Experimental Performance Evaluation of Networked Virtual Reality Services

Lea Skorin-Kapov§, Danko Vilendečić†, Dario Mikić†

§Research and Development Centre
Ericsson Nikola Tesla
Krapinska 45, 10000 Zagreb, Croatia
Tel: +385 1 365 39 65 Fax: +385 1 365 35 48 E-mail: lea.skorin-kapov@etk.ericsson.se

†Department of Telecommunications
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia
Tel: +385 1 612 97 38 Fax: +385 1 612 98 32 E-mail: danko.vilendecic@fer.hr

Abstract - Networked virtual reality (NVR) services with integrated multimedia components and perceived "real-time" interactivity impose certain Quality of Service (QoS) requirements at the user/application level as well as on the underlying network. In this paper we are concerned with measuring end-to-end performance parameters for NVR services and determining the effect of various network conditions on user perceived quality. We discuss measurements performed in an emulated network environment on four heterogeneous NVR applications. Test procedures have been outlined and results have been analyzed to determine the network requirements of each application.

I. INTRODUCTION

Networked virtual reality services with integrated multimedia components and perceived "real-time" interactivity may be considered a good representative of advanced services in the new generation network [9]. Such services impose certain QoS requirements at both the user/application level as well as on the underlying network [10]. Network level QoS parameters related to NVR services include bandwidth (to support large volumes of media data and multiple users), latency and jitter (support for "real-time" interactivity), and reliability. Multi-user virtual environments are thus characterized by unpredictable traffic flows due to dynamic user interactions. Existing 3GPP standards describe the QoS requirements of various types of services and define accepted parameter values from an end user viewpoint [1,2]. However, no clear description is provided describing the requirements of NVR services.

This paper describes measurements of QoS parameters that have been performed using four different prototype NVR applications, grouped together as single-user and multi-user virtual environments. The idea in each case was to determine the effects of various network conditions on user perceived quality. Under certain conditions (i.e. if an end user has limited bandwidth due to access network capabilities), a service may not achieve its intended functionality and may therefore be considered unacceptable. A set of network requirements for a service is therefore needed for the adaptation and negotiation of QoS between a VR service and a VR end user. The performed measurements provide some empirical data on the QoS requirements of different types of NVR services.

The paper is organized as follows: Section II discusses related work addressing the network requirements of NVR services. Section III describes the testbed used for measurements and the *NIST Net* tool used for emulating network parameters. Sections IV and V cover the test procedures and achieved results for measurements performed on single-user and multi-user virtual reality applications. A discussion and conclusions are presented in section VI.

II. RELATED WORK

Previous research on NVR communication requirements includes a comprehensive overview of communication architectures, protocols, and mechanisms [17]. Managing dynamic shared state and resource management have been considered as key issues for achieving scalability and performance.

In [8], bandwidth, latency, distribution schemes, and reliability are identified as critical when addressing network QoS for large scale and distributed virtual environments (VE). VEs supporting collaborative experiences among users, termed collaborative virtual environments (CVE), in general require high bandwidth and low latency in order to maintain natural interactions. The effects of latency and jitter on human performance were measured in [13], with multiple users cooperating on a teleoperation task in a CVE. Another example may be found in [6], where the effects of latency and jitter have been tested on haptic force feedback display in the teleoperation of a distant microscope. When considering a virtual environment with integrated multimedia components, the quality of the VE depends largely on the quality of the synchronization between media components. Some experimental values describing human perception of media synchronization in terms of delay and jitter can be found in [20].

3GPP standards define 4 different QoS classes based on delay tolerance [2]:

- Conversational (real-time) class
- Streaming class
- Interactive class
- Background class

Key performance parameters and target values have been outlined for each QoS class in [1]. The parameter values are commonly accepted values from an end-user viewpoint as proposed in [4]. An example of the end user QoS requirements for the conversational class is given in Table I (taken from [1]).

Table I. End-User Performance Expectations – Conversational/Real-Time Services.

Medium	Application	Degree of symmetry	Data rate	Key performance parameters and target values		
				End-to-end one way delay	Delay variation within a call	Information Loss
Audio	Conversational voice	Two-way	4-25 kb/s	<150 ms preferred <400 ms limit	<1 ms	<3% FER
Video	Videophone	Two-way	32-384 kb/s	<150 ms preferred <400 ms limit Lip-Synch: <100 ms		<1% FER
Data	Telemetry -two way control	Two-way	<28.8 kb/s	<250 ms	N.A.	Zero
Data	Interactive games	Two-way	<1 KB	<250 ms	N.A.	Zero
Data	Telnet	Two-way (asymmetric)	<1 KB	<250 ms	N.A.	Zero

Due to the fact that no clear description is provided of the QoS requirements of NVR services, our goal was to address this area. Performance measurements have been conducted for different NVR services to determine the effects of various network QoS values on end-user perceived quality.

III. TESTBED CONFIGURATION

In our testbed, we used a *NIST Net* software emulator [3] to test the behavior of each prototype application under various network conditions.

The *NIST Net* network emulator tool, implemented as a kernel module extension to Linux, was installed and used to emulate numerous network conditions. Performance scenarios which can be emulated include bandwidth limitations, tunable delay distributions, congestion and background loss, and packet reordering/duplication. *NIST Net* offers a command line interface which allows the user to define desired performance parameters for a selected IP traffic stream passing through the router. It was thus possible emulate various network conditions.

Maximum bandwidth in the testbed configuration corresponds to 10 Mbits/s. In addition, delay was measured to be < 1ms and packet loss 0%. By using *NIST Net* desired values for these parameters could be set.

The testbed configuration used to perform measurements is shown in Figure 1. The following hardware and software has been used:

- **PC 1**: Pentium IV (1.6 GHz, 512 MB RAM) with Linux 2.4.17 (RedHat 7.2) OS. Additional software: NIST Net network emulation package version 2.0.10.
- **PC 2**: Pentium III (750 MHz, 256 MB RAM) with Windows 2000 Professional OS. Additional software: Cortona 3.0 VRML plug-in; Blaxxun Contact multi-user 3D plug-in; Java Media Framework 2.1.1. Beta 3; SteadyHand 1.0 Dynapel Systems Inc. (for analyzing video frames).
- **PC 3**: Pentium III (750 MHz, 256 MB RAM) with Windows 2000 Professional OS. Additional software: Cortona 3.0 VRML plug-in; Blaxxun Contact multi-user 3D plug-in; Java Media Framework 2.1.1. Beta 3; Apache httpd Server v.1.2.17; Savant Web Server; Alicebot.net Server 4.0 (Demy).

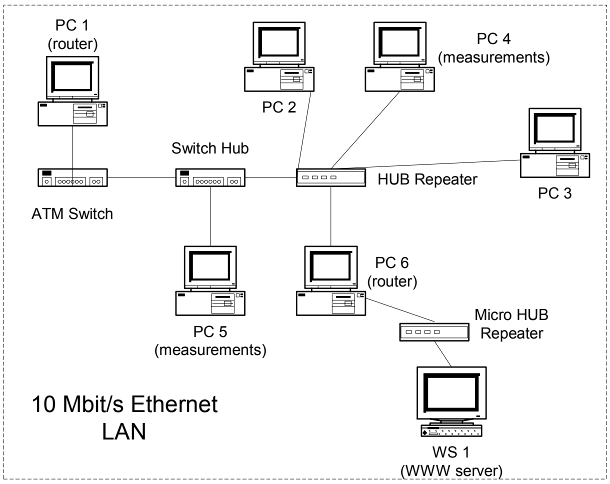

Figure 1. Testbed Configuration.

- **PC 4**: Pentium I (133 MHz, 64 MB RAM) with Linux 2.4.0 (Debian) OS. Additional software: Ethereal (network protocol analyzer) version 0.8.16.
- **PC 5**: Pentium III (600 MHz, 256 MB RAM) with Windows 2000 Professional OS. Additional software: Cool Edit 2000 (for audio recording).
- **PC 6**: Pentium II (200 MHz, 96 MB RAM) with Linux 2.4.0 (Debian) OS.
- **WS 1**: Sun Sparc Ultra 5, Sparc v9 (270 MHz, 256 MB RAM) with Solaris 8 OS. Additional software: Blaxxun Virtual Worlds Platform 5.1.

IV. SINGLE USER VIRTUAL ENVIRONMENTS

In the case of single user VEs, we are looking only at interactions between the user and the environment, rather than interactions between multiple users. The VE can be downloaded from the Internet or locally resident. In this case we are concerned with the network QoS requirements for VEs on the Internet. One of the key issues is the time necessary for scene download. The question concerning the end user is how long of a wait may be considered acceptable. User interactions with a VE may require additional network traffic, such as requesting one way audio/video streaming, or additional file download. For example, if a virtual world consists of a number of virtual spaces that are not all initially downloaded, user navigation from one space to another causes additional download and deterioration of navigation quality [12]. In this paper, we address two different test cases involving single user VEs.

A. *Test Procedure*

In the first case, a user accesses a virtual gallery of mobile phones from a WWW server and then freely navigates through the gallery. In the second case, we look at an example conversational virtual character designed for the Web that is capable of having a meaningful conversation with a user who types in the input [19]. The key parameter influencing performance is the time to answer (TTA), or the time that the user must wait for a response.

In both cases, measurements were performed with PC 2 acting as a WWW client and PC 3 as a WWW server. All traffic was routed over PC 1 (running NIST Net) in order to control network parameters.

Traffic capture was performed on PC 4 using Ethereal version 0.8.16 [5]. *Ethereal* is a network protocol analyzer that enabled us to capture, filter, and analyze network traffic. Measurements were conducted according to the following steps:

- Start NIST Net on PC 1 and set desired network parameters for all traffic being routed from PC 2 to PC 3 and vice versa.
- Restart IE browser on PC 2, clear the memory cache and the disk cache.
- Start Ethereal capture on PC 4.
- Generate network traffic (click to download, or in the case of the conversational character, send text input and wait for response).
- Stop Ethereal capture.

Download time and TTA were measured from the time that the initial request was sent from PC 2 up until the last packet containing data from PC 3 arrived at PC 2.

B. *Results and Analysis*

1) *Virtual Gallery*

Virtual Gallery, shown in Figure 3, is an example of a single user virtual environment containing a number of virtual models of Ericsson mobile phones. A user can access this gallery from a WWW server, in which case the gallery is downloaded and the user is then free to navigate through the gallery. The user must have a VRML plug-in installed (i.e. Cortona). In this case, the only network traffic being generated is during download. Due to the fact that traffic was sent using TCP, it was expected that network delays and packet loss would have a direct effect on throughput due to TCP's congestion control mechanism. The total file size (including all *wrl* files and texture files) corresponding to the Virtual Gallery is 1,517 Mbytes. Measurements were conducted under various emulated network conditions. The results are shown in Table II.

Values for maximum bandwidth, delay and packet loss indicated in the table were set for both traffic being routed from client to server and from server to client. Therefore, a delay of 50 ms as shown in the table corresponds to 100 ms RTT (assuming symmetric delays).

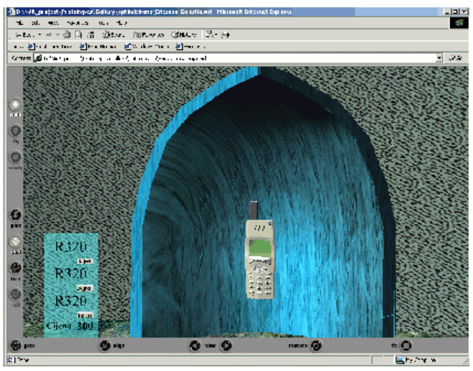

Figure 3. Virtual Gallery

Table II. Measurements of download time for Virtual Gallery

Max. Bandwidth	Delay [ms]	Packet Loss [%]	Average Download Time [s]	Download Time [s] with 95 % confidence level
10 Mbit/s	0	0	3,644	3,644 ± 0,134
10 Mbit/s	50	0	14,429	14,429 ± 0,955
10 Mbit/s	50	1	20,391	20,391 ± 0,920
10 Mbit/s	50	2	26,757	26,757 ± 2,412
375 kbit/s	0	0	34,029	34,029 ± 1,879
375 kbit/s	100	0	37,677	37,677 ± 0,868

Average download time in each case was calculated based on five repeated tests. Download time within a confidence interval of 95% was calculated based on *Student's t distribution*. The results show a clear increase in download time caused by an increase in delay and packet loss. This was expected due to TCP's congestion control mechanism, where the transmission rate of the TCP sender is determined by the level of congestion in the network. The modeling of TCP's steady state throughput as a function of loss rate and round trip time (RTT) can be found in [14]. We also see an increase in delay variation due to packet loss (greater range of values fitting into the 95 % confidence interval).

The main question in this case concerning the end user is how long the user is willing to wait for download. If we assume up to 30 s to be an acceptable download time, we can see that at 375 kbit/s (an example UMTS data rate in a suburban area) download time is already greater than 30 seconds. It is possible to then conclude that a decrease in data rate or an increase in delay would result in unacceptable waiting time.

2) *Demy: A Conversational Virtual Character*

Demy is an example of a conversational virtual character on the Internet. The user downloads a web page (using current versions of Netscape or IE) containing a Java applet displaying the animated VRML character (Demy) and a text box (Figure 4). The virtual character is rendered using *Shout 3D* technology (written in Java), thus eliminating the need for any extra plug-ins or downloads.

When the user types English text in the text box, Demy replies by talking. The artificial intelligence of the virtual character is based on the latest Java implementation of ALICE [19]. The answers are based on the "knowledge" which is contained in an AIML (Artificial Intelligence Markup Language) file. This file contains answers to known questions and rules for interpreting users' inputs and providing answers. The animated virtual character (Java applet) can be controlled by JavaScript and instructed to talk. The speech is stored on the standard HTTP server in the form of audio (*.au*) files and MPEG-4 lip synchronization files (*.fba*). These files are streamed on-the-fly when speech is requested. Details on the system architecture of this application can be found in [19].

The applet size is 223 Kbytes (Shout 3D rendering engine is 187 Kbytes, and the facial animation player implementation is 36 Kbytes) and the facial model size is 32 Kbytes.

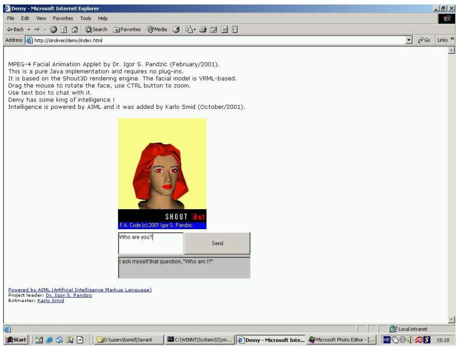

Figure 4. Demy: conversational virtual character

Measurements were performed to determine the time necessary for the virtual character to respond to a question asked by the user (TTA). This involved the generation and download of an *.au* file and an *.fba* file. Measurements of TTA where tests were performed in a best effort Internet environment can be found in [19]. Our goal was to perform these measurements in an emulated network environment where we could control network parameters. We measured the TTA for two different responses: ”*I ask myself that question, who am I ?*” (in answer to user input: “Who are you?”) and “*Hi there!*” (in answer to user input: “Hello”). TTA is defined as the time from when the user presses SEND until the moment when Demy starts to talk. Results are shown in Table IV. An increase in the size of the response corresponds to an increase in TTA.

The user perceived quality of virtual characters on the Internet depends largely on the purpose of the application, whether it be entertainment, commerce, education, or personal communications. The issue is to determine the requirements of the application in order for it to be functional and attractive. In the case of Demy, where we have a conversational virtual character, long response delays during the chat reduce the user feeling of ”real-time” interactivity. For practical implementation of this prototype, we feel that better results are necessary than those shown in Table IV. A TTA within the range of 1-2 s may be considered acceptable.

Table IV. Measurements of TTA for Demy

Max. Bandwidth	Delay [ms]	Packet Loss [%]	Average Response Time[s]	TTA [s] with 95% confidence level
Response: "*Hi there!*" (.fba file size = 7,33 KB, .au file size = 68 B				
375 kbit/s	0	0	4,724	4,724 ± 0,032
46,875 kbit/s	150	0	9,150	9,150 ± 0,048
Response: "*I ask myself that question, who am I ?*" (.fba file size = 23,4 KB, .au file size = 159 B)				
375 kbit/s	0	0	10,093	10,093 ± 0,165
46,875 kbit/s	150	0	13,009	13,009 ± 0,228

V. MULTI-USER VIRTUAL ENVIRONMENTS

In multi-user VEs, multiple users from geographically distributed locations can communicate, collaborate or interact with each other and the environment. Such services often require large bandwidth and low latency. In order to consider some of the requirements of such services, measurements were performed on two applications. The first is a real-time virtual audio chat between multiple users where we address audio-streaming as one of the key communication capabilities used in VEs. In the second case we look at a shared virtual community built upon Blaxxun’s Virtual Worlds Platform where multiple users can meet and interact.

A. Virtual Audio Chat

Virtual Audio Chat (VAC) is an application that enables real-time audio communication over the Internet between a multiple number of users [11]. The user interface includes a VRML model of a mobile phone (Figure 5). By way of user interactions implemented in VRML, a user can enter onto the phone the IP address and port number of another user (or a multicast address/port for a group session) with whom he wishes to communicate. A Java applet (using the Java Media Framework JMF RTP API) then opens an RTP based audio streaming session using the entered address and port, and waits for the other user to join in an analogous manner.

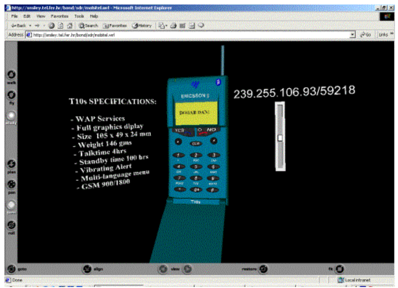

Figure 5. VRML model of mobile phone.

The network traffic corresponding to this application includes TCP traffic during initial download, and afterwards a continuous RTP stream for the duration of the audio chat. An audio chat was started between two users and measurements of bandwidth, delay, jitter, and packet loss during RTP streaming were performed to determine their effect on user perceived quality.

1) Test Procedure

The VRML mobile phone model was downloaded from WS 1 onto PC 2 and PC 3 acting as WWW users. Each user then entered the IP address of the other user in order to initiate the audio chat. All traffic was routed over PC 1.

In general, delay and packet loss are the key factors determining the QoS of real-time multimedia applications. Components comprising total end-to-end delay, as outlined in [7], include network delay (transmission, propagation, and queuing delay), operating system delay at the sender or receiver, hardware input/output delay, possible look ahead delay, and application delay (introduced by the receiver to compensate for jitter). Due to the fact that all of these components together affect user perceived quality, measurements were conducted to determine overall end-to-end delay. We define T1 as the moment when the sender (user 1) speaks, and T2 as the moment when that sound reaches user 2’s output device (speaker). We define end-to-end delay as T2-T1. T1 and T2 were determined by using

COOL EDIT 2000 software for recording audio.

Measurements of jitter were performed in conjunction with the RTP [16] definition of jitter that defines interarrival jitter as the mean deviation (smoothed absolute value) of the difference *D* in packet spacing at the receiver compared to the sender for a pair of packets.

2) *Results and Analysis*

Initial measurements were performed without the use of NIST Net to determine end-to-end delay and interarrival jitter. Measurements were repeated 20 times and an end-to-end delay of *532,35* $\pm$ *8,905 ms* and interarrival jitter of *3,466* $\pm$ *0,203 ms* were determined. Due to the fact that delay in our LAN is < 1 ms, we conclude that the measured end-to-end delay and jitter are a result of OS delay, hardware input/output delay, and application delay. Audio is captured from a live source (using a microphone) and then passed on for further processing (filtering, compression, format conversion). After passing through the network and reaching the receiver, data once again needs to be manipulated before being presented to the user. In our case, an ADPCM codec is used with dvi audio format. A playout buffer size was set to 250 (in ms) throughout all tests. This means that 250 ms of audio data is buffered prior to being passed on for further processing.

The next step was to use NIST Net to increase delay to see the effect that this would have on conversational quality as perceived by the user. Figure 6 shows how measured end-to-end delay increased with an increase in network delay. Values for jitter remained the same as in the initial test without NIST Net.

We found that up to 300 ms network delay in both directions was noticeable, but conversation was possible without greater difficulty. In our opinion, 400 ms network delay ($\approx$920 ms end-to-end) was the border of acceptable quality. Once again, buffer size was set to 250 ms.

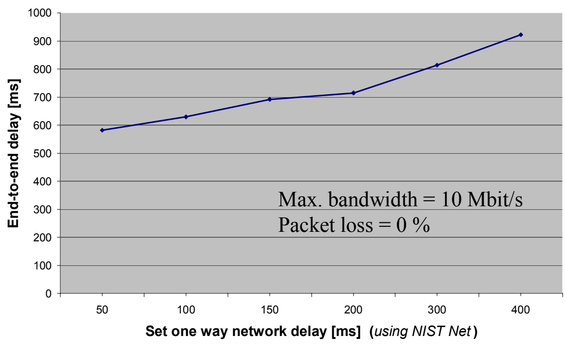

Figure 6. Measured end-to-end delay for VAC.

In order to test the effect of jitter, we set the NIST Net delay standard deviation parameter (*delsigma*) to values as shown in Figure 7 to achieve an increase in measured interarrival jitter.

Up until a set deviation of 15 (average measured jitter 11,48 ms) there was no degradation detected. At a set deviation of 20 (average measured jitter 14,3 ms) jitter became noticeable, but users had no problem understanding each other. A set deviation of 35 (average measured jitter 17,5 ms) resulted in increased degradation, but speech was still understandable. At a set deviation of 50 (average measured jitter 19,0 ms) jitter was highly noticeable and conversation required attention. At 75 (average measured jitter 20,0 ms) users needed considerable effort to understand each other, and quality was no longer acceptable. Therefore, we consider a value of 19,0 ms as the border of acceptable quality.

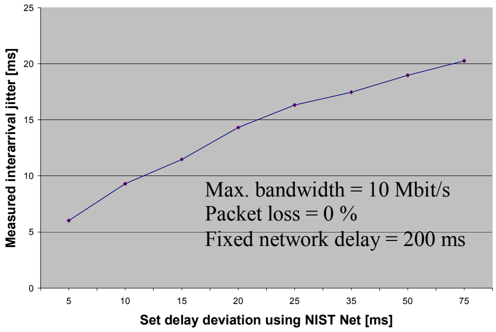

Figure 7. Measured interarrival jitter for VAC.

In order to determine the bandwidth limitations of our application, maximum bandwidth was limited to see at which point packet loss would start to occur (due to limited queue size at the router) and what kind of effect this would have on delay and user perceived quality. Queue size was set using NIST Net by defining minimum and maximum *derivative random drop* parameters (in number of packets). No packets are dropped if the queue length is under the specified minimum (50), and 95% dropped if queue length is greater than the specified maximum (80). Results showed that 4800 bytes/s was the minimum bandwidth required to achieve acceptable quality. At a bandwidth less than 4800 bytes/s, the router queue begins to fill up, causing great increases in delay and packet loss. This results in unacceptable speech quality and the inability to lead a normal conversation.

C. *Blaxxun Multi-User Community*

The application used for test purposes is a shared virtual environment built upon Blaxxun's Virtual Worlds Platform that allows multiple users (*community members*) to meet and interact. A client server architecture is used to distribute information among community members whenever they need updates automatically and in real time using the UDP protocol. This is necessary for cases such as chat, avatar motion, and shared events/objects. In multi-user interactive virtual environments, all changes resulting from a user's actions need to be made visible to other user's in a consistent manner in order to achieve "real-time" interactivity. Delay is therefore the key factor determining user perceived quality. Our tests involved measuring the end-to-end delay between two users, represented by 3D avatars, interacting in the shared community. The goal was to determine the maximum allowed delay in order to maintain acceptable user perceived quality of real-time interactivity. After initial download, network traffic corresponds to dynamic sending of updates depending on the degree of user interactivity.

1) *Test Procedure*

The Blaxxun Virtual Worlds Platform was installed on WS 1. Two users on PC 2 and PC 3 joined the community by downloading the world description from WS 1. All traffic was routed over PC 1.

End-to-end delay was measured from the moment when one user clicked on the keyboard to trigger a waving gesture until that gesture was made visible on the other user's screen (Figure 9). In the case of an avatar gesture, only packets carrying information necessary to trigger the accompanying action are sent. Due to such low bandwidth utilization, after initial download there was no need to test bandwidth limitations, rather delay proved to be the key factor influencing quality.

Measurements were conducted by using a camera (Video Blaster WebCam 3 USB) to record the moment when one user clicked to trigger a waving action and the moment when that action was made visible on the other user's screen. The recording frame rate was 30 frames/s. The software used for analyzing video frames was *SteadyHand 1.0* (http://steadyhand.dynapel.com).

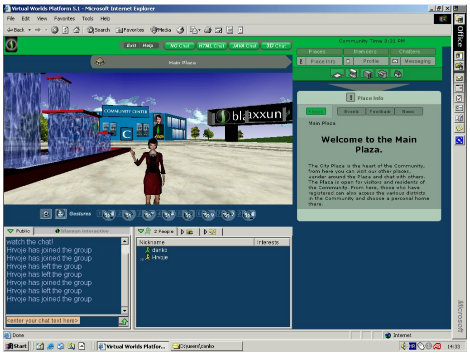

Figure 9. Avatar waving gesture in Blaxxun community

2) Results and Analysis

Initial measurements were performed without the use of NIST Net to determine end-to-end delay. Measurements were repeated 20 times and the end-to-end delay with a 95% confidence level was measured to be $201{,}15 \pm 19{,}721$ *ms*. Due to the fact that LAN delay is < 1 ms, this delay corresponds to processing delay at the server and rendering at the end user. It is also important to take into account measurement errors due to a maximum frame rate of 30 frames/s while recording (33,33 ms between frames).

The next step was to use NIST Net to increase network delay to see the effect that this would have on perceived quality. We found that delay up to 300 ms in each direction remained practically unnoticed by the users, who reported a feeling of real-time interactivity. At 400 ms delay in each direction, delay became more noticeable. Specific values of acceptable delay for interactive VR applications depend on the nature of the application. It is clear that in this particular scenario, involving only user communication through gestures, delay is more tolerable than in multi-user interactive games [18] or military simulations [15].

VI. DISCUSSION AND CONCLUSIONS

Achieved test results serve to provide some empirical data on the QoS requirements of different types of NVR services. In the case of downloading a single user VE from the Web, we are looking at a type of service comparable to classical web browsing. However, VE services designed for specific purposes such as a conversational virtual character pose stricter requirements on the underlying network. In the case of multi-user NVR services, requirements are characterized by unpredictable traffic flows due to dynamic user interactions. We can see this example in various network games, simulations, or CVEs.

Further work in this area will include additional tests involving a greater number of heterogeneous NVR services for the purpose of modeling such services with respect to standardized QoS classes as defined by 3GPP.

ACKNOWLEDGEMENTS

This research was partially funded by Ericsson Nikola Tesla under project R00101 "Networked Virtual Reality".

REFERENCES

[1] 3G TS 22.105: Services and Service Capabilities (Release 5), 2001-10.
[2] 3GPP TS 23.107: QoS Concept and Architecture (Release 5), 2002-01.
[3] NIST Net, http://snad.ncsl.nist.gov/itg/nistnet/
[4] ITU-T Recommendation F.700: A framework recommendation for audio-visual/multimedia services., Nov. 2000
[5] --, Ethereal protocol analyzer, http://www.ethereal.com
[6] K. Jeffay, T. Hudson, M. Parris. Beyond Audio and Video: Multimedia support for Distributed Immersive Virtual Environments. Euromicro, September 2001.
[7] W. Jiange, H. Schulzrinne. QoS Measurements of Internet Real-Time Multimedia Services. Technical Report CUCS-015-99, Columbia University, New York, Dec 1999.
[8] M.R. Macedonia, M.J. Zyda. A Taxonomy for Networked Virtual Environments. IEEE Multimedia, Vol. 4, No. 1, pp. 48-56, Jan.-Mar. 1997.
[9] M Matijasevic, I. Lovrek, D. Mikić, A Carić, D. Huljenić, Designing Bandwidth-Aware Virtual Reality Services for the New Generation Networks, Proc. 9th Int'l. Conf. on Telecom. Syst., Modeling and Analysis, pp. 84-89, Dallas, TX, 2001
[10] M. Matijasevic, D. Gracanin, K. P. Valavanis, I. Lovrek. Interconnection model for networked virtual reality applications. Proc. 24th LCN, pp. 52–61, Lowell, MA, 1999.
[11] M. Matijašević, L. Skorin-Kapov. Design and Evaluation of a Virtual Audio Chat. Accepted at TERENA Networking Conference, Limerick Ireland, June 2002.
[12] S. Oh, D. Kado, K. Fujikawa, T. Matsuura, S. Shimojo, M. Arikawa. QoS mapping for Networked Virtual Reality System. *Proc. SPIE Conf. on Performance and Control of Network Systems*, pages 18-26, Nov. 1997. Dallas, USA.
[13] K.S. Park, R.V. Kenyon. Effects of Network Characteristics on Human Performance in a Collaborative Virtual Environment. *Proc. IEEE VR `99*, Houston, TX, March 1999.
[14] J. Padhye, V. Firoiu, D.Towsley, J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. *Proceedings of the ACM SIGCOMM '98 conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 1998.
[15] J. Pullen, D. Wood. Networking Technology and DIS. *Proceedings of the IEEE*, 83(8), August 1995.
[16] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 1889, Internet Engineering Task Force, January 1996.
[17] S. Singhal, M. Zyda. Networked Virtual Environments: Design and Implementation, ACM Press Siggraph Series and Addison-Wesley, New York, NY, 1999.
[18] J. Smed, T. Kaukoranta, H. Hakonen. Aspects of Networking in Multiplayer Computer Games. *Proc. Int'l Conf. on Application And Development of Computer Games in the 21st Century*, 2001.
[19] K. Smid, I.S. Pandzic. A Conversational Virtual Character for the Web. Accepted at Computer Animation 2002.
[20] R. Steinmetz. Human Perception of Jitter and Media Synchronization. IEEE Journal on Selected Areas in Communications, Vol. 14, No. 1, pp. 61-72, 1996.