

인터넷을 통한 원격제어기의 임의 시간지연의 예측가능성에 대한 연구

A Study on the Predictability of Random Time Delay of Telecontroller via Internet

심현승, 허경무, 김장기

(Hyun-Seung Shim, Kyung-Moo Huh, and Jang-Ki Kim)

Abstract : One of the important problems which should be solved in the telecontroller control is the time delay problem. In this paper, we propose a method of solving a random time delay problem using QoS(Quality of service), and we show not only how to solve an unpredictable time delay problem but also how to compute a maximum time delay that could satisfy a basic assumption of many telecontroller methods. Using our proposed method, it is find that we can offer more stable time delay in telecontroller than using TCP and UDP.

Keywords : QoS, time delay, predictability, telecontroller, internet, protocol

I. 서론

원격제어를 함에 있어서 가장 먼저 해결해야 할 문제점은 시간지연(time delay)의 문제점이다[1]-[3]. 기존의 원격제어 방법에 있어서는 이 시간지연문제를 해결하기 위하여 다양한 방법을 연구해 왔으나, 대부분의 방법들의 기본 가정은 시간지연이 예측 가능한 범위내에 있어야 한다는 것이다. 그러나 인터넷을 이용하여 원격제어를 구현할 때에는 이러한 방법들을 이용할 수가 없다. 왜냐하면 인터넷은 공유되어 있는 자원이기 때문이다. 따라서 본 논문에서는 인터넷을 이용한 원격제어시스템을 구축하기 위하여 기존의 인터넷에서의 예측 불가능한 시간지연을 예측가능한 시간지연으로 바꾸는 방법에 대하여 논의하고 테스트하고자 한다.

II. 인터넷을 이용한 원격제어 방법

불과 몇 년 전까지만 해도 컴퓨터간의 데이터 전송은 텍스트에 불과하였다. 그러나 지속적인 인터넷의 성능향상과 새로운 어플리케이션이 등장함에 따라 다양한 데이터를 전송하는 것이 가능하게 되었다. 그러나 인터넷을 통한 실시간 데이터 전송은 아직도 많은 문제점을 갖고 있다. 그 이유는 실시간 데이터 전송은 인터넷의 전형적인 전송 특성이 동적으로 불규칙적으로 변화하더라도 항상 안정하게 유지되어야 하기 때문이다. 엄격한 시간지연과 데이터 손실 특성을 가지고 있는 실시간 데이터는 전적으로 인터넷의 특성에 의존한다. 현재 제어시스템 디자인은 이러한 인터넷의 특성에 대하여 부적합하다. 왜냐하면 인터넷의 전송 특성을 고려하지 않았기 때문이다. 인터넷을 통하여 향상되고 있는 수많은 새로운 기술을 이용하여 새로운 시스템을 디자인하는 경우는 많았으나 인터넷의 특성을 고려하여 기존의 제어

시스템 디자인 이론을 수정하려는 시도는 부족하였다. 특히 인터넷을 이용한 원격제어시 가장 중요한 시간지연에 대하여 실험을 통한 최대값에 대한 고려만 하였을 뿐 그 값의 동적인 변화에 대해서는 신경을 쓰지 않았다.

1. 원격제어 시스템

대부분의 원격조종 시스템에서는 시간지연을 최대값으로 고정하여 연구하여 왔다. 대표적으로는 passivity theory [4], remote compliance control[5], wave variables decomposition[6] 등이 있다. 이러한 방법들은 모두 시간지연에 대하여 독립성을 어떻게 획득할 수 있는냐에 목적을 두고 연구되었다. 따라서 이러한 이론들은 불규칙적으로 변화하는 인터넷상의 시간지연 특성에서는 직접적으로 적용할 수 없다. 왜냐하면 인터넷상에서 정확한 시간지연 특성을 분석하는 것은 불가능하기 때문이다. 또한 시간지연을 $\tau=T$ 로 고정하여 설계된 제어시스템은 $0 \leq \tau < T$ 에서조차 불안정하게 될 수 있을 뿐만 아니라[7], 원격제어 시스템의 안전성이 시간지연 자체보다는 시간지연의 변화에 따라 달라질 수 있기 때문이다[7].

2. 인터넷상의 시간지연 문제에 대한 연구

공유된 인터넷에서의 시간지연은 임의 시간지연(random time delay)임을 알 수 있다. 이러한 인터넷의 시간지연은 다음과 같이 정의할 수 있다[8].

$$\text{시간지연} = \text{고정지연시간} + \text{큐잉지연시간} = (\text{광속전파지연시간} + \text{전송지연시간}) + \text{큐잉지연시간} \quad (1)$$

$$\text{광속전파지연시간(propagation delay)} = \text{거리}/\text{광속} \quad (2)$$

$$\text{전송지연시간(transmission delay)} = \text{패킷 크기}/\text{대역폭} \quad (3)$$

$$\text{큐잉지연시간} = \text{nondeterministic} \quad (4)$$

접수일자 : 2000. 12. 30., 수정완료 : 2001. 6. 11.

심현승, 허경무(주저자), 김장기 : 단국대 전자컴퓨터 학부(tema@shinbiro.com/huhkim@anseo.dankook.ac.kr/jkkim@anseo.dankook.ac.kr)

이 중 광속전파 지연시간과 전송 지연시간은 계산가능하고 데이터 전송중 고정된 항목임을 알 수 있다. 그러나 세 번째 시간지연 항목인 큐잉지연시간은 현재 네트워크 자원에 접속중인 사용자에 의해 결정되므로 계산이 불가능하다. 따라서 인터넷을 이용한 원격제어에서 시간지연 예측시 시간지연은 큐잉지연에 의하여 불규칙적으로 변화함을 알 수 있다.

1) 큐잉 지연 : 네트워크에서 여러 사용자가 공유하는 자원으로서 많은 패킷이 하나의 링크를 사용하기 위해서 경쟁하는 경우 큐는 그 즉시 포화상태가 될 수 있으며 일정수의 패킷은 전달이 취소되기도 한다. 이러한 상태를 혼잡상태라하며 큐잉 지연은 바로 이 상황에서 발생한다. 따라서 QoS는 기존의 최상의 노력(best effort) 서비스 모델을 확장하여 자원의 할당과 다양한 스케줄링 알고리즘을 이용하여 혼잡상태를 해결하며 보장된 자원의 할당하에서 파라미터를 통하여 시간지연이 계산 가능하도록 한다. 따라서 RSVP (Resource reSerVation Protocol)[9]의 트래픽 컨트롤 컴포넌트는 효율적인 큐잉 알고리즘과 스케줄링을 이용하여 시간지연을 계산가능하도록 한다. QoS는 그림 1과 같이 $\langle b, r, p, M, m \rangle$ 를 기본 파라미터로 갖는다. (b =토큰 크기, r =토큰율(token rate), p =피크 데이터율(peak data rate), M =최대 데이터그램 크기, m =최소 패킷 크기이다)[10][11]. 이를 이용하여 각 시스템(호스트 포함)은 자원을 할당하여 지정된 데이터 흐름의 예약을 주기적으로 수행하게 된다. 큐잉 지연을 계산화하기 위해 우선 살펴봐야 할 것은 패킷을 처리하기 위한 버퍼에서의 패킷의 처리속도에 관해서 이다. 그림 2는 지연시간 계산을 단순화하기 위하여 그림 1의 큐잉의 버퍼 모델중 토큰 버킷(token bucket) 모델만을 도시해놓은 그림이다. 토큰 버킷 모델이란 VBR(Variable Bit Rate)로 들어오는 패킷들을 자원의 낭비없이 수용가능토록 설계된 버퍼 모델이다. 즉 VBR 패킷에 따라 토큰율 r (byte/sec)과 토큰 깊이(token depth) b (bytes)를 설정함으로써 다양한 패킷을 수용할 수 있도록 한 모델로서, QoS의 기본 버퍼 모델이다. 그림 2에서와 같은 모델에서 패킷의 최대 처리시간은 식 (5)와 같다.

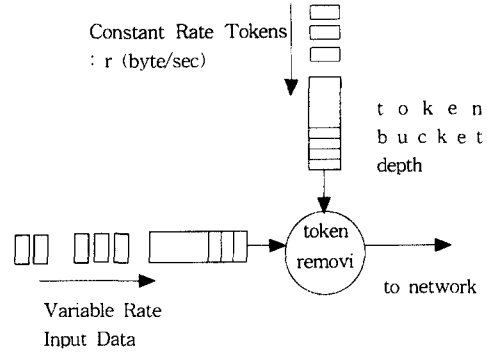


그림 2. 단순 토큰 버킷 모델.
Fig. 2. Simple token bucket model.

$$\text{최대처리시간} = b/r \text{ (sec)} \tag{5}$$

$$\text{최대처리량} = (b + r*t) \tag{6}$$

두 번째로 살펴봐야 할 사항은 버퍼에 저장되어 있는 패킷을 처리하는 큐잉 알고리즘이다. 이 알고리즘의 성능에 따라 전체 라우터의 성능의 차이가 나타나며 또한 혼잡상황에서도 각각의 패킷의 흐름이 독립적으로 이루어질 수 있다. 큐잉 알고리즘에는 FIFO, 우선순위 FIFO, FQ(Fair Queueing), WFQ(Weighted FQ) 등이 있다. 이 중 QoS는 WFQ를 기본으로 큐잉 알고리즘을 선택하여 계산하고 있다. 그림 3은 기본적인 WFQ의 다이어그램을 보이고 있다. 이 그림에서의 최대 시간지연은 식 (7)과 같이 계산되어질 수 있다. 여기에서 d_{max} 는 최대시간 지연(sec), b_i 는 버킷 깊이(바이트), C 는 링크 용량(바이트/sec), w_i 는 가중치, i 는 현재 보내는 클래스, Σw_i 는 전체 클래스의 가중치를 나타낸다.

$$d_{max} = \frac{b_i}{C * \frac{w_i}{\Sigma w_j}} \tag{7}$$

여기서 WFQ를 이용하여 QoS를 지원시 QoS의 모든 서비

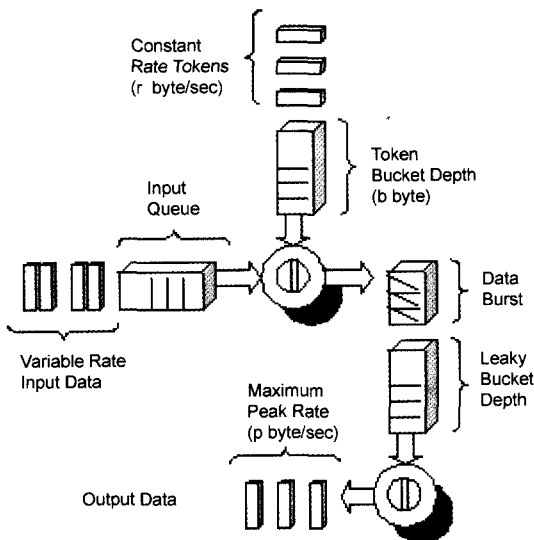


그림 1. 리키 버킷을 갖는 토큰버킷.
Fig. 1. Token bucket with leaky bucket.

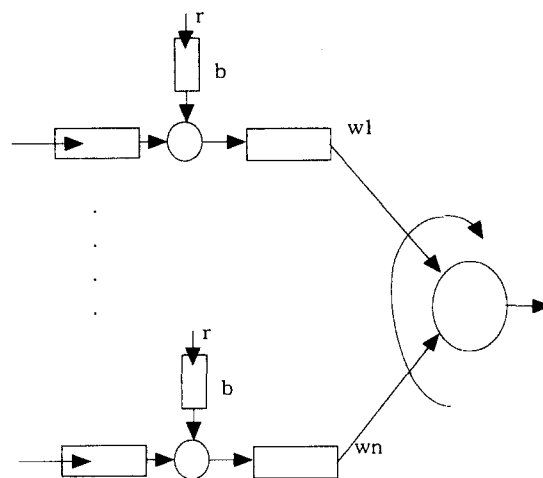


그림 3. WFQ 스케줄링에서의 N-multiplexed 토큰 버킷 플로우.
Fig. 3. N-multiplexed token bucket flows with WFQ scheduling.

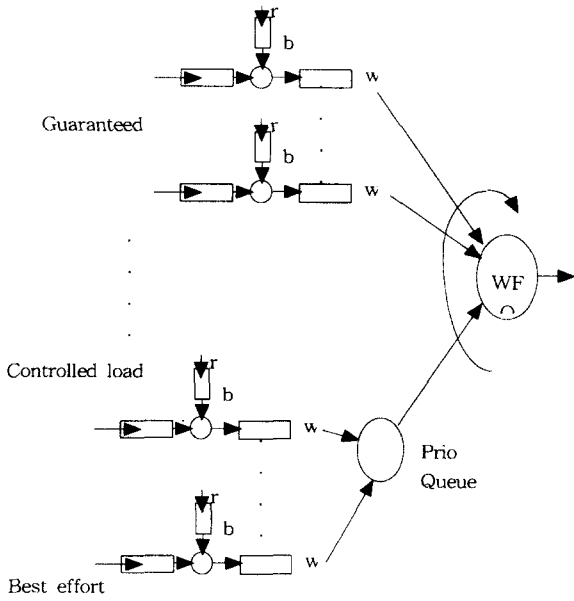


그림 4. QoS 모델을 적용한 WFQ 스케줄링.
Fig. 4. WFQ scheduling adapted for QoS model.

스 모델을 지원하는 간략화한 WFQ 모델은 그림 4와 같다. 각각의 서비스 모델별로 버퍼와 큐잉이 관리됨을 알 수 있다. 즉, 보장된 서비스(guaranteed service)는 WFQ에 의해 독립적으로 관리되어지며, 제어된 부하 서비스(controlled load service)는 최상의 노력 서비스보다 우선 순위가 높은 패킷으로 분리되어 다루어지고 있음을 알 수 있다. 따라서 QoS를 지원하는 각 라우터에서의 최대 지연시간은 식 (8)과 같다.

$$d_{max} = \frac{bi}{ri} + \frac{bi}{C * \frac{W_i}{\sum W_j}} \quad (8)$$

위의 식 (8)은 WFQ에서의 시간지연 항목을 계산한 것이다. QoS는 식 (8)을 좀 더 일반화하여 시간지연을 식 (9), (10)과 같이 계산한다[11].

$$d_{max} = \frac{b}{R} + \frac{Ctot}{R} + Dtot \quad (\text{if } p > R \geq r) \quad (9)$$

$$d_{max} = \frac{M}{R} + \frac{Ctot}{R} + Dtot \quad (\text{if } R \geq p \geq r) \quad (10)$$

여기서 b는 버킷 깊이(바이트), M는 최대 전달 단위(바이트), R은 수신자측에서 요구하는 데이터율 r이다. 그리고 Ctot와 Dtot는 각 네트워크 요소에서 발생하는 오차 성분을 말한다. 즉, Ctot는 비율에 지배되는(rate-dependent) 오차 성분들의 합계이며, Dtot는 비율에 독립적인(rate-independent) 각 엘리먼트 오차 성분들의 합계이다. Ctot는 식 (8)의 WFQ의 링크 용량을 예로 들 수 있다. 이것은 각 네트워크 요소의 전달 속도에 영향을 받으므로 비율에 지배되는 오차 성분이라 하며, 그러한 요소들의 총 합이 Ctot이다. Dtot 오차 성분은 전달률과 관계가 없는 요소들의 합으로서, 예를 들면, 가장 좋은 경우와 나쁜 경우의 시간차나 출력 인터페이스를

처리하는 네트워크 카드의 처리능력 및 수용가능한 최대 버퍼수 등을 들 수 있다. 오차 성분인 Ctot과 Dtot은 계산되어 전체 dmax의 이탈(deviation) 항목으로 처리된다.

그 다음으로 중요한 항목은 각각의 라우터에서의 시간지연을 어떻게 양쪽의 종단 호스트로 전달하는가 이다. 전달하는 방법은 앞에서 살펴본 바와 같이 RSVP 프로토콜을 이용하여 전달한다. 이를 위해 QoS의 각 파라미터 들을 표 1과 같이 정형화하였다[11]. 여기서 S는 슬랙(slack) 성분으로서 각 라우터에서 Ctot 과 Dtot를 계산하여 S에 저장하여 전송되어진다. RSVP는 표 2, 3과 같은 객체들을 각 서비스별로 정의하여 자원예약(대역폭과 최대 지연 시간)을 수행하게 된다.

표 1. QoS 사양과 파라미터.

Table 1. Specification and parameters of QoS.

QoS 사양	파라미터	전송방향
TSpec (trafficspecification)	<b,r,p,m,M>	sender → receiver
RSpec(receiver TSpec)	< R, S >	receiver → render

표 2. 제어된 부하 플로우 사양.

Table 2. Controlled load flow spec.

Length = 36		Class Num=9	C-Type=2
Version Num	Reserved	IS Length	
Controlled Load	Reserved	Service Data Length	
ParamID=127	Parameter Flags	Parameter Data Length	
Token Rate (r)			
Token Size (b)			
Peak Data Rate (p)			
Minimum Policed Unit (m)			
Maximum Packet Size (M)			

표 3. 보장된 서비스 플로우 사양.

Table 3. Guaranteed service flow specification.

Length = 48		Class Num=9	C-Type=2
Version Num	Reserved	IS Length	
Guaranteed	Reserved	Service Data Length	
ParamID=127	Parameter Flags	Parameter Data Length	
Token Rate (r)			
Token Size (b)			
Peak Data Rate (p)			
Minimum Policed Unit (m)			
Maximum Packet Size (M)			
ParamID=130	Parameter Flags	Parameter Data Length	
Rate (R)			
Slack Term (S)			

3. 원격제어 시스템의 데이터와 인터넷 특성의 관계

현재 인터넷을 이용한 원격제어 시스템의 데이터는 크게 두가지로 나뉜다. 하나는 원격제어 시스템을 제어하기 위한 제어 데이터와 사용자에게 원격지의 환경을 정확히 알려주기 위한 영상 데이터가 있다. 두가지 데이터의 특성을 살펴보면 표 4와 같다. 원격제어 시스템, 즉 각 호스트의 제어 시스템에서의 제어/영상 데이터에 상관없이 설계시 기준이 되는 데이터는 크게 샘플링 데이터와 전송 데이터로 구분할 수 있다. 샘플링 데이터는 샘플링 크기(Mh)와 샘플링률(Rh)이 있다. 전송 데이터로는 종단간 시간지연(end-to-end delay, Dh)과 우선순위 데이터(Ph)로 나눌 수 있다. 그리고 원격제어 시스템으로부터 취득된 데이터를 네트워크로 전송시 필요한 데이터는 크게 처리량과 트래픽 데이터로 나눌 수 있다. 처리량과 관련된 데이터로는 패킷 크기(Mn)와 패킷률(Rn), 그리고 트래픽 데이터로는 데이터 전송간격시간(interval time, Pn), 종단간 시간지연(end-to-end delay, Dn), 그리고 우선순위 데이터(Pn)가 있다. 이를 정리하면 그림 5와 같다. 이때 원격시스템 데이터와 네트워크 데이터의 관계를 살펴보면 다음과 같다. 네트워크를 통해 전송되는 데이터는 패킷크기 Mn으로 한정되어 있으므로 샘플링 데이터의 크기는 Mh/Mn의 정수배로 나누어지므로 패킷률 Rn은 식 (11)로 계산된다.

표 4. 원격 제어기의 데이터 특성.
Table 4. Data characteristics of teleoperation.

특성	데이터	제어 데이터	영상 데이터
신뢰성		high	low
종단간 시간지연		20ms	200ms
처리량		low	high

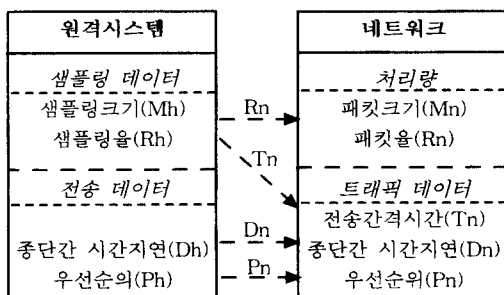


그림 5. 원격시스템의 데이터와 네트워크 데이터의 관계.
Fig. 5. Relation between teleoperation data and network data.

$$Rn = ([Mh/Mn]) * Rh \quad (11)$$

네트워크를 통한 데이터 전송간격은 패킷당 샘플링률에 따라 결정된다.

$$Tn = (1/Rh) / [Mh/Mn] \quad (12)$$

종단간 시간지연은 원격제어시스템에서 필요한 시간지연에서 각각 위고쓰는데 소요되는 시간을 감산하면 되므로

표 5. 네트워크 데이터와 QoS 데이터의 관계.
Table 5. Relation between network data and QoS data.

네트워크	QoS	
	Controlled load	Guaranteed
패킷크기 (Mn)	→	Token size (b)
패킷률 (Rn)	→	Token rate (r)
전송시간간격(Tn)	→	Peak Data rate (p)
종단간 시간지연(Dn)	→	dmax(b/r), dmax(b/r), rate (R), Slack term(S)
우선순위(Pn)	→	Controlled ID, Guaranteed ID,

$$Cn = Ch - RPT - \text{ST} \quad (13)$$

(RPT= Read Process Time, WPT= Write Process Time)
우선 순위는 같다.

$$Pn = Ph \quad (14)$$

호스트에서 네트워크로 전송된 데이터는 각각의 네트워크를 통하여 전송되며 이때 데이터 전송경로의 각 네트워크 기기(라우터, 스위치 등)의 QoS 파라미터와의 관계를 살펴보면 표 5와 같다. 따라서 호스트로부터 네트워크 그리고 각 네트워크 기기간의 전송파라미터 관계를 설정할 수 있다. 이를 통하여 원격제어기 설계시 원하는 파라미터 값들을 계산할 수 있다.

4. 전체 시스템 구성

본 연구에서는 인터넷을 이용한 원격제어시 문제점인 시간지연을 해결하기 위하여 QoS를 이용한다. QoS를 테스트하기 위해서는 직접 테스트망을 구축하고 필요한 컴포넌트를 설치해야 한다.

실험을 위해 구축된 테스트망은 그림 6과 같고, 테스트망의 각 시스템의 구성요소는 표 6과 같다.

○ 테스트망의 시나리오는 다음과 같다.

- master(192.168.1.3)는 slave(10.1.5.4)에게 일정한 제어신호를 보낸다. 이때, 192.168.1.6과 10.1.5.6은 서로 가능한 많은 패킷을 보내 네트워크에 과부하를 일으킨다.
- TCP, UDP 등을 테스트하고 QoS를 적용하기 전과 후의 성능을 테스트하여 차이점을 알아본다.

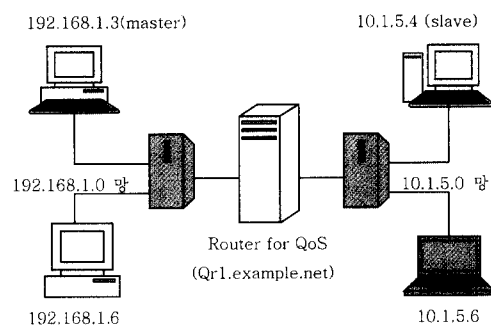


그림 6. QoS 테스트망.
Fig. 6. The network for testing QoS.

표 6. 시스템의 구성요소.

Table 6. Components on each system.

시스템	항목	내용
host	플랫폼	윈도우2000
	트래픽 제어	QoS 패킷 스케줄러
	RSVP 프로그램	GQoS API를 이용한 프로그램
Router	플랫폼	FreeBSD 3.4
	트래픽 제어	ALTQ 2.2
	RSVP 프로그램	rsvp4.2a4의 rsvpd
Hub	속도	10Mbps 스위칭 허브

III. 결과 및 고찰

본 연구에서 제시한 방법을 확인하기 위하여 프로그램과 실험환경을 구성하여 테스트하였다. 그림 7은 QoS와 TCP 같은 프로토콜들을 테스트하기 위해 작성한 프로그램이다. 프로그램 화면은 4부분으로 나누어져 있다. 좌측 상단은 테스트할 프로토콜을 선택하는 창이며, 우측 상단은 결과 데이터가 그려지는 창이며 우측 하단은 데이터의 수치를 표현하는 창이다. 그리고 네트워크의 성능을 나타낸 그래프는 윈도우2000의 퍼포먼스 데이터를 이용하여 작성한 프로그램으로 측정된 것이다. 참고로 그림 9는 10Mbps 이더넷에서 과부하를 일으키기 위한 프로그램의 대역폭을 측정된 것이다. 그림 8은 모든 실험을 수행하는 테스트망의 사진이다. 본 실험은 QoS의 세가지 서비스 모델인 최상의 노력, 제어된 부하, 보장된 서비스 모델을 테스트하였다. 본 실험 환경은 그림 6과 같은 10Mbps 이더넷 네트워크상에서 수행된다. 하지만, 간단한 실험의 구성으로 인하여 트래픽 생성 방법이 필요하다. 따라서 10Mbps의 용량을 2Mbps로 줄여 테스트하였다. 그림 9는 10Mbps 이더넷의 대역폭을 2Mbps로 줄였을 때의 그림 7과 같은 과부하 파형의 그림이다. 즉 가로축은 데이터 전송시간을 의미하며 세로축은 전송크기를 의미한다. 그림 9에서 보는 것과 같이 2Mbps인데도 3Mbps 정도의 대역폭이 나오는 것은 2Mbps를 실제 10Mbps 이더넷을 소프트웨어적으로 프로그램을 통해 줄였기 때문이다. 즉 물리적으로는 10Mbps이지만 소프트웨어적으로 2Mbps 환

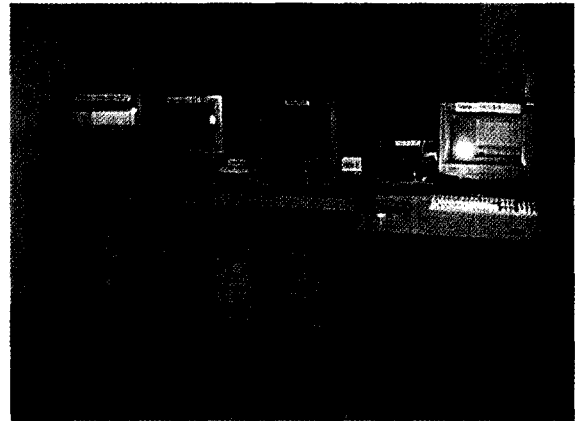


그림 8. 테스트 네트워크 사진.

Fig. 8. Picture of test network(좌측부터 192.168.1.4host(master), 192.168.1.6 host, Hub, Router, Hub, 10.1.5.6 host, 10.1.5.4 host(slave)).

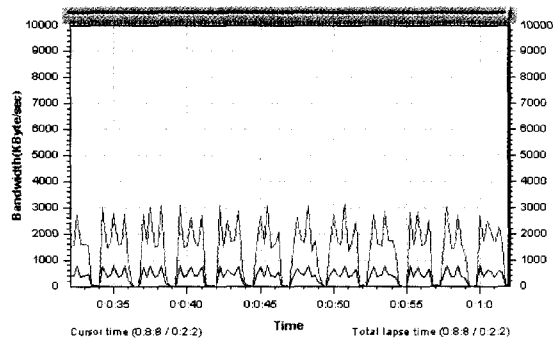


그림 9. 2Mbps 이더넷에서의 과부하 파형.

Fig. 9. Result in case of load and 2Mbps ethernet.

경으로 임의적으로 만든 것이므로 오차가 있음을 알 수 있다. 또한 그림 9의 잠음파형은 네트워크에 과부하를 일으키기에 충분함을 알 수 있다.

가) 대역폭 확보 실험 결과 : 대역폭 실험은 다음과 같은 시나리오로 행하여졌다.

- 무부하시와 과부하시에서 각 프로토콜을 테스트한다.
- 무부하시와 과부하시에서 일정한 제어신호를 보내서 대역폭이 확보가 되는지 실험한다.
- 수신측과 송신측의 대역폭을 모두 비교한다.

1) TCP : 그림 10과 같이 TCP의 경우 부하가 있을 때에는 부하가 없을 때에 비해 데이터 전송이 불규칙적으로 이루어지고 전송시간도 길어졌다. 따라서 TCP를 이용하여 원격조종을 할 시에 네트워크의 환경과 시간에 따라 원격조종이 안될 수도 있다는 것을 의미한다.

표 7. 테스트 서비스 모델.

Table 7. Service model of each test.

QoS 서비스 모델	프로토콜	트래픽 사양
최상의 노력	TCP	
제어된 부하	RSVP TCP	H263QCIF
보장된 서비스	RSVP TCP	G711

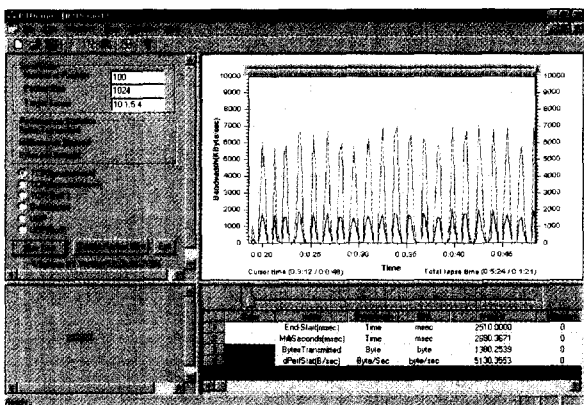
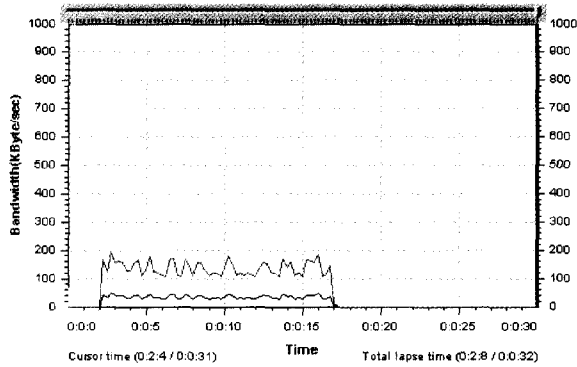
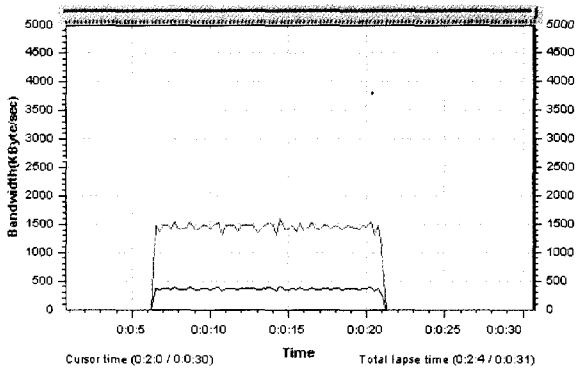


그림 7. QoS 및 프로토콜 테스트 프로그램.

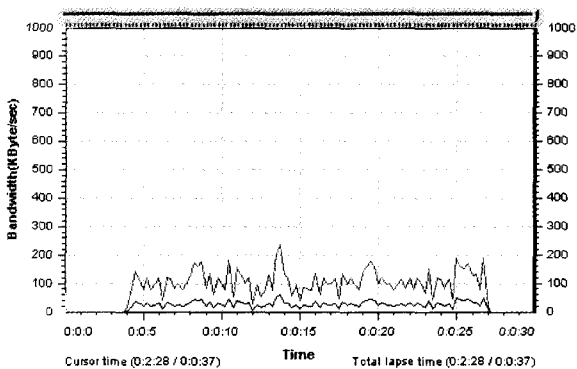
Fig. 7. Program for QoS and protocols' test.



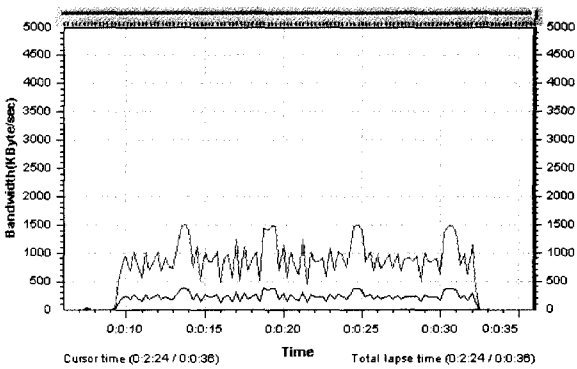
(a) 무부하시 송신측 전송률
(a) Bandwidth of sender in no load



(b) 무부하시 수신측 전송률
(b) Bandwidth of receiver in no load



(c) 부하시 송신측 전송률
(c) Bandwidth of sender in load



(d) 부하시 수신측 전송률
(d) Bandwidth of receiver in load

그림 10. TCP의 전송률.
Fig. 10. Bandwidth of TCP.

2) H263QCIF : 그림 11과 같이 H263QCIF를 이용한 제어 된 부하는 과부하시에도 일정한 대역폭과 전송시간을 유지 함을 알 수 있다. 여기에서 H263QCIF는 실시간 전송 프로토 콜인 H263의 비디오 프로토콜이다.

3) G711 : 그림 12와 같이 G711을 이용한 보장된 서비스 실험은 부하가 없을 때와 비교하여 거의 변화가 없음을 알 수 있다. 여기에서 G711는 실시간 오디오 전송 프로토콜이 다. 따라서 기존의 인터넷에서 QoS를 이용할 시 원활한 원 격조종을 할 수 있음을 알 수 있다.

표 8. H263QCIF의 플로우 사양.
Table 8. Flow spec. of H263QCIF.

항 목	H263QCIF
서비스 형태	Controlled Load
토큰율 r	12000byte/sec
피크 대역폭 p	-1
토큰 버킷 크기 b	6000
MaxSduSize M	2500
MinimumPlicedSize m	80

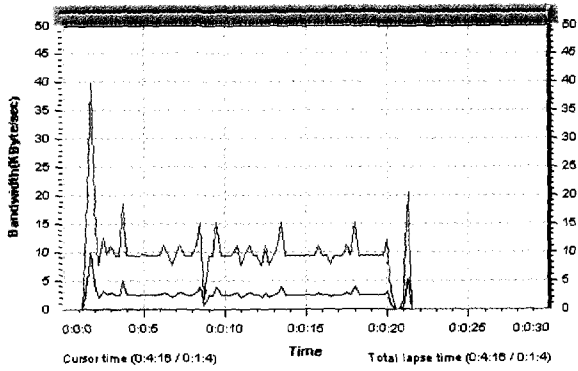
표 9. G711의 플로우 사양.
Table 9. Flow spec. of G711.

항목	H263QCIF
서비스 형태	보장된 서비스
토큰율 r	9250byte/sec
피크 대역폭 p	13875
토큰 버킷 크기 b	680
MaxSduSize M	340
MinimumPlicedSize m	340

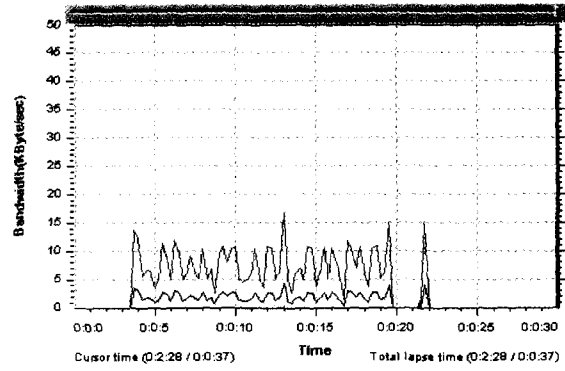
나) 최대 지연 시간 실험 결과 : 본 연구의 목적은 인 터넷을 통한 원격제어시 발생할 수 있는 시간지연의 문제를 해결할 수 있는 방법을 제공하는 것에 있다. 일반적인 원격 제어 시스템의 설계는 시간지연의 범위에 따라 크게 달라진 다. 따라서 본 실험에서는 일반적으로 인터넷을 이용시 동 적으로 불규칙하게 변화하는 시간지연의 범위를 이론적으 로 예측가능하고 계산할 수 있음을 보이고 실험을 통하여 이론상의 수치가 실제 실험에서도 그대로 입증될 수 있음을 보이고자 한다. 또한 본 논문에서 제시하는 QoS를 이용하여 원격제어시 정해진 범위내에서 시간지연의 순간 변화폭의 크기를 조사하여 각 서비스 모델에 따른 시간지연의 통계적 인 수치를 제시하고자 한다. 이를 위해 본 실험에서는 그림 13의 콘솔 프로그램을 작성하여 실험하였다. 좌측의 그림들 은 송신프로그램이며 우측은 수신프로그램의 순서에 따른 그림이다. 프로그램의 순서는 다음과 같다.

시작(a,b)→자원예약 체결(c,d)→데이터 송/수신(e,f)
→종료

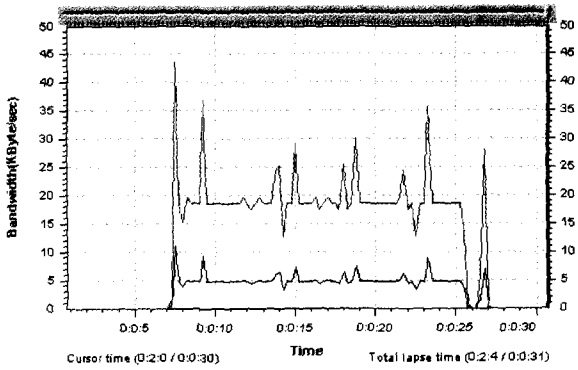
참고로 프로그램을 콘솔로 작성한 것은 GUI로 작성시



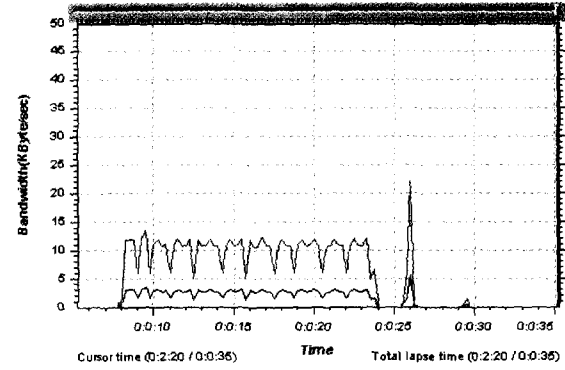
(a) 무부하시 송신측 전송률
(a) Bandwidth of sender in no load



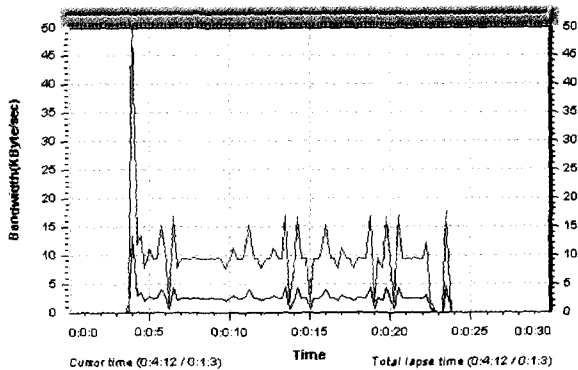
(a) 무부하시 송신측 전송률
(a) Bandwidth of sender in no load



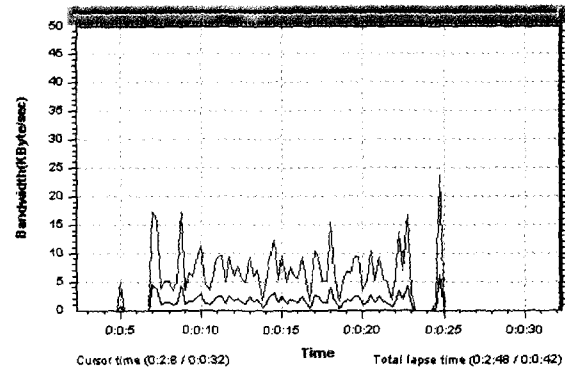
(b) 무부하시 수신측 전송률
(b) Bandwidth of receiver in no load



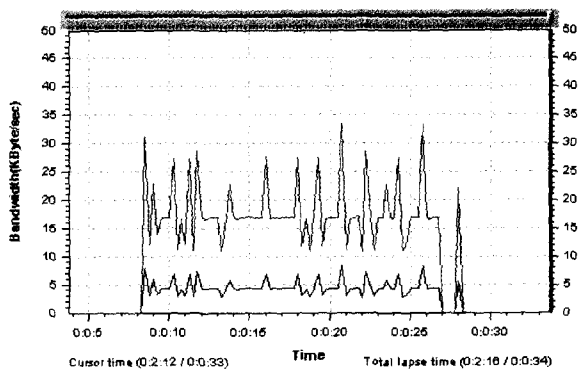
(b) 무부하시 수신측 전송률
(b) Bandwidth of receiver in no load



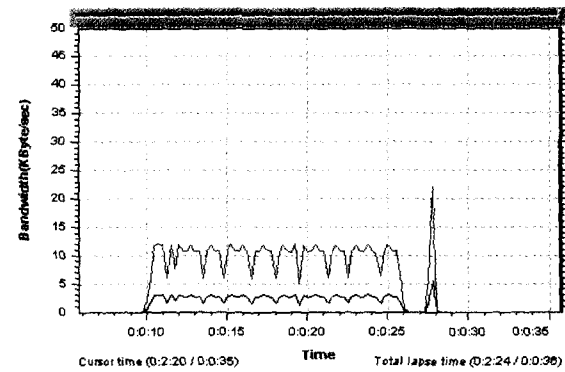
(c) 부하시 송신측 전송률
(c) Bandwidth of sender in load



(c) 부하시 송신측 전송률
(c) Bandwidth of sender in load



(d) 부하시 수신측 전송률
(d) Bandwidth of receiver in load



(d) 부하시 수신측 전송률
(d) Bandwidth of receiver in load

그림 11. H263QCIF의 전송률.
Fig. 11. Bandwidth of H263QCIF.

그림 12. G711의 전송률.
Fig. 12. Bandwidth of G711.

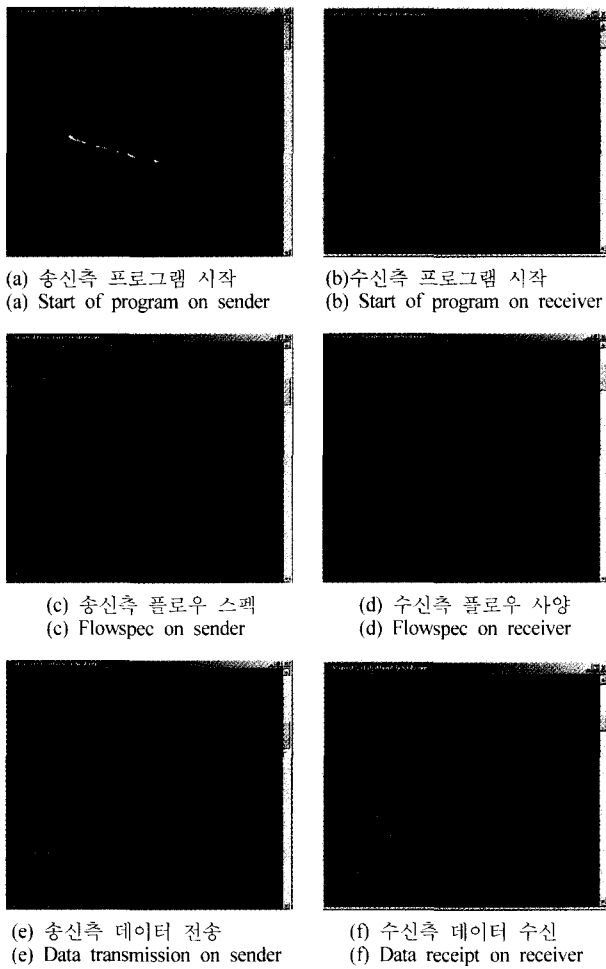


그림 13. 최대지연 시간 측정 프로그램.
Fig. 13. Maximum time-delay measurable program.

올 수 있는 OS상의 시간지연을 없애기 위함이며 본 실험은 인위적인 병목현상을 위하여 물리적으로는 10Mbps 이더넷이지만 소프트웨어적으로 제어된 2Mbps 이더넷상에서 수행하였다. 실험방법은 아래와 같은 총 1040바이트의 데이터를 송신측에서 수신측으로 전송한다. 계산을 단순히 하기 위하여 제어 데이터와 영상 데이터를 같이 두었다. 여기서 제어 데이터 항목은 제어신호를 전송하기 위한 데이터로서 그림 14와 같은 총 200개의 sine 값을 전송한다. Time 항목은 보내는 시간을 나타내며, seq_no는 보낸 데이터의 순서를, 그리고 image[1024]는 영상 데이터이지만 본 실험에서는 보내는 데이터의 크기를 조정하기 위한 임의의 데이터로서 사용하였다. 그리고 송신측에서 데이터를 일정하게 보내기 위하여 200ms 단위로 데이터를 수신측으로 전송한다. 따라서 송신측에서 수신측으로 보내는 데이터는 다음과 같은 항목으로 보내어진다.

- 샘플링 데이터 크기 : 1040 byte
- 샘플링률 : 5sample/sec (200ms간격)
- 샘플링 데이터의 총 수 : 200개
- 샘플링 데이터 값의 성격 : sine value(-1~1)

그림 13의 (f)를 보면 200개의 데이터가 수신측에 수신됨에 따라 송신시간과 수신시간 그리고 1040 바이트가 수신됨

을 알 수 있다. 참고로 그림 18은 서비스 모델 중 G711 테스트시의 그림들이다.

```
typedef struct _TransferData
{
    double      controldata;
    DWORD      time;
    USHORT     seq_no;
    BYTE       image[1024];
} TransferData;
```

본 실험에서 최대 지연시간의 측정 방법은 다음과 같다.

- 송신측에서 전송시간을 QoSData.time 항목에 저장하여 송신한다.
- 수신측에서 수신한 시간을 측정하여 시간지연을 계산한다. 시간지연의 식은 식 (15)와 같다.

$$\text{지연 시간} = \text{수신측 시간간격} - \text{송신측 시간간격} \quad (15)$$

그리고 본 실험에서는 최대지연시간 계산을 단순화하기 위하여 식 (9)과 (10)에서 $C_{tot}=D_{tot}=0$ 으로 하여 계산한다. 따라서 본 실험에서 이론적인 최대지연시간은 식 (16), (17)과 같다. 그리고 각각의 서비스 모델에 대하여 모든 실험은 동일한 조건하에서 수행된다.

$$d_{max} = \frac{b}{R} \quad (\text{if } p > R \geq r) \quad (16)$$

$$d_{max} = \frac{M}{R} \quad (\text{if } R \geq p \geq r) \quad (17)$$

1) TCP : TCP의 경우, 즉 최상의 노력의 경우 이론적으로 계산된 최대 지연과 실험을 통해 측정된 최대지연 시간은 다음과 같다.

- 계산된 최대 지연 시간 : 계산할 수 없음 (왜냐하면 TCP는 최상의 노력이기 때문이다.)
- 측정된 최대 지연 시간 : 19/190ms (부하가 없을 때/있을 때)

실험에서 보인 것과 같이 TCP의 경우 부하가 있을 때와 없을 때의 차이가 크게 나타남을 알 수 있다. 따라서 현재 인터넷의 이용자에 따라 동적으로 불규칙하게 시간지연이 변화함을 알 수 있다.

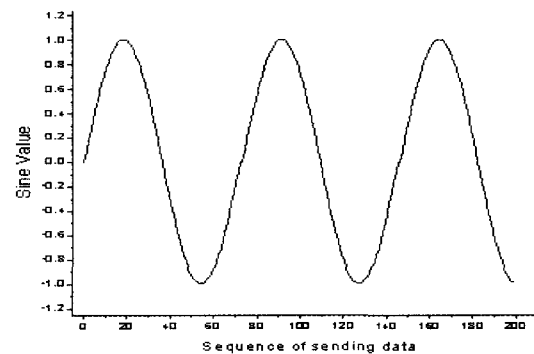


그림 14. sine 파형.
Fig. 14. Sine waveform.

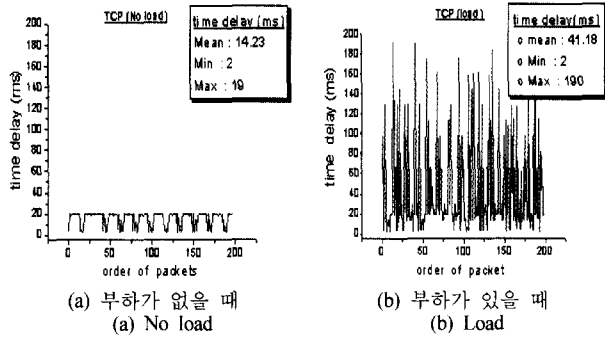


그림 15. TCP의 지연시간.
Fig. 15. Maximum time delay in TCP.

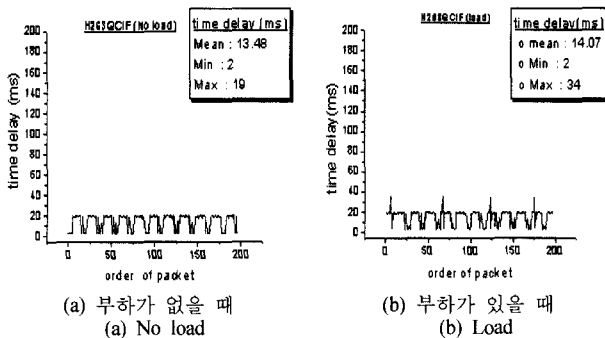


그림 16. H263QCIF의 지연시간.
Fig. 16. Maximum time delay in H263QCIF.

2) H263QCIF : H263QCIF의 경우 대역폭만을 확보하는 제어된 부하 서비스이므로 수신측에서 R을 설정하지 않는다. 따라서 이 경우 R은 r과 같다.

- 계산된 최대 지연 시간 : $d_{max} = 6000/12000 = 0.5(\text{sec})$
- 측정된 최대 지연 시간 : 19/34 ms (부하가 없을 때/있을 때)

H263QCIF, 즉 제어된 부하 서비스 실험에서 이론적인 최대 지연시간은 500ms 이내가 되어야 한다. 실제 실험에서는 부하가 없을 때는 TCP의 경우와 최대지연시간이 19ms로 같았으나, 부하가 있을 때는 이와는 다르게 34ms 이내로 데이터의 전송이 이루어짐을 알 수 있다. 계산된 최대지연시간인 500ms 이내임을 실험을 통해 알 수 있다.

3) G711 : G711의 경우 라우터(Qr1.example.net)에 접근하여 예약되어 있는 RESV 상태 정보를 확인하면 그림 17와 같다. 여기서 G는 보장된 서비스를 의미하며 T는 트래픽 플로우 사양이다. 그리고 S=0, R=10.3K임을 알 수 있다. 따라서 본 실험에서는 오차성분인 Ctot 과 Dtot을 제로로 둔 가정이 성립함을 알 수 있다. 이는 본 실험환경이 단 하나의 라우터를 거치고 부하를 일으키는 호스트가 적기 때문이라고 볼 수 있다.

G711의 경우 최대지연시간은 다음과 같다.

- 계산된 최대 지연 시간 : $d_{max} = 680/10300 = 0.066(\text{sec})$
- 측정된 최대 지연 시간 : 66/66 ms (부하가 없을 때/있을 때)

```

Sender: 192.168.1.3/1096  PHOP: <192.168.1.3 LTH=0>  TTD: 190729
In_if 0=>x10  Outlist <1,2>  flags *PE  ip_ttl 128
T=[10.3K(760) 15.5KB/s 380 380]  Adspec: ( 0 hop 250KBW 4.29Gus 1500B, G=(0 3800)

----- End of Dest state dump -----

11:16:15.245]  Snd Raw  PATH  10.1.5.4/5001[6]  1=>x11 > 10.1.5.4/127
PATH: Sess: 10.1.5.4/5001[6]  R: 30000  PHOP: <10.1.5.2 LTH=1>
192.168.1.3/1096  T=[10.3K(760) 15.5KB/s 380 380]
Adspec: ( 1 hop 250KBW 0us 1500B, G=(0 3800 0 3800))

11:16:15.265]  Rcv Raw  RESV  10.1.5.4/5001[6]  x11<=1 < 10.1.5.4/128
RESV: Sess: 10.1.5.4/5001[6]  R: 30000  NHOP: <10.1.5.4 LTH=1>
FF 192.168.1.3/1096  [G R=10.3K S=0 T=[10.3K(760) 15.5KB/s 380 380]]

11:16:15.280]  AddFlow  x11  10.1.5.4/5001[6]  Flg= =>handle=1352419
flowspec=[G R=10.3K S=0 T=[10.3K(760) 15.5KB/s 380 380]]  Tspec=T=[10.3K(760)
11:16:15.285]  AddFlit  x11  10.1.5.4/5001[6]  Flg= =>handle=1352419
Filter= 192.168.1.3/1096  Fhandle=135267296
    
```

그림 17. 라우터의 RESV 정보.
Fig. 17. RESV information of router.

G711, 즉 보장된 서비스의 경우 예약이 체결된 후 66ms의 최대 지연시간안에 모든 데이터가 전송되어야 함을 알 수 있다. 측정된 최대 지연시간을 보면 부하가 없을 때와 있을 때와 상관없이 66ms임을 알 수 있다. 다른 서비스 모델과 비교하여 부하의 영향이 없음을 알 수 있다. 그리고 이론적으로 예측된 지연시간과 똑같음을 알 수 있다. 따라서 G711의 경우 QoS에 의해 지정된 대로 엄격하게 최대지연시간을 유지함을 알 수 있다. 따라서 인터넷을 이용하여 원격제어 시스템을 설계시 현재 인터넷상의 사용자에 상관없이 엄격하게 정의된 그리고 예측가능하고 계산가능한 최대시간지연 값을 이용할 수 있음을 알 수 있다. 그림 18은 G711의 경우 시간지연을 측정된 그래프이다. 표 10은 부하가 있을 때와 없을 때의 각 템플릿의 최대 지연시간의 통계이다. TCP의 경우 부하가 있을 때와 없을 때의 평균과 최대지연시간에서 큰 차이를 보이지만, H263QCIF와 G711의 경우 TCP와 비교하여 안정된 지연시간을 보임을 알 수 있다. 그리고 G711의 경우 최대 지연시간이 부하가 없을 때와 있을 때 모두 66ms임을 알 수 있으며 평균에서도 가장 안정된 결과를 나타낼 수 있다. 표 11은 지연시간의 순간 변화를 나타낸 표이다. 즉 현재 지연시간에서 전단계의 지연시간 변화의 절대값의 통계치를 나타낸 표이다. TCP의 경우가 가장 급격한 변화를 나타내고 있음을 확인할 수 있다. 표에서 보이는 바와 같이 부하가 없는 경우에는 모든 서비스 모델에서 10ms 이내의 평균 순간변화폭을 보였다. 그러나 부하가 있는 경우에는 TCP의 경우 52ms의 큰 변화폭을 보였으나, QoS가 적용된 경우에는 최대 15ms 이내의 평균 순간변화폭을 보임

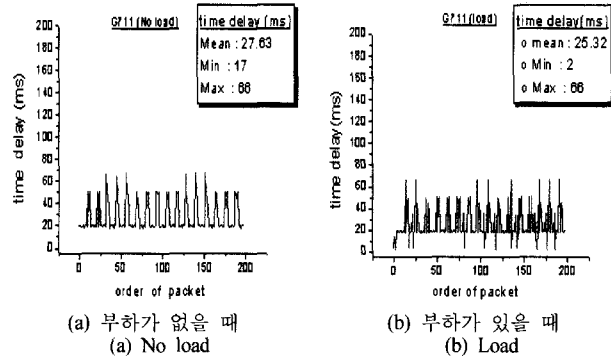


그림 18. G711의 지연시간.
Fig. 18. Maximum time delay in G711.

표 10. TCP/H263QCIF/G711의 평균/최소/최대 지연시간.
Table. 10. Mean, min, and max. time delay in case of TCP, H263QCIF and G711.

	TCP	H263QCIF	G711
평균	14.23/41.18	13.48/14.07	27.63/25.31
최소	2 / 2	2 / 2	17 / 2
최대	19 / 190	19 / 34	66 / 66

(단위:msec, 부하가 없을때/있을때)

표 11. TCP/H263QCIF/G711의 평균/최소/최대 지연시간 편차.
Table. 11. Mean, min, and max. time delay deviation in case of TCP/H263QCIF/G711.

	TCP	H263QCIF	G711
평균	3.74/52.09	4.06/4.81	7.69/14.19
최소	0 / 0	0 / 0	0 / 0
최대	17 / 187	17 / 31	48 / 63

(단위:msec, 부하가 없을때/있을때)

을 알 수 있다. 즉 QoS가 적용된 경우와 그렇지 않은 경우의 평균 순간변화폭의 차이는 37ms의 차이를 보임을 알 수 있다.

IV. 결론

본 연구에서는 인터넷을 이용한 원격제어시 문제점으로 지적되는 동적으로 불규칙하게 변화하는 예측 불가능한 시간 지연을 해결할 수 있는 방법을 제시하고, 이를 적용한 실험결과를 보였다. 본 논문의 결론은 다음과 같이 세가지로 압축된다.

1) 최대 지연 시간을 미리 예측하여 설정할 수 있으므로 이에 따라 원격제어기를 효율적으로 설계할 수 있다.

2) 특정한 시간 범위에 따라 설계된 원격제어기의 안정성에 영향을 미치는 지연 시간의 순간변화폭이 기존의 인터넷에 비해 QoS를 이용할 시 훨씬 작다. 따라서 좀더 안정적인 원격제어기의 성능을 기대할 수 있다.

3) 다양한 템플릿을 통하여 응용프로그램의 특성과 적용 환경에 따라 유연하게 대처할 수 있음을 알 수 있다.

참고문헌

[1] Ken Goldberg, Michael Mascha, Steve Gentner, Nick Rothenberg, Carl Sutter, and Jeff Wiegley, "Desktop teleoperation via the world wide web," *IEEE Int. Conf. on Robotics and Automation*, 1995.

[2] Kevin Brady "Internet-based remote teleoperation," *Proc. of 1998 IEEE Int. Conf. on Robotics and Automation*, May, 1998.

[3] Kazumasa hirai and Yoshiaki satoh, "Stability of a system with variable time delay," *IEEE Tr. on Automatic Control*, vol. ac-25, no. 3, June, 1980.

[4] Anderson, R., "A modular control architecture for tele-robotics," *IEEE Robotics and Automation Magazine*, 2(3) 10-18, 1995.

[5] Kim, W., Hannaford, B., and Bejczy, A., "Force reflection and shared compliant control in operating telemanipulators with time delay," *IEEE Tr. on Robotics and Automation*, 8(2), 175-185, 1992.

[6] Niemeyer, G. and Slotine, J., "Stable adaptive teleoperation," *IEEE Journal of Oceanic Engineering*, 16(1), pp. 152-162, 1991.

[7] Eusebi, A. and Melchiorri, C., "Stability analysis of bilateral teleoperation robotic systems," *3rd European Control Conference(ECC '95)*, 1995.

[8] Larry L. Peterson, Bruce S. Davie, "Computer networks : A system approach 2nd," Morgan kaufmann 2000.

[9] Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S., "Resource ReSerVation Protocol (RSVP) - Version 1 functional specification," *RFC 2205, September 1997*, Proposed Standard.

[10] Shenker S., Prtridge. C., "Specification of controlled load quality of service," *IETF Internet Draft*, July 1995.

[11] Microsoft Inc., "MSDN library," July 1999.



심 현 승
1988년 단국대학교 전자공학과 졸업, 동대학 석사과정.



허 경 무
1979년 서울대학교 전자공학과 졸업 한국과학기술원 전기 및 전자공학과 석사(1981). 동대학 박사(1989). 현재 단국대학교 전자컴퓨터학부 전자공학전공 부교수.



김 장 기
1972년 단국대학교 전자공학과 졸업, 동대학 석사(1976). 현재 단국대학교 전자컴퓨터학부 교수.