Measurement and Emulation of Time Varying Packet Delay with Applications to Networked Haptic Virtual Environments

Ganesh Sankaranarayanan, Lauren Potter and Blake Hannaford
BioRobotics Laboratory
Department of Electrical Engineering
University of Washington
Seattle, Washington 98195
Email: {ganeshs,denz410,blake}@u.washington.edu

***Abstract*— Networked haptic virtual environments (NHVEs) are increasingly being used in medical simulation, aircraft maintenance training, and other similar fields. In this paper we present the implementation of a network emulator that can create realistic Internet-like characteristics in a laboratory setting for networked haptics. We compare the quality of this delay emulator to actual measurements taken on the Internet by reflecting UDP data packets and analyzing their round-trip delay distribution and packet loss.**

I. Introduction

Networked haptic virtual environments (NHVEs) connect multiple users who can be located remotely through dedicated networks or the Internet. They can use either client-server or peer-to-peer architectures for connecting multiple users. In networks like the Internet, the delays are time varying and the packet loss depends on the network traffic. These networks use a packet-switching mechanism to direct the packets to their destinations. As a result, the delay encountered by a packet consists of a fixed propagation delay between the routers, and a varying component that includes processing and queuing delays at each of these routers. Packets may also be dropped or rejected at these routers because of buffer overflows or packet errors. The drop rate and delay characteristics differ considerably depending on the rate and size of the data packets.

In NHVEs the information often needs to be exchanged at a rate of 1000 packets/sec because of the high sampling rate that many of the haptic devices use [1]. Depending on the packet size, this can occupy a significant portion of the available bandwidth. Unlike the connection-oriented TCP, where there is a back and forth transmission of packets, UDP is faster and much better suited for high packet transmission rate NHVE applications, since it is a connection-less lightweight protocol.

It is often difficult to test the robustness of a NHVE system unless it is exposed to a realistic network condition. At the same time, it is desirable to achieve this in a laboratory setting so that the systems can be fine-tuned for performance. These test systems could also be useful for other delay-sensitive systems such as teleoperation.

Network characteristics are important to a variety of interactive applications including cooperating robots, teleoperated robots, and multi-user virtual reality. Future applications such as coupled multi-user, multi-robot teleoperation systems would be enabled by robust network performance. For example, a junior surgeon and a mentor surgeon could jointly control a surgical robot with both surgeons and the patient in three separate locations.

Our current research focus is NHVEs. However in this paper we study several different servo rates so that the results map to multiple applications.

Network emulators are a type of software that can locally recreate various network conditions such as delay, packet loss, bandwidth limitations, and router congestions in a laboratory setting. NIST Net [2] is a popular one.

A. Goals of this Study

The objective of this paper is to evaluate NIST Net for its suitability as a network emulator for NHVE systems.

II. Background

Previous work in NHVEs can be classified based on the type of control used, as either *centralized* (client-server) [3, 4, 5, 6] or *distributed* (peer-to-peer) architecture [7, 8, 9]. In [10, 11], both centralized and distributed architectures were used to study NHVEs. Based on the type of force rendering, NHVEs can be further classified into: *simultaneous* [4, 5, 6, 9, 10, 11] where users simultaneously manipulate a virtual object, and *one-at-a-time* [7, 8, 3], where each user takes turns in manipulating the virtual object.

In [5, 6], delay interference was produced using two workstations with one of them sending fixed data size packets to the other. The two workstations share the same Ethernet hub with the client and server used for the experiments. In [4] time varying delay was assumed to be a Gaussian distributed with an expectation value of δ. In [9] the actual Internet was used for the experiment. The bandwidth usage was reduced by sending packets only when at least one of the participants was in contact with the virtual object in the simulation.

ROBOCOMM 2007, 14th – 16th Oct 2007, Athens, Greece.
Copyright © 2011 – 2012 ICST
ISBN 978-963-9799-08-0
DOI 10.4108/ICST.ROBOCOMM2007. 2210

Several popular network emulator software packages exist, such as Dummynet for FreeBSD, and the Hitbox pseudo-device for SunOS, which can recreate different network conditions inside of a local area network. NIST Net [2] is one of such Linux-kernel based open source network emulators. It has features for inputting a user-defined delay table, while some of its configurable parameters are: mean delay, standard deviation, bandwidth, packet drop percentage, and packet duplication. In addition, it has both graphical and command line interfaces for user input.

In [12] the end-to-end packet delay and loss behavior of the Internet was studied using a UDP probe packet at different data rates. They found rapid fluctuations of queuing delays over small intervals, and compression of the probe packets and packet loss were random, unless the probe packets used a large amount of the available bandwidth. In [13] the Internet delay and loss behavior was studied using TCP packets to model out-of-order packets and reordering. They found that a large percentage of packet reordering was common throughout the Internet and it was correlated to routing fluctuations.

III. Methods

A. Generation of Random Delay Variables

The NIST Net uses a very simple method to generate delay random variables with specified mean and standard deviation based on the input delay distribution table.

Let X be the normalized random variable having a probability density function $f(x)$. Then the cumulative distribution function *cdf* of the continuous random variable X is given by equation 1.

$$F_X(x) = P[X < x] = \int_{-\infty}^{x} f(t)dt \tag{1}$$

Assume that inverse of the *cdf* exists and is given in equation 2.

$$Z_Y(y) = F_X^{-1}(x) \tag{2}$$

Let U be a uniformly distributed random variable in the interval [0,1]. Then, a desired random variable Y can be obtained as shown in equation 3. The proof of this is straightforward and it is shown in equation 4.

$$Y = Z(U) \tag{3}$$

$$\begin{aligned} P[Y \leq x] &= P[Z(U) \leq x] \\ &= P[U \leq F_X(x)] = F_X(x) \end{aligned} \tag{4}$$

In NIST Net, $F_X(x)$ is the *cdf* of the normalized delay distribution which is obtained through measurements of a network like Internet. Let the new distribution that is generated using $F_X(x)$ have a desired mean delay $D\ ms$ and standard deviation $\sigma\ ms$. Then, the NIST Net generated delay time is given in equation 5.

$$Y = D + Z(U) * \sigma \tag{5}$$

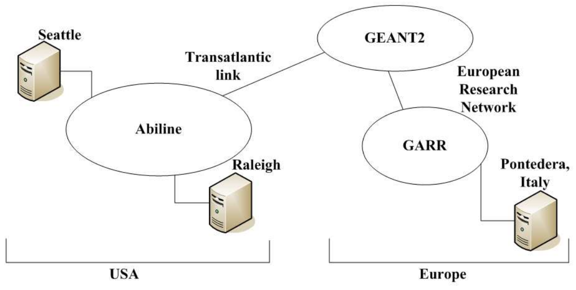

Fig. 1. Network topology for the delay characterization experiment

For a more detailed implementation of correlation, random packet drop, and other parameters refer to [2].

IV. Experiment setup

A. Delay Characterization Experiment Setup

To study the characteristics of the Internet for haptic data rates, a packet reflector program was hosted at a server in North Carolina State University, Raleigh, North Carolina, and Scuola Superiore Sant'Anna, Pontedera, Italy. The setup consists of two computers, one at our laboratory at the University of Washington, Seattle and the other at Raleigh or Pontedera, Italy, respectively. UDP data packets with similar data structure as the one used in our NHVE study were used for transmission. The structure of the data packets is given below; it consists of two time stamp fields for the server and client computers, an integer field for the packet sequence number, data arrays for transmitting position information, and a checksum field for doing a simple checksum-based error.

```
typedef struct{
unsigned long int servoTick;
RtUTCtime s_timestamp;
RtUTCtime c_timestamp;
double g_pos[3];
double c_pos[3];
double g_rb[3];
double t_pos[3];
int checksum;
}commData;
```

The total size of the packet payload was 120 bytes. The packet reflector program is a simple UDP server which receives predefined packets at a port and reflects them back to the sender with time stamp information. Packet round-trip time was used in analyzing the delay characteristics because of the potential drifts in system clocks at the server and the client computers. The packet sequence number was used to detect the out of sequence packets and the checksum field was used to detect corrupt packets. The simple checksum method used in this experiment is the integer value of the sum of all data arrays and the sequence number in the packet data structure.

The network topology for the delay characterization experiment is shown in Fig. 1. Both our lab and the North

```
1    astrovac-V1.cac.washington.edu (128.95.205.100)
2    uwbr-ads-01-vl1998.cac.washington.edu (140.142.155.23)
3    hnsp2-wes-ge-0-0-0-0.pnw-gigapop.net (209.124.176.12)
4    abilene-pnw.pnw-gigapop.net (209.124.179.2)
5    dnvrng-sttlng.abilene.ucaid.edu (198.32.8.50)
6    kscyng-dnvrng.abilene.ucaid.edu (198.32.8.14)
7    iplsng-kscyng.abilene.ucaid.edu (198.32.8.80)
8    chinng-iplsng.abilene.ucaid.edu (198.32.8.76)
9    nycmng-chinng.abilene.ucaid.edu (198.32.8.83)
10   washng-nycmng.abilene.ucaid.edu (198.32.8.85)
11   rlgh1-gw-abilene-oc48.ncren.net (198.86.17.65)
12   rlgh7600-gw-to-rlgh1-gw.ncren.net (128.109.70.38)
13   ncsu7600-gw-to-rlgh7600-gw.ncren.net (128.109.70.26)
14   ncsu7609-1-to-ncsu7600-gw.ncren.net (128.109.70.46)
15   cmdfcore-6509-1-gi3-8.ipt.ncstate.net (152.1.6.205)
16   cmdfhub-6509msfc-2.ncstate.net (152.1.7.73)
17   surf.imaging.ncsu.edu (152.14.96.140)
```

Fig. 2. Route between University of Washington and North Carolina State University

Carolina State University are connected via the Abilene Internet2 backbone network. It guarantees a 100Mbps connection between any two computers connected via the network. The connection to Scuola Superiore Sant'Anna consists of three main networks: the Abilene, GEANT2 and GARR. Both GEANT2 and GARR are European research networks that guarantee high speed connectivity. A switching node at New York City connects GEANT2 to Abilene.

The packet switching routes for the two servers were obtained using the *traceroute* application and are shown in Figs. 2 and 3. There were 17 hops to server at Raleigh, NC and 25 hops to Pontedera, Italy.

B. Delay Emulator Experiment Setup

The delay emulator experiment setup consists of three workstations in a local network, with one acting as a router between the other two. The router (WS 3) was a RedHat Linux AMD Athlon 1.2 GHz computer with 256 MB RAM and 10/100 Mbs network cards. The other two workstations are AMD Opteron 1.5 GHz with 1 GB RAM and Intel Pentium 4 2.66 GHz with 1 GB RAM with both running on Fedora Core 4. They also have 10/100 Mbs high speed Ethernet cards for communication. The local network configuration is shown in Fig. 4. The router has two network interface cards connected to it with IP addresses 192.168.0.1 and 192.168.1.1 respectively. The workstations WS 1 and WS 2 have IP addresses 192.168.0.5 and 192.168.1.5 and are connected to each of the network interfaces of the router and their default gateway is set appropriately. The router was configured to transfer packets from each of the two workstations by establishing a local network. NIST Net was running on WS 3.

V. Experimental Procedure

A. Delay Characterization Experiment Procedure

For the delay characterization experiment, the packet reflector program was started at our server sites. For each trial 20,000 UDP data packets were transmitted to the servers. The packet rate was controlled by the haptic servo loop program.

```
1    astrovac-V1.cac.washington.edu (128.95.205.100)
2    uwbr-ads-01-vl1998.cac.washington.edu (140.142.155.23)
3    hnsp2-wes-ge-0-0-0-0.pnw-gigapop.net (209.124.176.12)
4    abilene-pnw.pnw-gigapop.net (209.124.179.2)
5    dnvrng-sttlng.abilene.ucaid.edu (198.32.8.50)
6    kscyng-dnvrng.abilene.ucaid.edu (198.32.8.14)
7    iplsng-kscyng.abilene.ucaid.edu (198.32.8.80)
8    chinng-iplsng.abilene.ucaid.edu (198.32.8.76)
9    nycmng-chinng.abilene.ucaid.edu (198.32.8.83)
10   198.32.11.51 (198.32.11.51)
11   so-7-0-0.rt1.ams.nl.geant2.net (62.40.112.133)
12   so-6-2-0.rt1.fra.de.geant2.net (62.40.112.57)
13   so-6-2-0.rt1.gen.ch.geant2.net (62.40.112.21)
14   so-2-0-0.rt1.mil.it.geant2.net (62.40.112.34)
15   garr-gw.it1.it.geant.net (62.40.103.190)
16   rt1-mi1-rt-mi2.mi2.garr.net (193.206.134.190)
17   rt-mi2-rt-to1.to1.garr.net (193.206.134.42)
18   rt-to1-rt-pi1.pi1.garr.net (193.206.134.74)
19   rt-pi1-ru-unipi-1.pi1.garr.net (193.206.136.14)
20   ser-smr.unipi.it (131.114.191.186)
21   jsmr-bd.unipi.it (131.114.191.206)
22   sssup-gw.unipi.it (131.114.191.42)
23   * * *
24   * * *
25   euron.sssup.it (193.205.82.131)
```

Fig. 3. Route between University of Washington and Scuola Superiore Sant'Anna

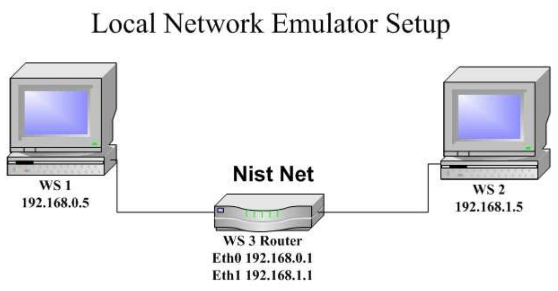

Fig. 4. Delay emulator experiment setup

For this work we used the *OpenHaptics Toolkit* from Sensable, Inc. The packet reflector program waits for 100 seconds after the last packet was sent before starting the next trial. There were 20 trials in total, 10 for each of the two servers.

B. Delay Emulator Experiment Procedure

For delay emulation we used WS 1 as the server and the packet reflector program was run on that machine. WS 2 was used to send UDP data packets through the router. We also used 20,000 UDP packets for every trial. The NIST Net parameters, mean delay in ms, and standard deviation in ms were varied according to Table I. Since delay was included in both directions from WS2 to WS1, the expected round-trip mean delay and standard deviation are also shown in Table I. NIST Net allows the input from a user-defined table to control the statistics of the generated delay values. In this case, we used tables from our characterization experiments to Italy. There were five trials for each delay condition and the trials were repeated for the packet rates of 1000, 500, 333, and 250 packets/sec respectively. Altogether, there were 120

TABLE I

EMULATOR DELAY PARAMETERS INPUT AND EXPECTED ROUND-TRIP CHARACTERISTICS

Delay ms	Std ms	RT Delay ms	RT Std ms
50	5	100	7.07
50	10	100	14.14
100	5	200	7.07
100	10	200	14.14
150	5	300	7.07
150	10	300	14.14

trials combining all input delay conditions and packet rates. The entire experiment procedure was repeated to study the behavior of NIST Net with the default input table that comes with the software.

VI. RESULTS

A. *Delay Characterization Test Results*

The test results from sending 20,000 UDP data packets to the server at North Carolina State University and Scuola Superiore Sant'Anna are given in Table II. The round-trip delay distribution from both experiments for one of the trials is shown in Figs. 5 and 6. The delay distribution to NCSU was mostly concentrated on two peaks close to the mean delay value of 82.52 ms as expected for the mean standard deviation value of 1.69 ms. The delay distribution for Italy is characterized by a "long tail" to the right (longer delay times) compared to a symmetric normal distribution. To better understand the delay characteristics, a phase plot in which the current value of the round-trip delay *n* is plotted against the next value *n+1* is shown in Fig. 7. At each router the UDP data packets get in a queue along with other Internet packets like TCP and FTP. Since our test packets are generated and transmitted at a rate of 1000 Hz, which corresponds to a 1 ms time interval, some of the series of these packets get in queue behind other Internet packets in a buffer at the routers. The router then processes these packets after finishing up the Internet packets already in queue. This phenomenon is called "probe compression" [12], in which a sequence of probe UDP packets that are generated at very high rates gets compressed between other Internet packets at the queue. The delay time is highly correlated for such packets and they accumulate along the slope of a line as shown in Fig. 7. Two trials were also conducted with a packet transmission rate of 10 and 2 packets/sec and their phase plot is shown in Figs. 8 and 9. These show that the delay values are much less correlated for lower packet rates.

B. *Results from the Delay Emulator Experiment*

Data from the Italy experiments were used as an input to the NIST Net emulator along with the default input table that comes with the software. On our initial run we found that there was more than 50% packet loss for trials at a packet rate of 1000 Hz. Most of the packet loss was due to out-of-sequence

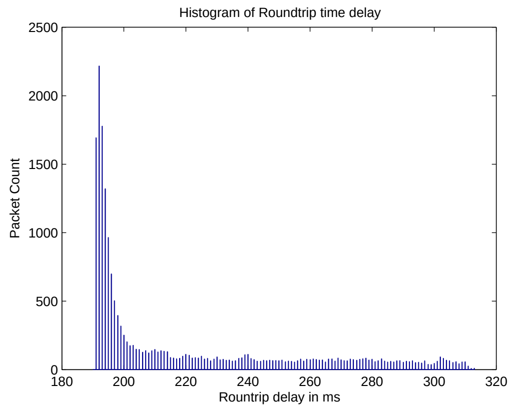

Fig. 5. Round-trip delay distribution for packets from Seattle to Italy

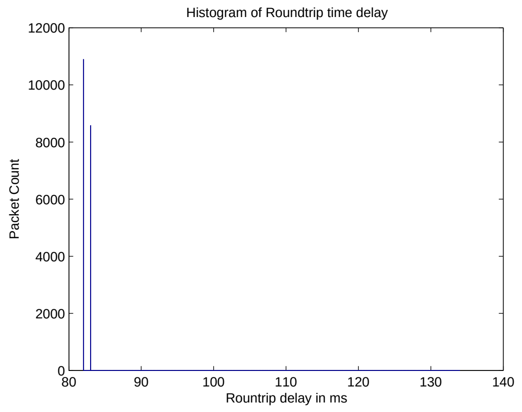

Fig. 6. Round-trip delay distribution for packets from Seattle to Raleigh

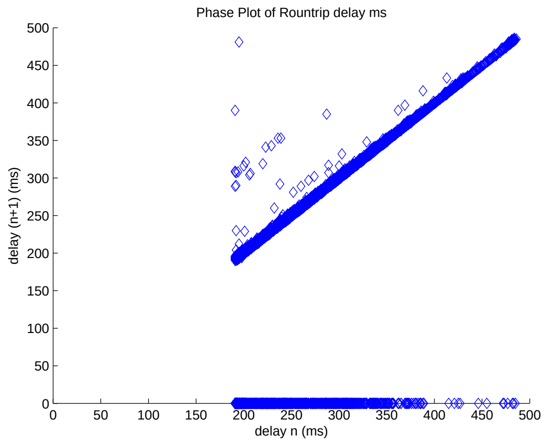

Fig. 7. Phase plot of round-trip delay for a packet interval of 1 ms

TABLE II
AVERAGE VALUES FROM INTERNET DELAY CHARACTERIZATION EXPERIMENTS

Location	**RT mean ms**	**RT Std ms**	**Lost+Dropped packets %**
Italy	222.12	38.38	4.4
NCSU	82.52	1.69	2.4

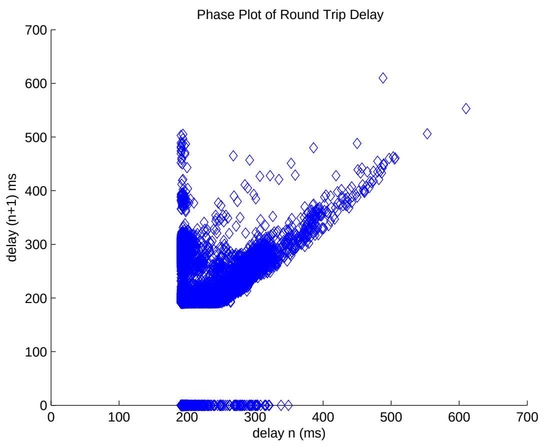

Fig. 8. Phase plot of round-trip delay for a packet interval of 100 ms

packets and happened at the FIFO of the output buffer of NIST Net. When each packet arrives at the network interface of the router, they are time-stamped and the output time for each packet is calculated based on statistically generated time using the empirical distribution curve provided at the input. When the desired standard deviation is greater or equal to the interarrival time of the data packets —which are 1, 2, 3 and 4 ms in this case— a lot of the UDP packets get swapped at the FIFO. This situation is unrealistic given that the real Internet routers have deterministic FIFOs. Therefore, we decided to use a NIST Net model with a modified FIFO, which prevents the datagram packets from swapping while it was waiting at the

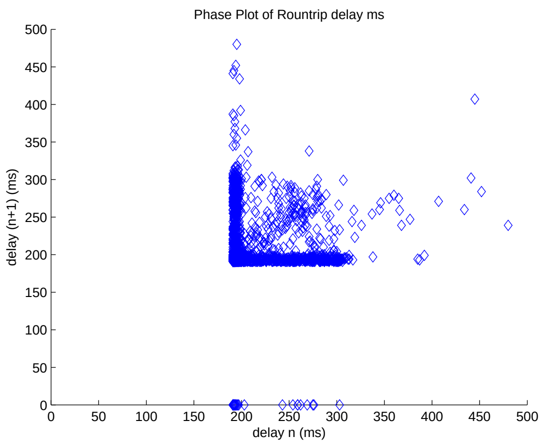

Fig. 9. Phase plot of round-trip delay for a packet interval of 500 ms

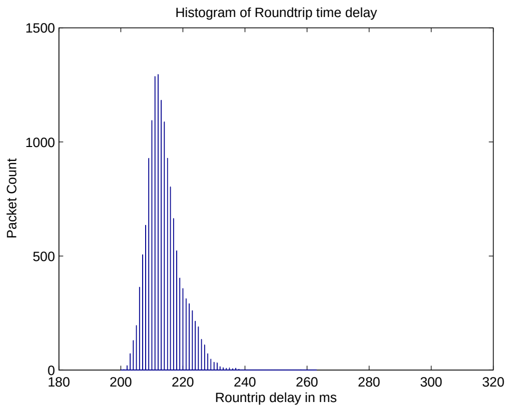

Fig. 10. Round-trip delay plot emulated using NIST Net default parameters table for expected delay condition of 200 ms and std 7.07 ms

output buffer. This modification forces the packet to stay in the output queue until all other packets before it have exited. As a result of this modification there are noticibly two effects:

- Each packet acquired additional queuing delay.
- The resultant distribution was different from the expected distribution based on the input data table.

The NIST Net-emulated delay distribution curves using both default and measured Italy parameters are shown in Figs. 10 and 11 repectively. The packet rate was 1000 Hz corresponding to an expected mean delay value of 200 ms and a standard deviation of 7.07 ms. The mean delay emulated when using default parameters was 213 ms with a standard deviation of 5.4 ms. Using the Italy parameters the mean delay was 216 ms with a standard deviation of 3.8 ms. In Fig. 12 we show the measured mean delay in emulator experiments versus the mean specified to NIST Net using the Italy acquired parameters. For an expected mean delay of 100 ms, the actual mean delay varied from 116 ms to 108 ms for packet transmission rates of 1000 Hz to 250 Hz. Similarly, for an expected mean delay of 200 ms, the actual mean delay varied from 216 ms to 208 ms and for an expected mean delay of 300 ms the actual mean varied between 316 ms to 308 ms.

In Fig. 13 we show the measured standard deviation in emulator experiments versus the standard deviation specified to NIST Net using the Italy acquired parameters. For an expected standard deviation of 7.07 ms, the actual standard deviation varied from 3.69 ms to 6.01 ms for a packet transmission rate of 1000 Hz to 250 Hz. Comparatively, for an expected standard deviation of 14.14 ms the actual standard deviation varied from 5.53 ms to 10.08 ms.

In Fig. 14 we show the percentage of data packets rejected at WS2 out of 20,000 packets for the emulation experiment using the Italy acquired parameters at each packet transmission rate. The packet rejection includes corrupt and out of sequence packets. At a packet transmission rate of 1000 Hz, the percentage of packets rejected was 27.6%, 27.5% and 28.3% for expected mean delay values of 100, 200 and 300 ms and a standard deviation of 7.07 ms. Analogously, for a packet

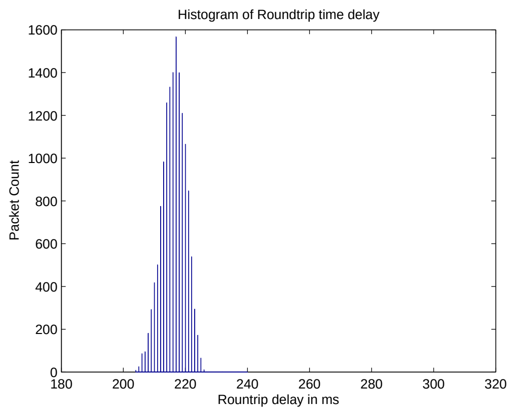

Fig. 11. Round-trip delay plot emulated using measured Italy parameters table for expected delay condition of 200 ms and std 7.07 ms

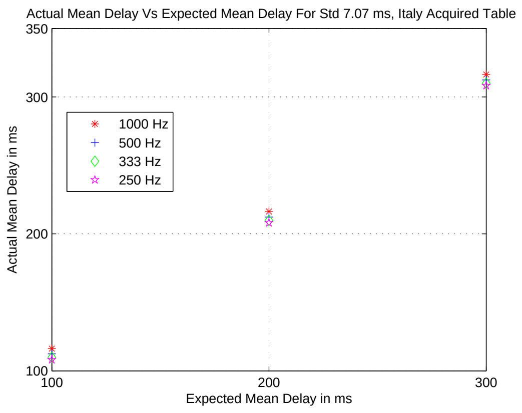

Fig. 12. Actual mean delay versus expected mean delay for std of 7.07 ms

transmission rate of 500 Hz the percentage of packets rejected was 21.9%, 21.5% and 21.7%; for a packet tranmission rate of 333 Hz it was 17.15%, 17.18% and 16.8%, and for a packet transmission rate of 250 Hz it was 13%, 12.9% and 12.5%.

In Figs. 15 and 16 the phase plot of emulated data using Italy acquired parameters for a packet transmission rate of 1000 and 250 Hz are shown. The delay parameters specified for the NIST Net were a mean delay of 200 ms and a standard deviation of 14.14 ms. For the packet transmission rate of 1000 Hz the delays varied from 212 ms to 254 ms and for the packet transmission rate of 250 Hz, they varied from 189 ms to 253 ms. Both phase plots show higher correlation for the delay values.

C. Simulation of out of Sequence Packets

A simulation of time-varying delays was performed based on equation 5 to mathematically predict an expected number of out of sequence packets. This simulation refers to software code running on a computer that calculates and applies delays to virtual packets generated during the simulation, as opposed to an emulator where the delays are applied directly to real packets connected via hardware. A plot of the percentage of out of sequence packets that arrived at WS2 for an expected

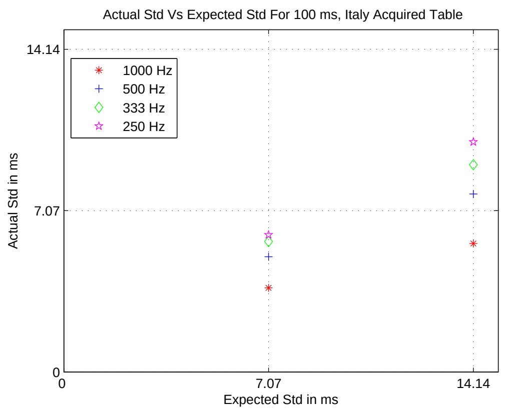

Fig. 13. Actual standard deviation versus expected standard deviation for 100 ms, Italy acquired parameters table

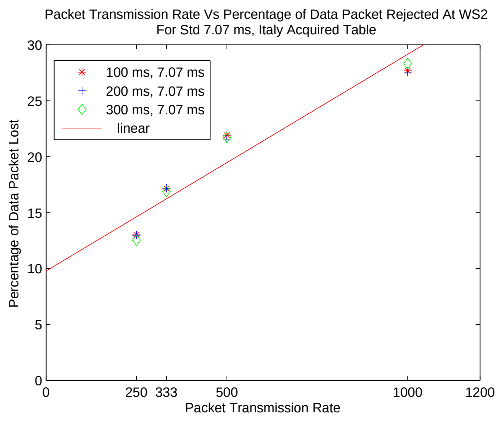

Fig. 14. Percentage of packets rejected at WS2 versus packet transmission rate

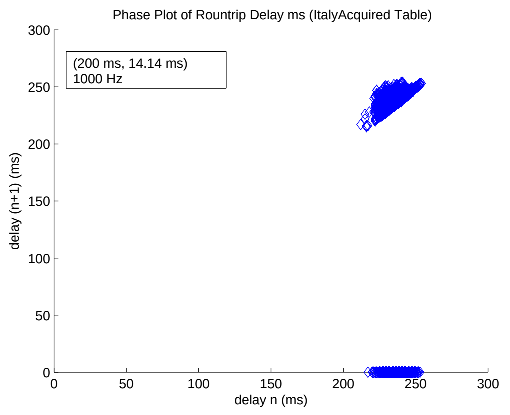

Fig. 15. Phase plot of emulated round-trip delay for a packet transmission rate of 1000 Hz and input delay parameters of 200 ms mean delay and 14.14 ms standard deviation.

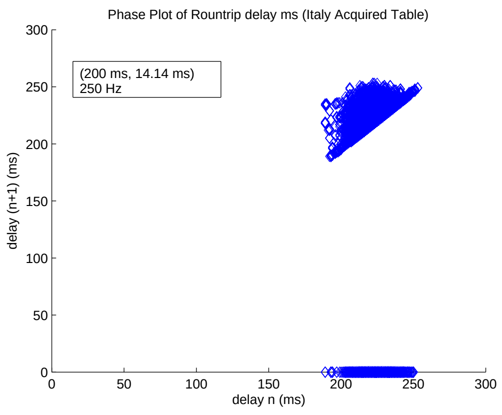

Fig. 16. Phase plot of emulated round-trip delay for a packet transmission rate of 250 Hz and input delay parameters of 200 ms mean delay and 14.14 ms standard deviation.

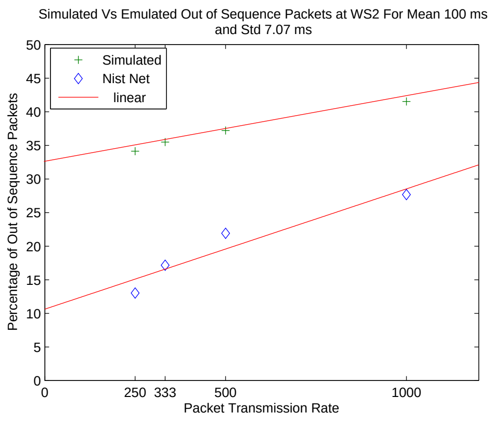

Fig. 17. Comparison of simulated versus emulated out of sequence packets for input delay condition of 100 ms mean delay and 7.07 ms standard deviation

delay of 100 ms and a standard deviation of 7.07 ms using the Italy acquired table is shown in Fig. 17. Also in the same plot the simulated expected percentage of out of sequence packets were plotted. The theoretical percentage value was 41.5%, 37.2%, 35.4% and 34.1% for packet transmission rates of 1000, 500, 333 and 250 Hz respectively. Their actual out of sequence packet percentages were 27.6%, 21.91%, 17.1567% and 13%.

VII. Discussion

We have implemented a NIST Net network emulator and configured it to recreate time varying delay conditions for packet types, sizes, and rates typical of NHVEs. This emulator can take experimental distribution values to generate time varying random delays. We found that the emulated mean delay values were higher than the desired output for all the four packet transmission rates. In addition, the emulated standard deviation was lower than desired for all the four packet transmission rates. Also, the emulated delay distribution was different than the actual distribution obtained using the delay characterization experiments. This deviation was due to the FIFO modification needed to avoid packet reordering. The percentage of out of sequence packets was considerably reduced due to the modification.

The packet loss was higher for a packet transmission rate of 1000 Hz, which is the typical rate of haptic applications. While it was lower for a packet transmission rate of 250 Hz, at this low packet transmission rate the networked haptic application may not perform well or may be unstable. We have further extended this work by testing a NHVE using NIST Net and compared its performance to that of using the real Internet [14].

The advantage of using this implementation of network emulator consists in its ability to recreate various network conditions in a laboratory setting. The input parameters for the emulator can be modified to generate the expected delay values. Moreover, NIST Net has additional input parameters like bandwidth, congestion, and random drop that could be used to emulate varying bandwidth, *QoS*, and wireless like networks. The emulator can also be used in testing teleoperation and other similar systems.

Acknowledgments

We would like to thank Oliver Tonet at the Scuola Superiore Sant'Anna, Pontedera, Italy and Krish Karthik, North Carolina State University, Raleigh for hosting the packet reflector program. Funding from Boeing and US Army, Medical Research and Material Command, grant number DAMD17-1-0202.

References

[1] C.B.Zilles and J.K. Salisbury. A constraint-based god-object method for haptic display. In *Proc.IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 146–151, Pittsburgh,PA, 1995.

[2] Mark Carson and Darrin Santay. Nist net: a linux-based network emulation tool. *SIGCOMM Comput. Commun. Rev.*, 33(3):111–126, 2003.

[3] H.R.Choi, B.H.CHoi, and S.M.Ryew. Haptic display in the virtual collaborative workspace shared by multiple users. In *Proc. IEEE International Workshop on Robot and Human Communication(RO-MAN'97)*, pages 478–483, 1997.

[4] Kenji Hikichi, Hironao Morino, Yasuhiko Yasuda, Isamu Arimoto, and Kaoru Sezaki. The evaluation of adaptation control for haptics collaboration over the internet. In *Proc. IEEE Communications Quality and Reliability(CQR) International Workshop*, May 2002.

[5] Yutaka Ishibashi, Takahiko Kanbara, Takahiko Hasegawa, and Shuji Tasaka. Traffic control of haptic media in networked virtual environments. In *Proc. IEEE Workshop on Knowledge Media Networking*, pages 11–16, 2002.

[6] Yutaka Ishibashi, Takahiko Hasegawa, and Shuji Tasaka. Group synchronization control for haptic media in networked virtual environments. In *Proc. IEEE International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems(HAPTICS'04)*, pages 106–113, 2004.

[7] Joao P. Hespanha, Margaret McLaughlin, Gaurav S. Sukhatme, Minoo Akbarian, Rajiv Garg, and Weirong Zhu. Haptic collaboration over the internet. In *Proc.Fifth PHANToM Users' Group Workshop(PUG00)*, pages 9–13, Aspen,CO, 2000.

[8] Pietro Buttolo, Roberto Oboe, and Blake Hannaford. Architectures for shared haptic virtual environments. *Computer and Graphics*, 21(4):478–483, 2000.

[9] Jung Kim, Hyun Kim, Hyun K. Tay, Manivannan Muniyandi, Mandayam A. Srinivasan, Joel Jordan, Jesper mortensen, Manuael oliveira, and Mel Slater. Transatlantic touch: A study of haptic collaboration over long distance. *PRESENCE*, 13(3):328–337, 2004.

[10] Ganesh Sankaranarayanan and Blake Hannaford. Virtual coupling schemes for position coherency in networked haptic environments. In *Proc. BioRob 2006. The First IEEE/RAS-EMBS International Conference On Biomedical Robotics And Biomechatronics.*, pages 853–858, Pisa,Italy, 2006.

[11] M. Fotoohi, S. Sirouspour, and D. Capson. Multi-rate control architectures for dextrous haptic rendering in cooperative virtual environments. In *Proc. IEEE Conference on Decision and Control*, pages 4478–4483, San Diego, CA, USA, 2006.

[12] Jean-Chrysotome Bolot. End-to-end packet delay and loss behavior in the internet. In *SIGCOMM '93: Conference proceedings on Communications architectures, protocols and applications*, pages 289–298, New York, NY, USA, 1993. ACM Press.

[13] Vern Paxson. End-to-end internet packet dynamics. *IEEE/ACM Trans. Netw.*, 7(3):277–292, 1999.

[14] Ganesh Sankaranarayanan and Blake Hannaford. Comparison of performance of virtual coupling schemes for haptic collaboration using real and emulated internet connections. In *proceedings of ROBOCOMM, the first International Conference on Robot Communication and Coordination*, Athens, Greece, 2007.