2000년 한국정보처리학회 추계 학술발표논문집 제7권 제2호

이동 컴퓨팅 환경에서 패킷 손실 특성을 고려한 실시간 데이터의 동적 부가 전송 방법†

오 연 주° · 백 낙 훈 · 임 경 식
경북대학교 컴퓨터과학과
yjoh@ccmc.knu.ac.kr, nhbaek@ee.knu.ac.kr, kslim@knu.ac.kr

A dynamic additive transmission scheme of realtime data based on packet loss characteristics in mobile computing environment

Yeunjoo Oh°, Nakhoon Baek, Kyungshik Lim
Dept. of Computer Science, Kyungpook National University.

요 약

본 논문은 이동 컴퓨팅 환경에서 실시간 데이터의 전송시 발생하는 패킷 손실에 관한 문제를 다룬다. 현재 인터넷 상에서의 실시간 데이터 전송은 주로 RTP/RTCP 프로토콜[1]을 이용하는데, 이 프로토콜은 안정된 전송을 보장받지 못하며, 따라서 패킷 손실을 피할 수 없다[2,3]. 특히, 제한된 무선 대역폭과 높은 BER(bit error rate) 특성을 갖는 이동 컴퓨팅 환경에서는 기존의 인터넷보다도 더 많은 패킷들이 손실될 수 있다. 본 논문에서는, 이동 컴퓨팅 환경에서 실시간 데이터 전송시 발생되는 패킷 손실 특성을, 길버트 모델에 기초하여 확률적으로 분석하고, 이를 기반으로 새로운 복구 방법을 제안한다. 제안하는 실시간 데이터의 복구 방법에서는 패킷에 부가하는 잉여 데이터의 개수를 가변적으로 조절함으로써, 사용 중인 네트워크의 패킷 손실 특성을 반영할 수 있다. 특히, 잉여 데이터들의 오프셋 값들을 비연속적으로 설정함으로써, 간헐적이거나 연속적인 패킷 손실 모두에 대처할 수 있는 특징이 있다.

1. 서 론

최근 인터넷뿐만 아니라 이동 컴퓨팅 환경에서도, 멀티미디어의 송·수신이나 음성 통신과 같은, 실시간 데이터의 전송에 대한 수요가 증가하고 있다. 이동 컴퓨팅 환경은 유선망에 비해 제한된 무선 대역폭과 단말기의 이동이나 무선 링크의 특성에 따른 가변적 트래픽 변화를 가진다. 따라서, 유선망에 비해 전체 패킷 손실률과 패킷의 연속 손실률이 비교적 크다. 그러므로, 이동 컴퓨팅 환경에서 실시간 데이터를 전송할 때에는, 급격한 트래픽 변화 및 비교적 높은 패킷 손실률에 적응성을 가지는 손실 패킷 복구 방법이 필요하다.

본 논문에서는 이동 컴퓨팅 환경에서의 특성을 고려하여, 실시간 데이터 전송 시에 발생하는 패킷 손실 특성을 확률적으로 분석하고, 이에 기초한 실시간 데이터 복구 방법을 제안한다. 제안하는 방법은 기존의 RTP/RTCP를 기반으로, 패킷 손실률이 가변적으로 변화하는 환경 및 다양한 패킷 손실의 특성에 대해서도 효율적으로 동작하도록 설계되었기 때문에 이동 컴퓨팅 환경과 같은 무선망 환경은 물론 유선망 환경에서도 적용이 가능하다.

2. 관련 연구

RTP는 페이로드(payload) 타입에 따라 다양한 형태의 실시간 멀티미디어 데이터를 전송할 수 있도록 설계되었다. RTP 명세에 함께 정의된 RTCP는 RTP 데이터의 전송을 제어하거나 관련 정보를 제공할 수 있도록 설계되어 있다. RTP의 실제 사용에 있어서는 주로 UDP를 하위 계층으로 사용하기 때문에, RTP 자체는 안정적인 전송을 보장받지 못한다. 특히 이동 컴퓨팅 환경에서는 낮은 전송률과 높은 BER의 링크 특성을 가지기 때문에, 실시간 데이터의 전송시 전체 패킷 손실률이 높아지는 것은 물론, 연속적인 손실의 경우도 증가할 것이다[4]. 따라서, 이러한 패킷 손실 특성들에 따른 적절한 복구 방법이 요구된다.

손실된 패킷을 복구하기 위해 적용될 수 있는 대표적인 방법으로는 ARQ(Automatic Repeat Request) 방식, FEC(Forward Error Correction) 방식, 잉여 정보를 이용한 부가 전송(redundant data) 방식, 인터리빙(interleaving) 방식 등이 있다. ARQ 방식은 손실된 패킷을 송신측에서 재전송하는 것을 기반으로 하는 폐루프 메카니즘(closed-loop)이다. 따라서, 완전한 복구를 지원한다는 장점이 있지만, 재전송으로 인한 지연때문에 실시간 데이터 복구 방법으로는 부적합하다[2,4,5]. FEC 방식은 최소한의 지연만으로 에러를 복구할 수 있지만, 패킷 손실이 연속적으로 발생할 때에는 복구율이 높지 않다는 단점이 있다[2,4,5]. 잉여 정보를 이용한 부가 전송 방식은 송신 측이 별도의 시간 지연 없

† 본 연구는 정보통신부 초고속정보통신 응용기술개발사업 지원으로 수행되었음.

이 패킷을 전송할 수 있고, 수신 측에서는 최악의 경우에도 다음에 도착할 몇 개의 패킷만을 기다리면 복구가 가능하다는 장점이 있다. 반면에 같은 데이터의 중복 전송에 따른 시스템의 추가 처리 시간과 전송량이 증가하고, 연속적인 패킷 손실이 많은 경우에는 복구율이 높지 않다는 단점이 있다[2,5]. 인터리빙 방식은 연속적인 패킷 손실을 분산시키는 효과는 가지지만, 전송을 위한 처리 과정과 수신 후 재배열하는 과정에서 상당한 지연을 야기시킨다[4]. 이 사실은, 특히 실시간 대화형 응용들에서 사용되기에는 부적합한 원인으로 지적되고 있다[6,7].

3. 실시간 데이터의 손실 복구 방법 설계 및 구현

3.1 제안된 손실 복구 방법의 특징

이동 컴퓨팅 환경에서는 전체 패킷 손실률과 연속 패킷 손실률이 유선망에 비해 비교적 큰 특성을 가진다. 손실률이 클수록 더 많은 잉여 데이터를 사용하는 것이 효율적이지만, 네트워크의 대역폭을 감안하면 어느 한계 이상으로 잉여 데이터의 개수를 늘릴 수는 없다. 따라서, 본 논문에서는 최대 4개의 잉여 데이터를 사용한다. 또한, 오프셋 값들을 -1, -2, -4, -8과 같이, 비연속적으로 설정한다. -1, -2의 오프셋 값들은 간헐적인 패킷 손실을 복구할 수 있는 기능을 수행한다. 반면에, -4, -8의 비교적 큰 오프셋 값들은 연속적인 패킷 손실의 복구를 가능하게 한다. 패킷의 연속적인 손실에 대비한다는 측면에서는 더 큰 오프셋 값이 유리하지만, 데이터의 실시간 처리를 보장하기 위해서는 어느 한계치 이상의 오프셋 값들은 사용이 불가능하다.

본 제안 방법에서는 위의 조합들 중에서, 총 5가지 경우의 잉여 데이터 부가 방식을 설정하고, 이들 각각을 R0-Method, R1-Method, R2-Method, R3-Method, R4-Method라고 표현한다. R0-Method는 잉여 데이터를 사용하지 않음을 뜻하고, R1-Method, R2-Method, R3-Method, R4-Method는 각각 (-1), (-1, -2), (-1, -2, -4), (-1, -2, -4, -8) 오프셋의 잉여 데이터들을 부가한다. 특히, R3-Method와 R4-Method에서는 비교적 큰 오프셋의 잉여 데이터를 부가함으로써, 인터리빙에 의한 전송 효과 즉, 연속적인 패킷 손실률의 감소 효과를 가진다. 연속적인 패킷 손실의 특성은 전체 패킷 손실률과 밀접한 관계를 가지므로, 오프셋의 조합을 선택할 때, 간헐적으로 발생하는 패킷 손실을 복구하기 위한 -1, -2의 오프셋들이 항상 포함되어야 한다.

한편, 잉여 데이터를 이용하여 복구하는 경우에는 패킷 자체의 손실보다는 잉여 데이터를 이용하더라도 복구하지 못하는 손실률이 더 중요하다[5]. 특히, 간헐적인 패킷 손실의 경우에는 비록 패킷이 손실되더라도 잉여 데이터를 이용하여 원시 데이터의 복구가 가능한 경우가 많으므로, 단순히 패킷의 손실률만을 따지는 것은 한계를 가진다. 따라서, 본 논문에서는 전체 패킷 손실률 외에, 특정 데이터가 잉여 데이터를 이용하더라도 복구되지 못하는, 완전 손실률을 고려하여 어떤 형태의 잉여 데이터를 사용할 것인지를 결정한다. 각각의 방식이 어떤 상황에서 사용될 수 있는지를 분석하기 위해서는 완전손실률의 계산이 필요한데, 이 계산시에 길버트 모델이 사용될 수 있다.

3.2 손실 특성 분석을 위한 확률 모델 도입

길버트 모델은 패킷의 손실 여부를 2-state 마코프 체인(Markov chain)으로 설명한다. 즉, 임의의 패킷이 무사히 수신자 측에 도달하는 스테이트(state-0)와, 반대로 패킷이 손실되는 스테이트(state-1)가 존재한다. 이 때, 패킷 손실 특성은 각 스테이트 간의 전이 확률 p와 q로 표현된다. n번째 패킷이 어느 스테이트에 있는지를 표현하는 확률 변수를 X_n이라 하면, $X_n=0$과 $X_n=1$은 각각 n번째 패킷이 수신자 측에 도착하거나 손실됨을 의미한다. 따라서, 전이 확률 p, q는 확률 변수 X_n을 이용하여 다음과 같이 표현할 수 있다.

$$p = \Pr[X_{n+1}=1 | X_n=0] \quad (1)$$

$$q = \Pr[X_{n+1}=0 | X_n=1] \quad (2)$$

연속된 패킷들이 특정한 스테이트를 계속 유지할 확률은 전이 확률 p, q에 대한 기하 분포를 보인다[5]. 따라서, 임의의 패킷이 state-0 또는 state-1에 있을 확률은 기하 분포의 특성에 따라, 다음과 같이 계산된다.

$$\Pr[X_n=0] = \frac{q}{p+q} \quad (3)$$

$$\Pr[X_n=1] = \frac{p}{p+q} \quad (4)$$

길버트 모델에서는 전이 확률을 이용하여, 간헐적인 패킷 손실은 물론, 연속 손실의 확률도 구할 수 있다. 즉, n, n+1, n+2의 3개 패킷이 연속적으로 손실될 확률은 다음과 같이 계산할 수 있다.

$$\Pr[X_n=1, X_{n+1}=1, X_{n+2}=1] = (1-q)\cdot(1-q)\cdot\frac{p}{p+q} = (1-q)^2\frac{p}{p+q} \quad (5)$$

따라서, 5가지 잉여 데이터 부가 방식들에 대한 완전 손실률을, 길버트 모델에 근거하여 계산하면 <표 1>과 같다.

<표 1> 잉여 데이터 부가 방식들의 완전 손실률

방 식	완전 손실률
R0-Method	$LossR0 = \Pr[X_n=1] = \frac{p}{(p+q)}$
R1-Method	$LossR1 = \Pr[X_n=1, X_{n+1}=1] = (1-q)\cdot\frac{p}{p+q}$
R2-Method	$LossR2 = \Pr[X_n=1, X_{n+1}=1, X_{n+2}=1] = (1-q)^2\cdot\frac{p}{p+q}$
R3-Method	$LossR3 = \Pr[X_n=1, X_{n+1}=1, X_{n+2}=1, X_{n+4}=1] = \{(1-q)^2+pq\}\cdot(1-q)^2\cdot\frac{p}{p+q}$
R4-Method	$LossR4 = \Pr[X_n=1, X_{n+1}=1, X_{n+2}=1, X_{n+4}=1, X_{n+8}=1] = \{(1-q)^4 + pq\cdot(6-4p+p^2-8q+3pq+3q^2))\cdot\{(1-q)^2+pq\}\cdot(1-q)^2\cdot\frac{p}{p+q}$

<표 1>에서 보는 바와 같이, 각 잉여 데이터 부가 방식들의 완전 손실률은 잉여 데이터의 개수가 많아짐에 따라, 그 손실률이 단조 감소하는 특성을 가진다. 즉, 전이 확률 p, q 값들이 0에서 1 사이의 값을 가지

면, 항상 다음 관계식을 만족시킨다.

$$LossR0 \geq LossR1 \geq LossR2 \geq LossR3 \geq LossR4 \quad (6)$$

3.3 알고리즘

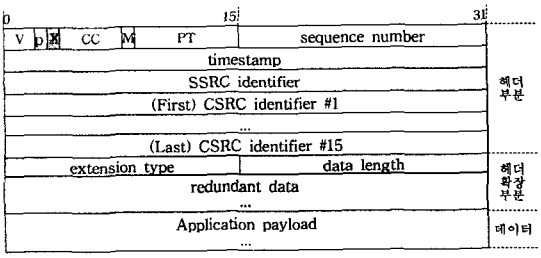
전체 알고리즘은 송신자 측과 수신자 측에서의 동작으로 구성된다. 송신자 측은, 주기적으로 피드백된 수신자 측의 손실 정보에 따라, 동적으로 부가 방식을 선택 및 적용함으로써 네트워크 상에서의 패킷 손실 특성이 변화하더라도 동적인 대처가 가능하다.

· RTP/RTCP 패킷의 확장

제안된 복구 방법에서는 실시간 데이터를 전송할 때, 잉여 데이터를 부가 전송하기 위한 방법과 수신자 측에서 전이 확률 p, q 값을 송신자에게 보낼 수 있는 방법이 필요하다. 이를 위해 RTP 데이터 패킷과 RTCP APP 패킷을 이용한다.

RTP 데이터 패킷은 전송되는 실시간 데이터 이외에도 부가적으로 필요한 정보들을 헤더 확장(header extension) 부분에 넣을 수 있도록 설계되어 있다[1]. (그림 1)은 RTP 헤더 확장 부분에 잉여 데이터를 넣은 형식을 보여준다. 하나의 RTP 데이터 패킷은 선택된 부가 방식에 따라, 헤더 확장 부분이 없거나 80~320 바이트의 잉여 데이터를 가지도록 하였다. 송·수신자 측은 RTP 헤더 확장 부분을 이용하여 잉여 데이터를 전송하거나 받는다. RTP 헤더 확장 부분에서 사용된 필드들은 다음과 같다.

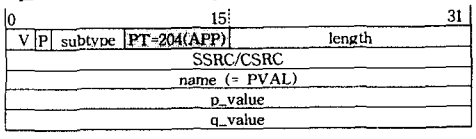
· extension type: 잉여 데이터 부가 방식의 종류(1~4).
· data length : 잉여 데이터 블록의 크기.
· redundant data: 잉여 데이터가 저장되는 블록.



(그림 1) 확장 헤더 부분에 잉여 데이터를 넣은 RTP 패킷 형식

APP 패킷 내부의 포맷은 응용 프로그램에서 설정할 수 있으므로, 수신자 측은 (그림 2)와 같은 형식으로 확장한 APP 패킷을 송신자 측에게 보낸다. APP 패킷 내부에서 사용하는 필드들은 다음과 같다.

· name : 수신 상태에 대한 확률 정보임을 표시.
· p_value : 전이 확률 p의 값을 기록하는 필드.
· q_value : 전이 확률 q의 값을 기록하는 필드.



(그림 2) 확장된 RTCP APP 패킷 형식

· 송신자 동작 알고리즘

송신자 측은 수신자 측으로부터 RTCP APP 패킷을 받게 되면, 패킷에 들어있는 p, q 값들을 이용하여 각 잉여 데이터 부가 방식의 완전 손실률을 계산한다. 이 계산된 값들 중에서 손실률 허용 임계값 α보다 작은 값을 갖는 잉여 데이터 부가 방식을 선택한다. 임계값 α는 네트워크 환경과 트래픽 및 실시간 데이터의 특성을 고려하여 송·수신자 측 사이에 결정된, 손실률에 대한 최대 허용치이다. 만일 두 개 이상의 잉여 데이터 부가 방식이 α보다 작은 값을 가지는 경우에는, 부가하는 잉여 데이터의 개수가 적은 쪽을 선택한다. 이렇게 함으로써, 완전 손실률은 α보다 작도록 유지하면서도, 패킷의 크기를 감소시킴으로써 네트워크 대역폭 증가를 최소화한다. 최종 선택된 부가 방식은 수신자 측으로부터 다음 번 RTCP APP 패킷을 전달받을 때까지 수행된다. (그림 3)은 송신자 측이, 패킷 손실 특성 p, q 값에 따른 최적의 잉여 데이터 부가 방식을 선택하는 알고리즘이다. 잉여 데이터 부가 방식의 선택에 있어서는, 식 (6)에서 보는 바와 같이, 각각의 손실률이 단조 감소한다는 성질을 이용하여 불필요한 손실률 계산을 줄였다.

```
Algorithm SelectMethod
  input α = (사용자가 정의한 손실률 허용 임계값)
  /* 1. activate R2-Method as the initial method */
  activate R2-Method
  /* 2. select the optimal method w.r.t p and q values*/
  while the RTP session is active do
     wait for the RTCP APP packet
     get the p, q values from the RTCP APP packet
     if ( LossR0 ≤ α ) then activate R0-Method.
     else if ( LossR1 ≤ α ) then activate R1-Method.
     else if ( LossR2 ≤ α ) then activate R2-Method.
     else if ( LossR3 ≤ α ) then activate R3-Method.
     else if ( LossR4 ≤ α ) then activate R4-Method.
     else{activate R4-Method.
          printf("All Methods don't satisfy the α value.").
     }
     endif
  endwhile
```

(그림 3) 패킷 손실 특성에 따른 잉여 데이터 부가 방식 결정

· 수신자 동작 알고리즘

수신자 측에서는 실시간 데이터를 재생하기 위해 재생 버퍼 블록을 두어 비순차 패킷의 재배열과 지터 보상의 기능을 담당한다. 수신자 측은 송신자 측으로부터 RTP 패킷을 받을 때마다 크게 두 가지 동작을 한다. 첫째는, 원시 데이터를 재생 버퍼 블록에 저장하고, RTP 패킷 헤더에 들어 있는 확장 필드가 1이면, RTP 헤더 확장 부분의 잉여 데이터를 가져온 후, 이 잉여 데이터에 해당하는 재생 버퍼 블록을 살펴보고 이미 그 블록에 데이터가 있다면, 이 잉여 데이터는 불필요하므로 무시한다. 해당 재생 버퍼 블록이 아직 비어있으면 이 잉여 데이터를 저장한다. 두번째는, 재생 버퍼에 들어있는 패킷들의 배열 상태를 탐색하여, p, q에 대응되는 경우의 수들을 갱신하고, RTCP APP 패킷을 보내는 시점에 p, q 확률을 계산하도록 한다. 이 계산 주기는, RTP/RTCP에서 RR 패킷을 전송하는 시간 간

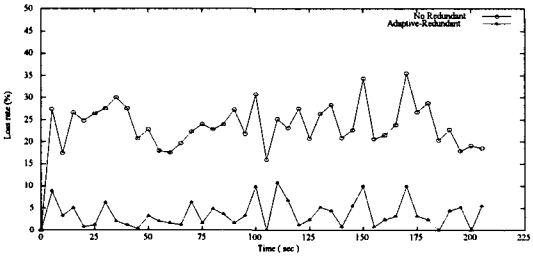
격으로 제안[1]되어 있는 평균 5초, 즉 8000Hz 음성 데이터인 경우에 약 500 여개의 패킷을 받을 수 있는 정도의 시간으로 설정한다. (그림 4)는 수신자 측이, 전달받은 RTP 패킷에 대한 처리 및 RTCP APP 패킷을 이용하여 송신자에게 p, q 값을 전송하는 과정을 간략하게 표현한 것이다.

```
Algorithm ReceiverProcessing
  while the RTP session is active do
    /* 1. receive data packets */
    원시 데이터 부분은 재생 버퍼로 이동
    if RTP header extension field == 1 then
       foreach redundant data in the RTPheader ext do
         if corresponding play buffer is empty then
            잉여 데이터를 원시 데이터 형태로 변환
            변환된 원시 데이터 형태를 재생 버퍼로 이동
         endif
       endfor
    endif
    /* 2. reply RTCP APP packets */
    if time to send RTCP APP packet then
      수신된 데이터들의 손실 특성으로부터 p, q 값 계산
      RTCP APP 패킷에 p, q 값을 실어서 송신자에게 전송
    endif
  endwhile
```

(그림 4) 수신자 동작 알고리즘

4. 모의 실험 및 분석

모의 실험을 위한 환경은 리눅스 운영체제가 동작하는 두 대의 시스템이, 서로 다른 서브넷을 가지는 환경에서 전송하도록 구성되었다. 송신자 측은 펜티엄 PC 기반의 리눅스 환경에 RTP 계층의 모듈들을 포함한 송신자 응용 프로그램이 UDP/IP 계층 위에서 동작한다. 수신자 측은 128M 바이트 메모리를 가지는 펜티엄 III 450MHz 급의 리눅스 플랫폼 노트북에 스피커 및 사운드 카드를 설치하였다. 여기에는 RTP 프로토콜 모듈과 Mobile IP를 포함한 수신자 응용 프로그램이 동작한다. RTP 프로토콜 모듈은 RFC1889 문서에 기반하여, 소스 코드가 공개되어 있는 rtptools[8], NeVoT [9]을 참조하여 구현되었고, Mobile IP는 HP사의 리눅스용 소스 코드[10]를 수정하여 사용하였는데, 이것은 사용자의 이동에 대한 투명성을 제공하기 위한 것이다. 실험은 간결성을 위해 유니캐스트 방식으로 행해졌다. 사용된 기본 코덱으로는 8000Hz 샘플링 비율의 64Kbps 전송률을 갖는 PCM 방식이다. 그리고, 잉여 데이터의 부가 전송으로 인한 대역폭 증가를 최소화하기 위해, 잉여 데이터의 형식은 4000Hz의 샘플링 비율을 갖도록 하였다. 또한 이동 컴퓨팅 환경에서의 패킷 손실 특성을 시뮬레이션하기 위하여, 패킷 전송 시에 송신자 측에서 길버트 모델을 적용한 랜덤 패킷 손실을 발생시켰다. 잉여 데이터는 RTP 패킷의 헤더 확장 부분에 삽입되어 전송되도록 하였고, 수신자 측에서는 패킷 손실에 대한 정보를 RTCP APP 패킷에 저장한 후, RTCP RR 패킷과 결합하여 전송하도록 하였다. (그림 5)는 제안된 복구 방법의 성능을 확인하기 위하여, 손실률 허용 임계값(α) = 5%, p = 0.12, q = 0.35인 경우에 대해, 비부가 방식의 경우와 동적 부가 전송 방법을 사용한 경우에 대한 손실률을 측정한 것이다. (그림 5)에서 보는 바와 같이, 동적 부가 전송 방법을 사용하였을 때의 손실률은 대략 5%를 중심으로 변화하였는데, 이것은 시간이 경과하더라도 임계값과 거의 유사하게 손실률을 유지한다는 것을 보여 준다.



(그림 5) 동적 부가 전송 방법의 적용 여부에 따른 손실률 비교 (α = 5%, p =0.12, q =0.35)

5. 결 론

본 논문에서는 이동 컴퓨팅 환경에서의 실시간 데이터 전송시 발생하는 간헐적인 패킷 손실 및 연속적인 패킷 손실을 복구하기 위하여 잉여 데이터의 동적 부가 전송 방법을 제안하였다. 제안된 방법은 현재의 패킷 손실 특성에 따라 잉여 데이터의 개수를 동적으로 부가하고, 특히 연속적인 패킷 손실이 많은 경우에는 비교적 큰 오프셋 값 즉, -4, -8의 오프셋 값을 가지는 잉여 데이터들을 선택하여 인터리빙 방식의 효과를 가지도록 함으로써, 전체 패킷 손실률뿐만 아니라 패킷 손실의 연속성도 감소시키고자 하였다. 또한, 이러한 동적인 적용은 패킷 손실이 적은 경우에는 잉여 데이터 부가 전송으로 인한 추가 대역폭 및 처리시간을 감소시킬 수 있다.

참고 문헌

[1] Internet Engineering Task Force, "RTP : A Transport Protocol for Real-Time Applications," RFC1889, Jan. 1996.

[2] 강민규, 궁상환, 김동규, "RTP/RTCP를 이용한 영상회의 시스템에서 오디오 패킷 손실 보상을 위한 동적 부가 전송 메카니즘 개발 및 성능 분석," 한국정보처리학회 논문지, 제5권, 제10호, Oct. 1998.

[3] 박준서, 고대식, "저비트율 잉여오디오 정보를 이용한 손실 패킷 복구 방법의 구현 및 성능평가," 대한전자공학회 논문지, 제35권, 제7호, 1998.

[4] K. Brown, S. Singh, "A Network Architecture for Mobile Computing," *IEEE INFOCOM'96*, pp. 1388-1396, Apr. 1996.

[5] J-C. Bolot, S. Fosse-Parisis, D. Towsley, "Adaptive FEC-Based error control for Internet Telephony," Proc. Infocom'99, New York, March 1999.

[6] C. Perkins, O. Hodson, "A survey of packet loss recovery techniques for streaming media," *IEEE Network*, Oct. 1998.

[7] V. Hardman, A. Sasse, M. Handley, "Reliable audio for use over the Internet," Proc. *INET'95*, Honolulu, pp. 171-178, June 1995.

[8] rtptools, http://www.cs.columbia.edu/~hgs/rtptools/

[9] NeVot, http://www.cs.columbia.edu/~hgs/nevot/

[10] MobileIP, http://www.hpl.hp.com/personal/Jean_Tourrilhes/MobileIP