Invited Paper

Experiments in Best-Effort Multimedia Networking for a Distributed Virtual Environment

Tom Hudson *Michele Clark Weigle* *Kevin Jeffay* *Russell M. Taylor II*
University of North Carolina at Chapel Hill
Department of Computer Science,
Chapel Hill NC, 27599-3175

ABSTRACT

Until there is greater consensus on proposals for realizing better-than-best-effort services on the Internet, developers of multimedia and distributed virtual environment applications must rely on best-effort media adaptations to ameliorate the effects of network congestion. We present the results of a study on the use of adaptations originally developed for audio and video applications for the data-flows generated by the UNC nanoManipulator. The nanoManipulator is a virtual environment interface to a scanned-probe microscope that has been used by scientists as a tool for basic research in the material and biological sciences. We are building a distributed version of the system for operation over the Internet and are investigating media adaptations for realizing application performance requirements. The results of early experiments with audio and video-centric media adaptations applied to the flows generated by a microscope and a haptic force feedback device are promising. A simple forward error correction scheme provides good recovery from packet loss and an elastic display-queue management scheme limits the impact of delay-jitter and results in more continuous playout of media samples. These preliminary results provide evidence that a sophisticated virtual environment interface can operate over modest distances over the Internet to control a remote microscope in real-time.

Keywords: multimedia networking, distributed virtual environments, quality-of-service, best-effort networking.

1 INTRODUCTION

The 1990s saw the advent of multimedia computing and networking and the proliferation of research into the problems of network and operating system support for digital audio and video. Commodity processors and I/O subsystems were finally powerful enough to acquire, store, and process digitized streams of audio and video in real-time. Moreover, as LAN switching was deployed in campus and enterprise networks, and Internet backbones steadily increased in capacity, communication of these streams across the Internet became feasible.

Throughout much of this time a debate was waged as to how the Internet should evolve to support applications that required real-time transmission of continuously generated media streams. At a high-level, the central question was whether or not a network should provide end-to-end services with guarantees of quality-of-service. Given the tremendous increases and availability in network bandwidth, it was argued that one could exploit the flexibility inherent in the way most applications generated media to ameliorate the effects network congestion encountered in a network without guarantees of quality-of-service. On the one hand, the prevalence of streaming media applications on the Internet today is an existence proof that techniques for best-effort delivery of real-time media are capable of realizing the real-time transmission requirements for many applications. However, on the other hand, the correspondingly small number and limited use of more interactive applications such as audio and video conferencing, suggests that the lack of alternate service models on the Internet may be impeding the realization of the advanced communication, collaboration, and entertainment environments envisioned for the Internet.

Our interest lies in exploring the use of best-effort multimedia networking techniques for realizing the performance requirements of a distributed virtual environment for advanced microscopy. As was the case for audio and video ten years ago, we seek to understand the extent to which one can design an interactive virtual environment application for effective operation over the commodity Internet. In the abstract, our application is similar to a videoconferencing system: a set of input devices is used that generate long-lived periodic and quasi-periodic data flows with low latency tolerances and high fidelity requirements. Therefore it is natural to ask whether best-effort multimedia networking techniques developed for managing audio and video streams "scale" to meet the requirements of our application. By studying this question we hope to

In: *Multimedia Computing and Networking 2001*,
Proceedings, SPIE Proceedings Series, Volume 4312, San Jose, CA, January 2001, pages 88-98.

shed additional light on the fundamental nature of the requirements (if any) for quality-of-service on the Internet and inform the discussion of design principles for interactive, distributed virtual environments.

In this paper we present the results of some early experiments in adapting an existing virtual environment for operation over the Internet. The system, the UNC nanoManipulator, is a virtual reality interface to a scanned probe microscope (SPM) that allows chemists, biologists, and physicists to "see" the surface of a material sample such as a virus particle or a carbon nanotube at nanometer scale, and "feel" the properties of the surface through the use of force-feedback. This is accomplished by integrating an SPM with high-performance 3D graphics and a tracking/force-feedback haptic device. This system has been in use at UNC by local scientists for a number of years and has enabled new scientific investigations that were otherwise not possible to be conceived and performed. We are currently investigating the distribution of system components (*e.g.*, the microscope and the haptic display) across the Internet. This distribution enables remote operation of the system and the sharing of potentially expensive components between research groups. In addition, distribution admits the possibility of modifying the system to support collaborative experimentation between multiple non-co-located users.

As currently designed, the nanoManipulator assumes the presence of reliable, dedicated communication channels between system components. To operate over the Internet the system must be modified to deal with the common IP network pathologies of high delay, delay-jitter, packet loss, out-of-order arrivals, and duplicate arrivals. All but the first two problems can be solved by the use of a reliable, in-order transport protocol such as TCP, however, TCP does not allow an application to control either transmission delay or transmission rate and hence is less than desirable. We have developed an application-level transport protocol for the nanoManipulator system to deal with the problems of loss, delay and delay-jitter. Our approach is to use well-understood media adaptations developed for audio and video flows and apply these adaptations to the flows generated by the nanoManipulator. These include the data that is streamed from the microscope to the display processing engine and the tracking data of the user's current hand position that is transmitted from the haptic device to the microscope. Specifically, we investigate the use of forward error correction for ameliorating the effects of packet loss and an elastic display-queue management algorithm for smoothing delay-jitter. We have performed a number of experiments operating the modified nanoManipulator system in a controlled network environment where the contention for network resources can be closely controlled. We find that when combined with delay-amelioration techniques developed for other tele-operation and distributed virtual environment systems, we find that these best-effort data management and transmission techniques can be effective for realizing the real-time throughput and latency requirements of the nanoManipulator.

The remainder of this paper is organized as follows. Section 2 describes the nanoManipulator system is greater detail and presents its requirements for real-time media transmission. Section 3 reviews the multimedia networking problem for audio and video applications and reviews the relevant literature in best-effort multimedia networking. Section 4 discusses the modifications to the nanoManipulator system we made in order to enable its operation over the Internet and the specific media adaptations we developed. Section 5 describes our experimental setup and results. We conclude in Section 6 with a discussion of the efficacy of existing best-effort techniques for distributed virtual environments.

2 A DISTRIBTED VIRTUAL ENVIRONMENT: THE UNC NANOMANIPULATOR

Scanned-probe microscopes, such as the atomic-force microscope (AFM), allow the investigation and manipulation of surfaces down to the atomic scale. An AFM is capable of positioning a tip very precisely (x, y positions within a fraction of an atomic diameter) over a surface in a wide variety of environments, including ambient, ultrahigh vacuum, and under water. It is capable of resolving individual atoms on crystalline surfaces, and molecules, proteins and viruses under physiological conditions. The AFM tip can provide quantitative data on a wide range of sample features including the surface topography, friction, adhesion, temperature, compliance, *etc*. Most important, the surface can be modified through the deposition of material or through mechanical tip/sample interaction by machining the surface or manipulating surface-bound objects.

An SPM generates a three-dimensional dataset measuring the height of a surface (z) over a two-dimensional region (x and y). Traditional SPM interfaces only display this data in two dimensions — a rectangular area whose color at a point indicates the height of the surface at the corresponding point. However, three-dimensional display has long been used as a post-process to help scientists understand their data after they have completed the experiment. By displaying the data from the microscope three-dimensionally as the experiment is in progress, scientists have a better grasp of the experiment as it happens and are able to change their plans to suit their observations.

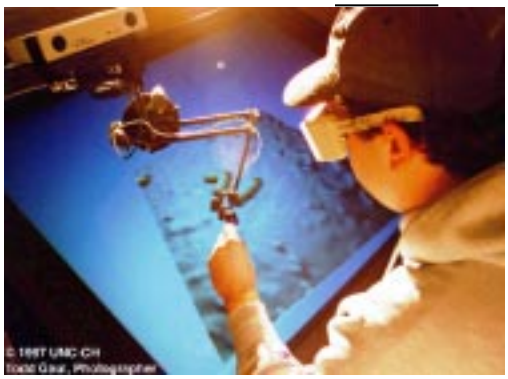

Figure 1: The nanoManiplator interface (showing stereo, head-tracked shutter glasses, and a haptic, force-feedback pointing device).

The nanoManipulator integrates three-dimensional graphics and force feedback to give a virtual environment interface to SPMs. Force feedback is used to convey information about the surface (*e.g.*, topography, compliance, *etc.*) to the user in a tactile form. A force feedback device measures the position and orientation of a stylus held by the user a thousand times per second. Given a geometric model of the surface, the system computes the force that should be felt by the user at the measured position, and uses motors to display that force to the user by mechanically pushing on (moving) the stylus. Force feedback devices are used to control remote manipulators such as robot arms handling dangerous materials or working in inaccessible locations. We use force feedback devices to control manipulators that are remote in scale instead of in distance. In our case the probe tip of an SPM is only a few microns across and can be felt and made to respond as if it were a pen in the user's hand.

The nanoManipulator gives the scientist virtual telepresence on the surface, scaled by a factor of a million to one, blowing a virus up to the size of an apple, or a strand of DNA up to the size of a piece of spaghetti. A stereo display (usually combined with head tracking) presents the 3D surface floating in space within arm's reach. The hand-tracking and force-feedback device allows the scientist to actually feel surface contours and to manipulate objects on the surface, such as the tobacco-mosaic virus (TMV) particles in Figure 1. This direct, natural interface to SPMs has enabled new forms of experimentation that augment and amplify human cognition, has revealed new results from existing data sets, and has allowed new and fruitful experiments that could not be performed otherwise. The nanoManipulator is a production system that has been used by chemists, biochemists, physicists and gene therapists, who at times flew across the country to use the system. They have investigated the mechanical properties of TMV and adenovirus (a vector for gene therapy), manipulated a 20nm colloidal gold particle into a thin gap formed in a wire, broken and repaired tiny nano-wires, and studied the mechanical and electrical properties of carbon nanotubes.

The nanoManipulator application has two basic modes of operation: scanning and point mode. Scanning is used to provide a single, consistent representation of the topography of the entire surface that is rendered with three-dimensional graphics and displayed to the user. In scanning mode, the microscope probe moves across the surface in a raster pattern at a constant speed, taking quick measurements of surface height and other datasets at regular intervals. Scanning is the standard mode in which SPMs operate when imaging a surface. The nanoManipulator augments the standard SPM by allowing manipulation of the surface. In point mode, the user's hand is coupled to the position of the microscope's tip through the haptic force feedback device. The nanoManipulator tracks the user's hand and tells the microscope to move its tip to a specific point corresponding to the user's current position on the virtual surface. When the tip arrives, it responds with measurements of the sample at that point. The nanoManipulator also has limited robotic capabilities. For example, the tip can be commanded to move along a path on the surface. While this command is executed, the control loop between the application and the tip does not need to be closed across the network.

The current standard deployment of the nanoManipulator couples a Windows NT workstation with modern 3D graphics card and a force-feedback controller (a SensAble Devices, Inc. PHANTOM) via a dedicated, switched Ethernet LAN. (More advanced versions of the system use a graphics supercomputer such as the UNC PixelFlow machine or high-end SGI.) The dedicated network is used by the graphics computer to send and receive SPM data and commands and position/force descriptions. The nanoManipulator application is implemented as three processes running on two NT workstations as illustrated in Figure 2. One machine is directly connected to the SPM and uses a vendor-supplied DSP card and software to

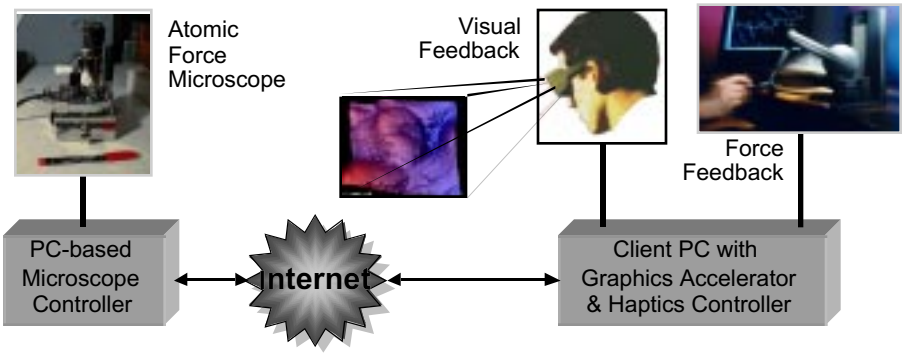

Figure 2: NanoManipulator high-level system structure.

interface with the SPM. The other machine is a two-processor machine with one CPU devoted to a hard-real-time process controlling the PHANTOM force-feedback device. The other CPU runs the main nanoManipulator application, including graphics, user interface, and network communications. Depending on the use of the system, other interface devices and scientific instruments may be attached, including the Magellan six degree of freedom input puck.

The real-time processing and communications requirements of the nanoManipulator system depend on the mode of operation of the system. When the system is in scan mode, the microscope controller sends data asynchronously to the graphics display as the tip rasters across the sample surface. This data updates each part of the displayed surface image as the scan data arrives. Because the raster involves a physical interaction, exact sampling rates vary widely with the instrument, the sample, and the sampling conditions (*e.g.*, whether the microscope is operating in air or under water), but typical numbers are 300 sixteen byte measurements per second. Each measurement is typically sent in a separate network packet generating a stream requiring approximately 38 kbps of bandwidth.

When the user is in direct control of the tip (point mode), there are no surface geometry updates (though images still need to be displayed from the user's current viewpoint). Instead, the microscope tip follows the trajectory of the user's hand as tracked by the force-feedback device. The surface is scanned at these locations and force feedback tells the user where the surface is and how it is changing. The nanoManipulator tracks the user's hand and tells the microscope to move its tip to a specific point corresponding to the user's current position on the virtual surface. When the tip arrives, it responds with measurements of the sample at that point. If the application is running at a frame rate of 30 Hz, it sends a 28-byte command to the microscope every frame, requiring 16 kbps of bandwidth. The microscope's response is 56 bytes. If the microscope can keep up with commands, it will generate a stream of responses requiring approximately 24 kbps of bandwidth.

In point mode, system latency becomes critical. Virtual reality applications have historically labored under as much as several hundred miliseconds of latency. Although this reduces their effectiveness, they may still be usable [9]. However, when the user is interacting with and controlling physical objects ("teleoperating") latency is much more critical. Teleoperation control loops generally become unstable with less than 100 ms of latency [1] and even at significantly lower levels of latency they may not be useful. Because of microscope and application latencies, there is very little room for network latency. A major focus of our work is reducing or ameliorating network latency so that the application can function in point mode over wide-area networks.

Force feedback systems consist of computer-controlled sensors and motors that sense the current position of the user, compute the force that should be exerted on the user at that position, and use the motors to display the force. Although video systems can run at 30 Hz to present the illusion of continuous motion to the sense of sight, to present the illusion of a continuous, stiff surface to the sense of touch requires 1,000 Hz update rates. With these high update rates, a pen-based system like the PHANTOM can provide a good simulation of a single point probing an arbitrary computer-generated environment.

To compute forces at these high update rates requires a very simple world model. The common solution is to have a complex world model in one process, which given the recent position of the force feedback device computes a simple local model and transmits that to a second real-time process that runs at the required high speed. For devices like the PHANTOM, the plane-and-probe model [6] works well. The coordinates of a plane tangent to the geometric model of the surface is

transmitted to the haptics controller process at least 20 times per second. This is sufficient to present a convincing image of the computer-generated world. Overall, the graphics process must maintain an image update rate of better than 20 frames/second to keep the illusion of immersion.

3 THE BEST-EFFORT MULTIMEDIA NETWORKING PROBLEM

Although the processing speeds in CPUs and raw bandwidth on network links is sufficient for many multimedia applications, the coordinated management of network and end-system resources to realize the processing and transmission requirements of applications with real-time constraints remains a significant challenge. Broadly speaking, research in this area has proceeded along two parallel fronts. Most visible is the work on new network service models for the Internet. The existing best-effort packet forwarding model has been proposed to be extended to include guarantees of quality-of-service including guarantees of throughput and delay. This so-called Integrated Services architecture for the Internet ("INTSERV") has proposed a "call-setup" procedure wherein resources are reserved along the path from receiver(s) to sender and network switches maintain per-call (per-connection) state to ensure they allocate link bandwidth in accordance with application requirements. Because of concerns about the complexity of the INTSERV architecture, a second, and more simple scheme, the Differentiated Services architecture ("DIFFSERV"), has become the dominant architecture of study. Under DIFSERV the emphasis is less on end-to-end services and more on per hop forwarding behaviors such as priority forwarding or low-delay forwarding.

The second major focus of research activity has been in the development of media adaptation mechanisms and techniques for best-effort multimedia networking. Here, the guiding principle is the observation that in the absence of "better-than-best-effort" delivery models for the Internet, applications requiring real-time transmission of media must sense the state of congestion in the network and adapt the media streams they inject into the network to maximize the probability of in-time delivery. The fundamental assumptions guiding this research are that (1) network bandwidth, when averaged over sufficiently long intervals, is sufficient for some minimally acceptable operation of the application, and (2) applications have sufficient flexibility inherent in the manner in which they generate media streams that they can adapt to the natural changes in network bandwidth availability.

While much significant research has been performed in both best-effort and quality-of-service networking, for multimedia delivery on the Internet today, the best-effort model dominates. A complex combination of technical and non-technical issues have conspired to impede the deployment of the results of the research into integrated or differentiated services and despite the tremendous activity in this area, the status quo is not likely to improve in the near future. Thus in order to enable substantive distribution of the nanoManipulator system, we are forced to investigate media adaptation schemes for nanoManipulator data flows.

In this study we are primarily concerned with ameliorating the problems of packet loss and delay-jitter. Packet loss is primarily caused by network congestion. When the offered load for a particular outbound link at a switch is greater than the capacity of the link, a queue forms. If the situation persists eventually the queue overflows and packets are lost. Packet loss can be dealt with through a combination of two techniques (used either separately or in combination). First, to the extent that a multimedia application is contributing to the state of congestion, the application can reduce its transmission rate in an attempt to reduce the offered load at the bottleneck switch and thus eliminate the congestion. For example, the (strongly) recommended rate adaptation is the multiplicative decrease scheme used in TCP. If all connections transiting the congested link perform rate adjustment (in particular, the same rate adjustment), the queue will drain and packets will cease to be dropped. Second, a multimedia application can ameliorate the effects of packet loss by performing some form of error control. The two dominant approaches here are to detect lost packets (typically by numbering packets at the sender and observing a gap in the sequence numbers at the receiver) and retransmit, or, introduce sufficient redundancy into the data stream to enable the receiver to reconstruct lost packets on the fly. In the latter case there are a number of techniques for introducing redundancy into a media stream. Simple schemes include replicating media samples and transmitting them multiple times with each retransmission separated in time to maximize the probability that at least one copy arrives at the receiver. More complex schemes generate lower resolution versions of the redundant media samples to reduce the overhead (bandwidth) of redundant transmissions. Alternatively, one can trade efficiency for error coverage by generating redundant samples that represent the *k*-way exclusive-or of the last *k* samples.

There is a clear trade-off between these approaches. The reactive, retransmission schemes increase the end-to-end transmission delay of media samples as it takes time to sense a lost packet, request a retransmission, and then receive the

retransmitted packet. In general this increase in delay is on the order of 1.5-2 network roundtrip times. The proactive redundancy-introduction schemes (commonly called *forward error correction* or FEC), only increase delay by the time required to reconstruct a lost sample (a design parameter), however, FEC schemes require an application to consume more bandwidth than it would otherwise. This is arguably a counterproductive measure in times of high congestion and contention for network resources. As described below, we adopt a FEC for the distributed version of the nanoManipulator.

The delay-jitter is also caused by the build-up of queues in the network. Display processes for periodically generated media such as audio and video require that samples arrive at the receiver at regular (precise) intervals in order to avoid "gaps" in the playout. When queues grow and shrink in the network the queuing delay experienced by packets changes accordingly. Thus, assuming no loss, when a network becomes congested, the average arrival rate of data at the receiver equals the transmission rate, but the instantaneous arrival rate can vary dramatically. The obvious solution to ameliorating the effects of jitter is to introduce a buffer at the receiver to smooth the arrival process. If the depth of the buffer is equal to the largest end-to-end delay likely to be encountered divided by the sample period, then the media playout will be gap-free if the buffer is filled prior to the initiation of the playout process. However, there again is a clear trade-off. On the one hand, a deep buffer guarantees gap-free playout, however, on the other hand it will have the effect of delaying each sample as if it actually encountered the worst case delay in the network. If the worst case delay is rare then the acquisition-to-display latency of the media may be unnecessarily high. A shallow (or no) buffer may result in a few gaps in the playout process but ensures that receiver-side latency is minimal.

Given that in general it is not possible to determine the worst case end-to-end delay a packet will encounter, multimedia applications commonly use a buffer management algorithm to dynamically set the depth of the buffer. These so-called "elastic" buffering schemes sense the level of congestion (queuing) in the network and set the depth of the playout buffer accordingly. For example, for the nanoManipulator, we adopt a receiver playout buffer management scheme called *queue monitoring*. Queue monitoring is based on the observation that over a given interval, if the occupancy of a receiver's buffer remains constant, then the instantaneous arrival rate is the same as the playout rate (independent of the actual number of media samples enqueued). This implies that with respect to the duration of a media sample, the level of delay-jitter is "0." If the occupancy of a receiver's buffer changes by ±1 elements over time, then the level of delay-jitter is equal to the duration of 1 media sample. More generally, if the occupancy of a receiver's buffer changes by ±*k* elements over time, then the level of delay-jitter is equal to the duration of *k* media samples.

A second key observation is that if the occupancy of a receiver's buffer changes by only ±*k* elements over time, then a buffer of depth *k* is sufficient to smooth the delay-jitter currently being experienced. In particular, in this situation there is no utility in maintaining a buffer of depth greater than *k* elements as doing so only increases the ultimate acquisition-to-display latency of samples. Combined, these two observations form the basis of a buffer management algorithm. One can record the variation in buffer occupancy over time and use this as a measure of the current level of delay-jitter. In addition, one can use this measure to dynamically set the depth of the buffer.

In queue monitoring we manage the buffer as a simple FIFO queue. A count and a time threshold are associated with each position in the queue. The count for position *k* in the queue represents the duration (measured in units of packet interarrival times) that the queue always contained at least *k* elements. When a packet arrives at the receiver and the playout queue currently contains *n* items, the counts for positions 1 through *n* are incremented. If the count for any queue position *k* between 1 and *n* exceeds its threshold, then the arriving packet is dropped. This indicates that the queue has contained at least *k* elements for a sufficiently long time that we believe the current level of jitter is less than *k* sample inter-arrival times. By dropping a packet we reduce the acquisition-to-display of media samples arriving in the future by reducing the time they will spend in the receiver's queue. However, by dropping a packet we also introduce a gap into the playout of the stream. Thus we are explicitly trading off gap-free playout for low latency playout. While this may seem like an unpleasant trade-off, for continuously generated media it is a fundamental trade-off. For media that is continuously generated and displayed at a constant rate, the only way latency can be reduced is to "skip" the display of a media sample.

Research into FEC and elastic buffering schemes for audio and video applications have shown that they can effectively ameliorate the effects of network congestion for many applications. We therefore use these techniques as our starting point for managing nanoManipulator flows on the Internet.

4 BEYOND AUDIO AND VIDEO: SUPPORTING A DISTRIBUTED NANOMANIPULATOR

The nanoManipulator is a powerful, useful microscope control system. The current challenge is to make the system more widely available. The system works well over a dedicated 10Mbps switched Ethernet between the two computers in the system. This configuration is appropriate to the existing testbed environment, where the microscope, 3D graphics system, and force-feedback device are collocated within reach of a dedicated LAN on the UNC campus. It is unreasonable to expect that high-performance computer graphics equipment and advanced scanned-probe microscopy equipment will be so located at other institutions. Therefore, to allow the wide availability of this interface, it will be necessary to tailor the application for operation over shared networks, including the Internet.

Haptic force feedback over a distance is difficult. The traditional approach to force feedback and teleoperation sends raw force and position data between the controller and the force feedback device at 500 to 1,000 Hz [3]. This is only possible over a tightly synchronous, dedicated, short-haul network or a bus. When run over a shared or long-distance network, transmitting raw data is highly sensitive to delay. Instead of presenting a solid, sharp-edged, stable surface, delayed force feedback results in soft, mushy surfaces, making the use of haptics ineffective or unstable [1].

The most common approach used by virtual reality applications to control force feedback without requiring high update rates is the plane approximation [6]. In the nanoManipulator's implementation of plane approximation, each new measurement from the microscope is combined with the previous measurement to determine a plane that approximates the shape of the surface near the two measurements. The user feels this plane until the next measurement arrives and a new plane is determined (see Figure 3). If the two planes are far apart, or the user's hand is "inside" the new plane, the force feedback device interpolates from the old plane to the new to avoid sharp discontinuities. Although the force feedback system is reading the user's position and changing the force it exerts at 1,000 Hz, the plane approximation only needs to be updated at 20 to 30 Hz to simulate stiff, sharp-edged surfaces.

Because the plane approximation interpolates between adjacent samples, it is moderately tolerant of sample loss caused by packet loss. It is still, however, highly sensitive to delay. As illustrated in Figure 4, with moderately high delay, a plane approximation displays incorrect and inconsistent surface topography to the user. Since the plane is a first-order approximation to the shape of the surface near the location of a measurement, if the user moves far from the point at which the measurement was taken, the approximation is invalid. Another problem in using the plane approximation with our AFM is that the plane approximation can be highly sensitive to noise in the microscope. Since the plane approximation is derived from the last two measurements reported by the microscope, if the user holds the tip still laterally but the height measured by the microscope varies noisily, the orientation of the plane will vary and the user will feel an unstable ("see-saw"), inaccurate surface.

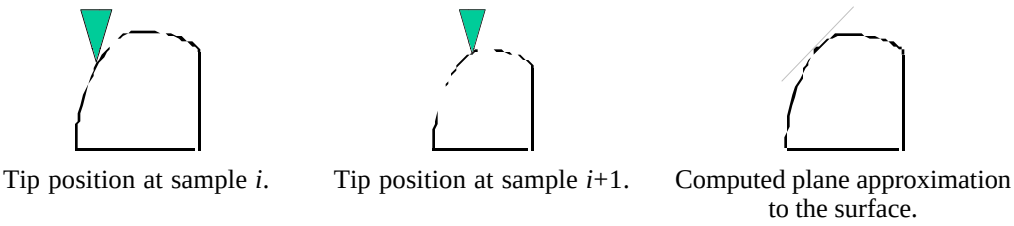

Figure 3: Local plane approximation example.

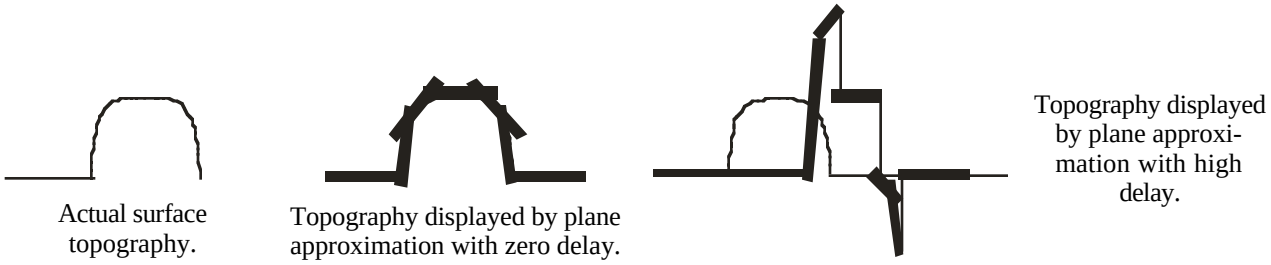

Figure 4: Local plane approximation in the face of high delay. (Motion of microscope tip is from left to right.)

We have performed multiple test deployments of a distributed nanoManipulator. For two of these we had dedicated circuits over Internet 2: one from North Carolina to Washington, DC, the other from North Carolina to Ohio. In both cases we had consistent one-way delays of 30-40 ms (as measured with ping) and no loss. Under these conditions the simple plane approximation was adequate. However, both dedicated circuits had very high administration and coordination costs to set up for the short term, and would have been prohibitively expensive to maintain over a long term. When we have run our application cross-country over the commodity Internet, we have seen a far more dismal picture of network performance, including high latency and high loss. Moreover, latency and jitter are higher in the presence of loss because of the application's use of TCP.

The nanoManipulator's initial use of TCP was motivated by a desire to reliably archive all data generated by the microscope during an experiment so as to have a complete scientific "lab notebook" of all experiments. The use of TCP made the archival process trivial and on the dedicated and campus LANs the additional added latency due to error-correction and congestion control was tolerable. For operation over the Internet we had to decouple logging from basic sample streaming and move the logging process to the microscope controller machine. This eliminated the requirement for reliable transmission of samples from the microscope to the display/haptics machine (since the force-feedback display process is moderately loss tolerant) and allowed us to use UDP.

However, in the event packet loss on the network is too high, we also experimented with a simple FEC scheme for the data flow from the microscope to the haptics/display. We sought to determine the extent to which we could effectively reduce application-level loss without significantly increasing stream latency. Since the microscope stream has a small base bandwidth, in the absolute the cost of FEC is small. Our FEC scheme is controlled by two parameters: a number of retransmissions n and a minimum inter-transmission interval t. We send each message a total of n+1 times. Each retransmission is separated from other packets carrying the same message by at least t milliseconds. An application-level header identifies the sample and hence the receiver not need know FEC parameters as it can detect and ignore any unneeded retransmissions. Increasing t increases mean latency and jitter but reduces the chance of correlated losses affecting every copy of a message. Some of the IP stacks we have used also seem to react poorly to our attempting to send too many packets too quickly, requiring us to insert extra inter-transmission time for the FEC packets.

Even with the latency reduced by switching from TCP to UDP, moderate levels of delay and delay-jitter can make the plane approximation useless. We are investigating alternate representations that have better delay tolerance. The first of these is the warped plane approximation, which uses a user-interface compensation to compensate for the delay. As illustrated in Figure 5, although the warped plane approximation tolerates delay much better than the simple plane approximation, it is sensitive to delay-jitter, which can distort the shape of the surface being felt. When operating the nanoManipulator over a LAN, delay-jitter is small and does not significantly disturb our application. Over longer Internet paths with more routers, delay-jitter increases significantly. To ameliorate the effects of delay-jitter, we adapted the Queue Monitoring buffer management algorithm for use in the nanoManipulator haptic display queue.

Queue Monitoring assumes that a media stream has a source and a sink that continuously produce and consume packets at equal rates. This allows it to abstract away or quantize time, dealing only with buffer occupancies at regular polling intervals. In our application, the two processes do not have equal rates: the user sends requests to the microscope at anywhere from 10 to 60 Hz, while the microscope's response rate varies widely because of device delays. We use an *updateable queue* [5] at the microscope to scale back the rate of requests received to a rate that the microscope can process. An updateable queue is a FIFO buffer that associates a key with every entry and drops any entry in the buffer with a key k before inserting a new entry with key k.

The user process needs to attempt to reconstruct the timing at which the microscope took samples. This requires extending Queue Monitoring to deal explicitly with sub-frame timing, making the implementation more complex and less

Actual surface topography.

Topography displayed by plane approximation with zero delay.

Topography displayed by plane approximation with high delay.

Figure 5: Warped plane approximation in the face of high delay-jitter. (Motion of microscope tip is from left to right.)

deterministic. Instead of playing out one sample from the queue every frame, the receiving process must explicitly track the mean interarrival time from the microscope and use that as the effective interframe time for queue playout. Tracking time like this, the efficacy of the queue monitor depends on the accuracy of our interarrival time computation, but so long as the computation is reasonably accurate we should be able to compensate for device jitter (variation in the time it takes the microscope to process a request) in addition to network jitter.

"Feel patch" is another surface representation that is both tolerant of delay and jitter and one that we are investigating as a delay-jitter adaptation. This higher-order approximation, related to the supervisory control approaches that dominate remote teleoperation (*e.g.*, [2]), is also less susceptible to device noise and can run adequately at a lower control rate (1-3 Hz). Instead of sampling the surface at the single point the user requested, the microscope takes samples at several nearby points and transmits those back to the client, which attempts to reconstruct the shape of the surface between them. This approach trades verity (output that is true to the underlying measurements) for verisimilitude (output that feels right), giving the physicists a less direct view of the surface; a tradeoff our users are not always willing to accept.

5 EXPERIMENTAL RESULTS

We conducted a number of experiments to evaluate our loss, delay, and delay-jitter adaptations. Our experiments were performed using the network infrastructure described in [4]. Briefly, we used a controlled laboratory network that models an enterprise or campus network having a single wide-area link to an upstream Internet service provider (ISP). We placed one endpoint of the nanoManipulator system on the "campus" side of the network and the other endpoint on the "ISP" side of the network. The distributed nanoManipulator system shares the ISP link with a large collection (thousands) of simulated Web browsers. Web browsers on the campus generate requests for Web servers out in the ISP's network which return responses that are sampled from an empirical model of Web pages. By appropriately configuring the number of Web browsers and servers we can closely control the load (*i.e.*, the level of congestion) on the campus-ISP link. NanoManipulator traffic competes with the web traffic on this link and hence is subject to varying (but predictable and reproducible) levels of congestion. The experiments reported below are the results of operating the nanoManipulator in one of two environments. In the first, the Web browsers and servers kept utilization of the campus-ISP link (a 10 Mbps link) at 90%. In the second environment the number of browsers and servers was increased to achieve an average link utilization of 98%.

In the first set of experiments, we measured the throughput of the nanoManipulator system in the face of congestion on the experimental network. Table 1 presents these results. The first experiment considered the performance of the original, unmodified system. This system performed no application-level adaptations and used TCP for transmitting all data-flows. Including headers, the force feedback stream consumed approximately 23 Kbps of bandwidth. In the second experiment we modified the system to use UDP instead of TCP. UDP bandwidth was only 20 Kbps, due to savings on headers. In the third and fourth experiments we augmented UDP to perform simple FEC. In these experiments samples were retransmitted in 15 *ms* intervals. All data reported in this section was obtained by placing a passive tap on the campus-ISP link and monitoring flows with *tcpdump*.

These results confirm the trade-off of latency for reliable delivery. With UDP we experience approximate 2% sample loss but display the stream with a mean latency and an average level of delay-jitter that is significantly lower than what we

Table 1: NanoManipulator throughput on the campus-link (link utilization = 90%).

Protocol	Mean Bandwidth (bps)	Application-Level Loss (per second)	Mean Latency (*ms*)	Jitter (*ms*)
TCP	23,474	0	145	124
UDP	20,229	0.34 (1.8%)	97	33
UDP + single FEC	39,509	0.04 (0.21%)	94	33
UDP + triple FEC	71,974	0.004 (0.02%)	95	33

observed with TCP. In the FEC experiments, single FEC doubled the bandwidth requirements but reduced the loss rate by an order of magnitude with no increase in latency and delay-jitter (over non-FEC UDP). Triple FEC doubled the bandwidth requirements again but produced another order of magnitude decrease in loss.

Although the use of UDP reduces total system latency from 140-150 *ms* to 90-100 *ms*, this is still significantly more delay than the 40-60 *ms* our science users are used to when working over the UNC campus LAN. The teleoperation literature leads us to expect reduced effectiveness and possible instability in the 90-100 *ms* region [1]. We are thus motivated to shift to a more delay-tolerant surface representation, such as the warped plane approximation, however, to do this we now need to address delay-jitter.

To study our ability to ameliorate delay-jitter we increased the load in our experimental network to 98% of the campus-ISP link's capacity. This produced significant jitter but also significant sample loss (approximately 10% under UDP). In this environment we evaluated the use of queue monitoring to smooth delay-jitter. Table 2 summarizes these results. We tested three queue monitoring settings and for each setting measured the loss rate (a measure of network congestion and a value that should remain constant between experiments as we are not addressing loss), the rate at which packets are dropped by the queue monitoring algorithm, the resulting gap-rate in the playout of this data in the application, and the end-to-end latency.

With no delay-jitter adaptation we measure a drop-rate of 12%, a gap-rate of 21.5%, and 90 *ms* of latency. Even with small parameter value settings for queue monitoring, the queue hardly ever empties. With a threshold of 30 packet arrivals for a queue of length 2 or greater (QM (30, 2)), we see a decrease in the drop-rate to 0.6%, and a gap rate just marginally above the loss-rate. (Since without error control lost samples always result in a gap in the playout, the loss-rate is always the lower bound for the gap-rate.) At QM (150, 2) and QM (3600, 2), the drop-rate continues to decrease, and the gap-rate falls to match the loss-rate. Although we measure a slight increase in latency with queue monitoring over the non-queue monitoring system, the increase quite tolerable and worth the decrease in the drop-rate (a subjective assessment).

Table 2: Impact of queue monitoring on delay-jitter smoothing (link utilization = 98%).

Protocol	Application-Level Loss (per second)	Drop-Rate (per second)	Gap-Rate (per second)	Latency (*ms*)
UDP	2.8 (9.7%)	3.4 (11.7%)	6.2 (21.5%)	89
UDP with QM (30, 2)	2.8 (10%)	0.18 (0.6%)	3.0 (10.6%)	94
UDP with QM (150, 2)	2.8 (9.7%)	0.06 (0.02%)	2.8 (9.7%)	96
UDP with QM (3600, 2)	2.8 (9.5%)	.003 (0.001%)	2.8 (9.5%)	91

6 SUMMARY & CONCLUSIONS

The UNC nanoManipulator is a state-of-the-art virtual environment interface for scanned-probe microscopes that is being used by scientists as a tool for basic research in the material and biological sciences. The success of the interface has motivated the construction of a distributed version of the system. However, operation of the interface over the Internet to control a remote microscope requires either modification to the system to accommodate the common IP network pathologies of high-delay, delay-jitter, and packet loss, or the provisioning of better-than-best-effort services on the Internet. Until there is greater consensus on proposals for realizing quality-of-service on the Internet, developers of multimedia and distributed virtual environment applications must rely on best-effort media adaptations to ameliorate the effects of network congestion. In this study we sought to investigate the use of adaptations originally developed for audio and video applications for the data-flows generated by the nanoManipulator. We have constructed a prototype distributed version of the system and are investigating media adaptations for realizing application performance requirements. We have presented the results of early experiments with audio and video-centric media adaptations applied to the flows generated by a microscope and a haptic force feedback device.

The results are promising. Very simple FEC schemes and a shallow delay-jitter buffer can individually ameliorate the effects of packet loss, high-delay, and high delay-jitter. Specifically, we observed good recovery from packet loss and more continuous playout of media samples. When combined with application specific adaptations such as the warped plane approximation we are optimistic that a distributed nanoManipulator system can be used for substantive scientific study. We conclude from these experiments that the use of existing media adaptations originally developed for audio and video transmission show promise for realizing the performance requirements of a distributed nanoManipulator system. These preliminary results provide evidence that a sophisticated virtual environment interface can operate over modest distance over the Internet to control a remote microscope in real-time.

7 ACKNOWLEDGEMENTS

Work supported by grants from the National Science Foundation (grants CDA-9624662, ITR-0082870, and ITR-0082866), NIH National Center for Research Resources Award RR02170 on Interactive Graphics for Molecular Studies and Microscopy. And grants from the Intel Corp., and the North Carolina Networking Initiative.

8 REFERENCES

[1] Anderson, Spong, "Bilateral Control of Teleoperators with Time Delay", *IEEE Transactions on Automatic Control* , v34, n5 (May 1989) pp. 494-501.

[2] Brunner, Landzettel, Steinmetz, Hirzinger, "Tele-Sensor Programming: a task-directed programming approach for sensor-based space robots," *Proceedings of International Conference on Advanced Robotics 1995*.

[3] Burdea, *Force and Touch Feedback for Virtual Reality*, John Wiley & Sons, New York, 1996.

[4] Christiansen, Jeffay, Ott, Smith, "Tuning RED for Web Traffic," Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August-September 2000, pp. 139-150.

[5] Kessler, Hodges, "A Network Communication Protocol for Distributed Virtual Environment Systems," *Proceedings of Virtual Reality Annual International Symposium '96* (March 1996) pp. 214-221.

[6] Mark, Randolph, Finch, Van Verth, Taylor, "Adding Force Feedback to Graphics Systems: Issues and Solutions", *Proceedings of Siggraph 96* (New Orleans, LA, August 4 - 9, 1996), pp. 447-452.

[7] Stone, Jeffay, "An Empirical Study of Delay Jitter Management Policies," *Multimedia Systems*, v2, n6 (January 1995) pp. 267-279.

[8] *http://www.cs.unc.edu/Research/nano/*.

[9] Ware, Balakrishnan, "Reaching for Objects in VR Displays: Lag and Frame Rate," *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 4, December 1994, pp. 331-356.