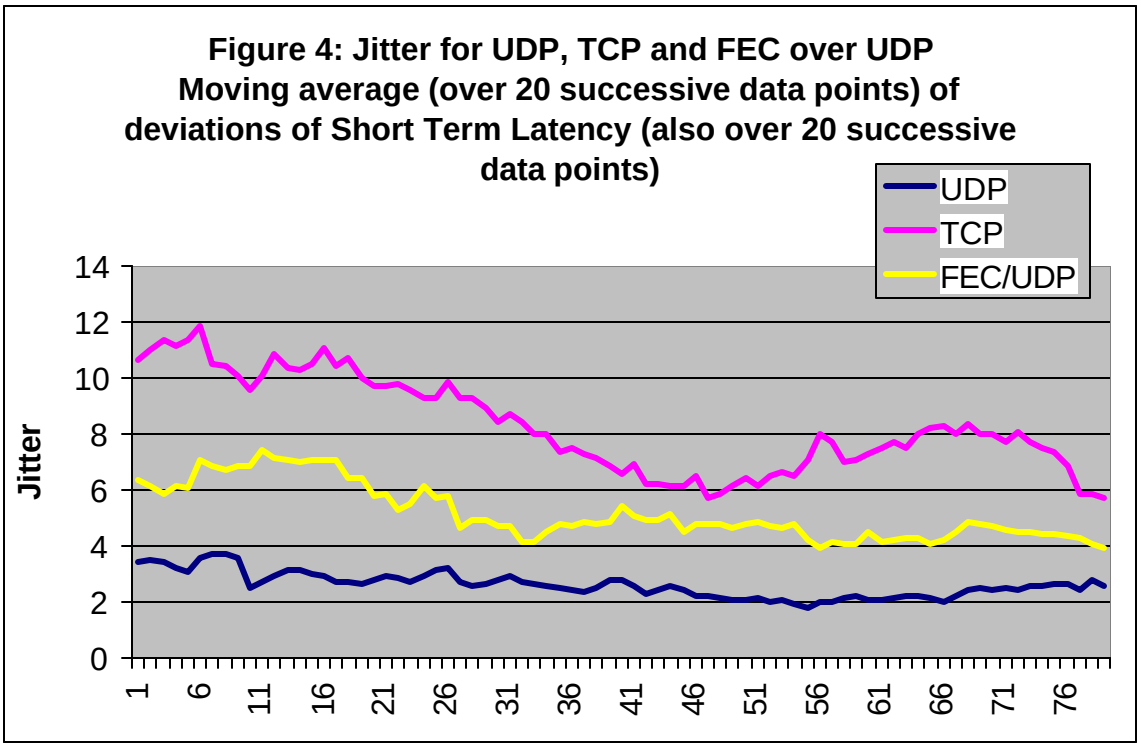


Figure 1 shows that our FEC scheme derives its greatest benefit when packet sizes are small as larger packet sizes incur additional buffer processing time. Figure 3 and 4 show that FEC also introduces jitter in the data stream. Jitter in Figure 4 is computed by first calculating the short term latency over every 20 data points and then computing the average deviation of the instantaneous latency as compared to the short term latency.

3.3 Experiment 2

Experiment 2 repeats Experiment 1 except using an additional UDP traffic generator to generate background traffic during the testing. The traffic created by traffic generator approached 10Mbps. However, there is a large loss rate when traffic approached 10Mbps. (The reason for this is unknown yet, initially we think it may be due to local buffer overflow.)

The results of this experiment generated data with little noticeable difference to those generated in the first experiment. We believe this is because we were not able to send enough background traffic over the 40Mbps link between EVL and SARA to impact the main experimental traffic.

3.4 Experiment 3

In this experiment packets were sent at a high rate between EVL and SARA.
Regular data packets were sent over port 6000, and redundant packets were sent over port 6001. The data rates of the regular data packets were tested at both 1Mbps and at 10Mbps. The redundant packets were generated from groups of either 3 or 5 data packets, respectively.

The results are tabulated as follows:

Data Rate	Packet Size	Packet Loss Rate in UDP	Packet Loss Rate in FEC over UDP
1Mbps	128B	0.4%	0%
1Mbps	256B	0.2%	0%
1Mbps	1024B	0.2%	0%
10Mbps	128B	30%	4%
10Mbps	256B	25%	3%
10Mbps	1024b	21%	1.5%

Table 3. Packet loss rate between UDP and FEC over UDP

From the above table, we can see that using FEC over UDP will reduce the packet loss rate. And as for the high packet loss at the 10Mbps rate, we believe that this may be due to an overflow of the receiver's UDP buffer. In any case FEC was able to significantly correct for the loss. Furthermore it was noted that the loss rates were higher for smaller packet sizes. This may have been because smaller packet sizes increased the number of packets sent hence incurring additional packet overhead.

4. Conclusions

From the three experiments, we can see that using FEC over UDP in a real time data transmission can reduce packet loss and can provide lower latency and jitter than TCP. FEC's benefit appears to be greatest with small packet sizes.

Future work will focus on the following:

- Refine the FEC scheme to further minimize redundancy requirements, data loss, latency and jitter.
- Re-examine FEC under high background traffic situations.
- Examine how FEC is affected by DiffServ congestion avoidance algorithms.

References

[1] Ernst W. Biersack, "Performance Evaluation of Forward Error Correction in ATM networks", Proc. of ACM SIGCOMM 92, pp.248-257, Baltimore, August 1992.

[2] E. W. Biersack, "A simulation Study of Forward Error Correction in ATM networks", Computer Communication Review, 22(1):36-47, January 1992.

[3] J. C. Bolot, "End-to-End Packet Delay and Loss Behavior in the Internet", Proc. of ACM SIGCOMM 93, pp 289-298, Ithace, September 1993.