Congestion Control for Network-Aware Telehaptic Communication

VINEET GOKHALE, JAYAKRISHNAN NAIR, and SUBHASIS CHAUDHURI, Indian Institute of Technology Bombay

Telehaptic applications involve delay-sensitive multimedia communication between remote locations with distinct Quality of Service (QoS) requirements for different media components. These QoS constraints pose a variety of challenges, especially when the communication occurs over a shared network, with unknown and time-varying cross-traffic. In this work, we propose a transport layer congestion control protocol for telehaptic applications operating over shared networks, termed as *Dynamic Packetization Module* (DPM). DPM is a lossless, network-aware protocol that tunes the telehaptic packetization rate based on the level of congestion in the network. To monitor the network congestion, we devise a novel *network feedback module*, which communicates the end-to-end delays encountered by the telehaptic packets to the respective transmitters with negligible overhead. Via extensive simulations, we show that DPM meets the QoS requirements of telehaptic applications over a wide range of network cross-traffic conditions. We also report qualitative results of a real-time telepottery experiment with several human subjects, which reveal that DPM preserves the quality of telehaptic activity even under heavily congested network scenarios. Finally, we compare the performance of DPM with several previously proposed telehaptic communication protocols and demonstrate that DPM outperforms these protocols.

Categories and Subject Descriptors: C.2.2 [**Network Protocols**]: Applications

General Terms: Design, Algorithms, Experimentation

Additional Key Words and Phrases: Telehaptic communication, congestion control, transport layer, dynamic rate adaptation, QoS, multimedia

ACM Reference Format:
Vineet Gokhale, Jayakrishnan Nair, and Subhasis Chaudhuri. 2017. Congestion control for network-aware telehaptic communication. ACM Trans. Multimedia Comput. Commun. Appl. 13, 2, Article 17 (March 2017), 26 pages.
DOI: http://dx.doi.org/10.1145/3052821

1. INTRODUCTION

Telehaptic applications, such as telesurgery [Anderson and Spong 1989], involve long distance transfer of haptic-audio-visual information between distantly located users. The performance of a telehaptic activity is governed by a set of Quality of Service (QoS) parameters, specific to each type of media. According to Miras et al. [2002], Marshall et al. [2008], and Szigeti and Hattingh [2004], the maximum permissible one-way delay and jitter for the different media types are, respectively, as follows: video - 400 ms and 30 ms; audio - 150 ms and 30 ms; and haptic - 30 ms and 10 ms.

Nonconformance to the preceding constraints leads to degraded human perception, and can potentially compromise the quality of the telehaptic activity [Jay et al. 2007]. In particular, a haptic QoS violation results in destabilizing the haptic global control

Authors' addresses: Department of Electrical Engineering, Indian Institute of Technology Bombay, Mumbai, India 400076; emails: {vineet, jayakrishnan.nair, sc}@ee.iitb.ac.in.

DOI: http://dx.doi.org/10.1145/3052821

loop [Ferrell 1965; Anderson and Spong 1989], and a deteriorated perception of haptic objects. Hence, multimedia data reception and display within the prescribed QoS deadlines play a pivotal role in determining the stability and the overall performance of a telehaptic task.

In a shared network, like the Internet, a telehaptic source shares the network resources with other concurrent traffic streams. As a result, the intensity of the cross-traffic encountered by a telehaptic stream on a shared network is both unknown as well as time-varying. In such a scenario, the transmission of telehaptic data in a *network-oblivious* manner can be highly suboptimal. In particular, at times when the network is severely congested, a network-oblivious telehaptic stream may suffer large delays and frequent packet losses, leading to QoS violations. Note that this is all the more likely in resource constrained networks, such as wireless ad-hoc networks. On the other hand, at times when the network is lightly loaded, it may be feasible to transmit telehaptic data at its peak rate. The preceding discussion motivates the need for a *network-aware* telehaptic transmission scheme. In this article, we propose such a scheme, which monitors network conditions in real time, and adapts the telehaptic data rate accordingly to achieve congestion control in a lossless manner.

1.1. Contributions of the Article

In this article, we focus on point-to-point telehaptic communication over a shared network. Specifically, we propose a network-aware protocol for multiplexing and transmission of haptic, audio, and video data between two telehaptic nodes connected via a shared network. The protocol monitors network congestion in real time and (losslessly) adapts the transmission rate on the forward and the backward channels to maintain QoS compliance.

The proposed protocol receives haptic, audio, and video frames from the respective capture devices at each node and delivers these frames to the corresponding display devices at the other end. By design, our protocol is robust to the type and resolution of the media devices, as well as the audio/video encoding standard employed. Thus, our protocol may be viewed as a transport layer congestion control solution for point-to-point telehaptic communication, akin to the Transmission Control Protocol (TCP) for elastic internet traffic.

The proposed protocol has two main components:

(1) *Network feedback module:* The network feedback module (see Section 2.3) is a novel mechanism for real-time monitoring of end-to-end delays on the network. It exploits the bidirectional nature of telehaptic traffic to convey delays on each channel to the corresponding transmitter. Specifically, the end-to-end delays as measured by a receiving node are piggybacked on telehaptic data packets on the reverse channel (see Figure 2). This provides real-time feedback of the network state to the transmitting node with negligible overhead (3 bytes per packet). The proposed network feedback module can also potentially be utilized for other (bidirectional) media streaming applications like video conferencing.
(2) *Dynamic packetization module:* The Dynamic Packetization Module (DPM) (see Section 2.4) is a lossless mechanism for telehaptic data rate adaptation, based on the delay feedback from the network feedback module. DPM is motivated by the following observation: Under telehaptic data transmission at the default packetization rate of 1,000 packets/sec, the overhead due to packet headers from various layers accounts for almost half the total telehaptic traffic. Thus, when the network is congested, DPM dynamically merges successive telehaptic fragments into a single packet, thereby lowering the overall transmission rate to match the available

network capacity. Naturally, this transmission rate reduction is achieved at the expense of additional packetization delay at the transmitter.

We evaluate the proposed telehaptic transmission scheme via extensive simulations as well as human subjective tests through a real-time telepottery experiment (see Section 4). Our simulations reveal that DPM meets the telehaptic QoS specifications even under extremely congested network settings. Our subjective tests confirm that DPM provides a seamless telehaptic user experience in a congested network. We also compare DPM with other recently proposed telehaptic communication protocols, and demonstrate that DPM outperforms these protocols with respect to QoS compliance.

Finally, to evaluate the performance of the proposed scheme analytically, we derive bounds for the maximum haptic, audio, and video delays on a network with a single bottleneck link, assuming Constant Bit Rate (CBR) cross-traffic (see Appendices B and C). These delay characterizations are useful in identifying network settings where QoS-compliant telehaptic communication is feasible.

1.2. Related Work

There have been several attempts to address the problem of large telehaptic bandwidth requirement. The standard input and output update rate of the haptic signal is 1 kHz. In order to reduce the packetization delay encountered by the haptic samples, the conventional approach follows fixed haptic packetization at 1 kHz (1 packet per sample) for transmission over the network. This approach is highly bandwidth demanding, and is not friendly to other network users. To counter this issue, the works in Hinterseer et al. [2005], Clarke et al. [2006], Hinterseer et al. [2008], Dabeer and Chaudhuri [2011], and Sakr et al. [2011] explored *adaptive sampling*, which exploits the perceptual limitation of the human haptic system to achieve lossy haptic signal compression. A *Just Noticeable Difference* (JND) metric adaptively marks the haptic samples that are not perceivable by the human users. The communication system refrains from transmitting such samples, thereby reducing the telehaptic data rate. The missing haptic samples are then reconstructed at the receiver using standard extrapolation techniques listed in Gokhale et al. [2013]. However, critical operations, like telesurgery, necessitate accurate replication of the surgeon's hand movements. In such scenarios, a minor loss of precision due to adaptive sampling could result in potentially irreparable damage. Also, a teleoperator, such as a robotic device, could practically sense all haptic samples; in such cases, adaptive sampling discards perceptually significant samples. Another networking-related issue with adaptive sampling is the following. The *instantaneous* source rate of adaptive sampling depends purely on the speed of haptic interaction, and can at times far exceed the *average* source rate. As a result, provisioning the network for the average source rate can lead to serious QoS violations; this is demonstrated in Section 4.3. In other words, adaptive sampling does not provide any real economies with respect to network bandwidth requirement - one needs to provision network capacity for the *peak* telehaptic data rate in order to avoid QoS violations.

Several application layer protocols have been specifically designed for telehaptic communication. ALPHAN: Application Layer Protocol for HAptic Networking, proposed in Al Osman et al. [2007], implements haptic and graphic data communication at the packetization rate of 1 kHz. AdMux: Adaptive Multiplexer [Eid et al. 2011] proposes a statistical multiplexing scheme for scheduling haptic-audio-video packet transmission based on the QoS requirements and changing network behavior. Haptics over Internet Protocol (HoIP) for point-to-point communication, proposed in Gokhale et al. [2015], addresses media multiplexing and telehaptic communication involving haptics, audio, and video data. The previously mentioned protocols carry out telehaptic transmission at the peak rate, and hence do not address the problem of congestion control. In Cizmeci

et al. [2014], the authors consider visual-haptic multiplexing over CBR communication links, employing adaptive sampling for haptic signal compression. However, the drawbacks of adaptive sampling mentioned previously apply here (see Section 4.3 for a demonstration).

The work in Fujimoto and Ishibashi [2005] explores the possibility of merging multiple haptic samples in a packet to reduce the telehaptic data rate. In contrast with the scheme proposed in this article, the scheme in Fujimoto and Ishibashi [2005] always combines a fixed number of haptic samples, irrespective of the network conditions. Note that this implies unnecessary packetization delay even when the network is uncongested. Moreover, the authors showed in a particular setting that a packetization interval of 8 ms results in a satisfactory user performance. On the contrary, we demonstrate (see Figure 4) that the packetization intervals greater than 4 ms result primarily in increasing end-to-end delays, without any substantial reduction in the telehaptic data rate.

Note that the previously mentioned proposals are all *network-oblivious*, that is, they do not adapt the telehaptic transmission rate based on network conditions. The literature provides a few works that have considered *network-aware* telehaptic rate adaptation. We discuss these next.

In Lee and Kim [2007], the authors propose a network adaptation scheme for merging haptic samples based on packet losses arising out of congestion. Such a scheme is reactive to network congestion, in the sense that data rate reduction is activated only after detecting persistent packet losses. Clearly, such a loss-based congestion control mechanism is not suitable for highly delay-sensitive telehaptic applications. We note that Lee and Kim [2007] does not provide much detail about the rate adaptation mechanism itself; also, the effects of this rate adaptation on other concurrent network flows are not analyzed.

The authors in Wirz et al. [2008] propose the first delay-sensitive haptic communication protocol named Efficient Transport Protocol (ETP). ETP detects congestion based on Round-Trip-Time (RTT) measurements. Once congestion is detected, ETP reduces the telehaptic data rate by increasing the interpacket gap, that is, by *downsampling* the haptic signal. In contrast, the protocol proposed in this article preserves the fidelity of the haptic signal, adapting instead the packetization rate based on the congestion level in the network.

The paper most closely related to ours is Kokkonis et al. [2015], which proposes NAFCAH: Network Adaptive Flow Control Algorithm for Haptic data. Like DPM, NAFCAH adapts the number of haptic samples to be merged into a packet on the forward channel based on network conditions. However, there are two crucial differences between NAFCAH and DPM. First, when congestion is detected, NAFCAH decreases its transmission rate in stages. In contrast, DPM responds to congestion with an aggressive rate reduction, which enables network buffers to get flushed quickly, minimizing the possibility of QoS violations. Second, NAFCAH monitors congestion based on RTT measurements. However, under asymmetric network conditions, RTT may not provide an accurate estimate of the (one-way) delay on the forward channel. In contrast, DPM estimates the delay on the forward and the backward channels separately. The performance implications of the preceding differences are demonstrated in Section 4.3.

The authors of Gokhale et al. [2016] propose a network-aware opportunistic adaptive haptic sampling mechanism, wherein the adaptive sampling threshold is varied based on the congestion level in the network. Note that the limitations of adaptive sampling discussed earlier apply to this work as well.

Finally, we contrast our work with the Real-time Transport Protocol (RTP) [Schulzrinne et al. 2003], which is the most commonly used protocol for audio/video streaming and has also been recommended by some researchers for telehaptic

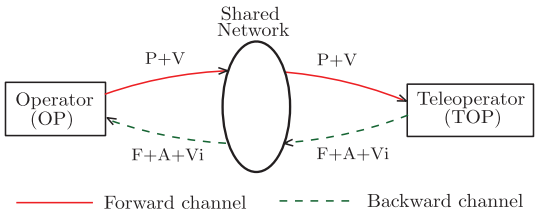


Fig. 1. A diagrammatic representation of master-slave based telehaptic setting showing the telehaptic dataflow in a shared network. Dataflow notations: P - position, V - velocity, F - force, A - audio, Vi - video.

communication (see, e.g., Steinbach et al. [2012]). RTP uses report-based notification for monitoring the network conditions at regular intervals of time. The multimedia receiving agent sends RTP Control Protocol (RTCP) receiver reports to the transmitters, for QoS monitoring, once in every 500 ms [Tos and Ayav 2011]. However, as we demonstrate in Section 4.3, such sparse feedback is insufficient for telehaptic applications, which are sensitive to network changes that occur over a time scale of tens of milliseconds.

1.3. Organization of the Article

The article is organized as follows. In Section 2, we describe the configuration of a typical telehaptic environment, and explain in detail the design and working of the proposed telehaptic communication framework. In Section 3, we discuss the setup for simulations and the real-time telepottery experiment. Section 4 presents the findings of the experiments, and in Section 5, we state our conclusions. Finally, in the Appendix, we describe the DPM header structure, and characterize the maximum end-to-end haptic/audio/video delay under the proposed scheme in a simple (single bottleneck) network setting.

2. DESIGN OF TELEHAPTIC COMMUNICATION FRAMEWORK

In this section, we explain the standard telehaptic setting on a shared network, and describe the techniques proposed in this article for a lossless, network-aware, adaptive telehaptic data communication.

2.1. Typical Telehaptic Environment

We consider a typical point-to-point telehaptic application, like telesurgery, running on a shared network as shown in Figure 1. The operator (OP) acts as the master and sends the current position and velocity commands to the teleoperator (TOP). In response, the TOP, acting as the slave, transmits force information to the OP, in addition to auditory and visual data. The channels on which the operator and teleoperator transmit telehaptic data are called forward and backward channels, respectively. Note that the telehaptic traffic is inherently bidirectional and asymmetric in nature. Moreover, the forward and backward channels are also asymmetric; in general, they may differ with respect to routing paths, capacity, as well as cross-traffic. Our network feedback module, described in Section 2.3, estimates congestion on the forward and the backward channels separately. Finally, we remark that the particular master-slave setup depicted in Figure 1 is assumed only for concreteness in exposition. Our proposed telehaptic communication protocol works in any general point-to-point telehaptic application.

2.2. Media Multiplexing Framework

In this section, we describe our media multiplexing framework. Multiplexing the media frames appropriately from the different capturing devices and forwarding them to the

transmitter is a critical task in any network based real-time interactive application, since it directly influences the QoS adherence of the respective media. The authors in Cizmeci et al. [2014] rightly explain the importance of splitting a large video frame into smaller parts for transmission. Naturally, if a large video frame is transmitted in a single packet, it would clog the network for a long time, thereby delaying the subsequent haptic/audio samples substantially. The media multiplexing framework proposed here is an adaptation of that in Cizmeci et al. [2014] and Gokhale et al. [2015].

Our media multiplexer works in synchronization with the sampling of the haptic signal, which we assume occurs at the default rate of 1 kHz. Each time a haptic sample is generated, our multiplexer generates a telehaptic fragment of size s_f, which contains the latest haptic sample, as well as audio/video data as explained in the following.[1]

Let f_a and f_v denote the peak frame generation rate (in frames per second) for audio and video, respectively. Let s_a and s_v denote the maximum size (in bytes) of an audio and video frame, respectively.[2] The (peak) telehaptic payload generation rate, denoted by D (in kbps), is expressed in terms of the individual media parameters as

$$D = (f_h \cdot s_h + f_a \cdot s_a + f_v \cdot s_v) \cdot (8/1000). \tag{1}$$

Here, f_h denotes the haptic sampling rate (assumed to be 1 kHz), and s_h denotes the size of a haptic sample.[3]

In order to maintain equilibrium between the payload generation and the multiplexing, the size s_f (in bytes) of the telehaptic fragment is given by

$$s_f = \frac{D}{8 f_h}. \tag{2}$$

Due to the mandatory haptic sample in each telehaptic fragment, the size of audio/video data in a fragment is given by

$$s_m = s_f - s_h. \tag{3}$$

Since audio has a stricter QoS constraint than video, our multiplexer gives audio data priority over video data. That is, in each telehaptic fragment, the multiplexer packs s_m bytes of audio/video data (not previously multiplexed), giving strict priority to audio over video. It can be shown that the proposed hierarchical priority-based multiplexing mechanism leads to substantially lower audio/video jitter compared to the first-come-first-serve multiplexing mechanism proposed in Gokhale et al. [2015].

2.3. Network Feedback Module

The network feedback module performs two functions: (i) it monitors the delays on the forward and backward channels separately through *in-header delay notification* mechanism, and (ii) based on the received piggybacked delays it generates triggers for the respective transmitters to adapt their data rates. We explain these functions in the following.

2.3.1. In-header Delay Notification. We exploit the bidirectional nature of the telehaptic traffic to convey end-to-end delays on each channel to the respective transmitter, without transmitting specialized reports (unlike RTP). The in-header delay notification mechanism inserts the end-to-end delay encountered by the latest received packet into

[1]If there is no audio/video data, as is the case in the communication from the OP to the TOP (see Figure 1), then each telehaptic fragment is composed of a single haptic sample.

[2] f_a, f_v, s_a, and s_v, depend on the audio/video encoding standards employed. It is important to note that the proposed protocol, which operates at the transport layer, is robust to the encoding standards used.

[3]Throughout this article, we use the terms sample and frame interchangeably.

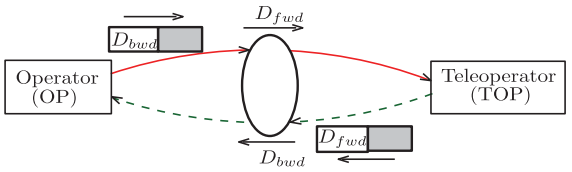


Fig. 2. A schematic representation of the in-header delay notification mechanism. D_{fwd} and D_{bwd} indicate the end-to-end delays on the forward and backward channels, respectively.

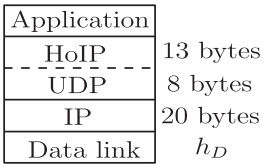


Fig. 3. Representation of the network protocol stack model, along with the corresponding header size at each layer.

the header of the packet to be transmitted, as shown in Figure 2. In particular, the headers of packets transmitted on the forward channel include the end-to-end delay experienced by the last received packet on the backward channel, and vice versa. This mechanism enables real-time monitoring of the state of congestion on each channel separately, with a negligible overhead of 3 bytes per packet.

The telehaptic nodes are time synchronized using Network Time Protocol (NTP) [Mills 1991]. The end-to-end delay encountered by a telehaptic packet received is thus calculated as the difference between the time of reception and the timestamp of the earliest haptic sample embedded in the received packet. Note that merging of multiple telehaptic fragments in a packet is explained in detail in Section 2.4.

The in-header delay notification mechanism is more effective than the report-based notification of RTP for three major reasons. Firstly, the higher rate of delay notifications provides finer details of network changes. This enables the telehaptic nodes to swiftly adapt the telehaptic rate to the changing network conditions. Secondly, our scheme does not transmit specialized packets to convey delay feedback, and thus induces a smaller overhead compared to RTP. Thirdly, the in-header delay notification mechanism estimates the delays on the forward and the backward channels separately, enabling each transmitter to adapt its rate based on the state of the corresponding channel.

2.3.2. Generation of Rate-Adaptation Triggers. Based on the trend observed in the measured delays on each channel, the network feedback module generates two triggers for the corresponding transmitter. The trigger I_C signals that the channel is getting congested; this causes the DPM module to reduce the telehaptic data rate if possible (see Section 2.4). The trigger I_S signals that the channel delays are steady; this causes the DPM module to probe if the channel has spare capacity by increasing the telehaptic data rate if possible (see Section 2.4).

In order to trace the delay pattern, we use an exponentially weighted moving average filter defined by

$$d_{avg}(n) = \alpha * d(n) + (1 - \alpha) * d_{avg}(n - 1), \tag{4}$$

where $0 < \alpha < 1$. Here, $d(n)$ denotes the n^{th} end-to-end delay measurement.[4]

The network feedback module generates triggers as follows. The trigger I_C is generated on observing N continuous increasing measurements in $d_{avg}(\cdot)$. Note that a steady increase in the end-to-end delays indicates that queues in the network are building up

[4]Note that the OP (TOP) may receive the same delay measurement multiple times; this can happen if the TOP (OP) makes multiple packet transmissions between successive receptions. To avoid the same delay measurement from resulting in multiple updates in Equation (4), we include a one-bit field named *delay indicator* (field D of Figure 20 in Appendix A) to the packet header. This field is set to 1 in the case of a repetitive transmission of a previously computed delay, and 0 in the case of the transmission of a newly computed delay.

due to congestion. The trigger I_S is generated if the most recent N entries in $d_{avg}(\cdot)$ satisfy two conditions: (i) the entries exhibit neither an increasing nor a decreasing trend, and (ii) the latter $N-1$ entries are within a specified tolerance interval (say around 10%) of the first. Note that generation of the trigger I_S signals that network conditions are steady. It is worth mentioning that since the generation of triggers is based on a *trend* of the end-to-end delays, the proposed rate adaptation scheme remains robust to time synchronization errors of NTP. In our experiments, reported in Section 4, we set $\alpha = 0.2$, as recommended in Montgomery [2007], and $N = 8$.

2.4. Dynamic Packetization Module

In this section, we describe the DPM, which adapts the telehaptic data rate based on the triggers generated by the network feedback module. We begin by presenting some calculations that illustrate the extent of telehaptic data rate variation possible by varying the packetization rate.

Assuming Ethernet on the data link layer, the overall overhead per packet due to the link layer ($h_D = 26$ bytes), IP, and UDP headers equals 54 bytes (see Figure 3). Adding to this our protocol's overhead of 13 bytes (see Appendix A), we arrive at a net overhead of 67 bytes/packet. If we transmit each telehaptic fragment as a separate packet (this corresponds to a 1 kHz packetization rate), this amounts to an overall overhead rate of $R_{oh} = 536$ kbps. For a standard TOP payload rate of 560 kbps (haptic - 96 kbps, audio - 64 kbps, video - 400 kbps), the overhead constitutes a substantial proportion (48.9%) of the telehaptic traffic.[5]

Now, suppose that we merge k consecutive telehaptic fragments into a single packet for transmission. We refer to this scheme as the *k-merge* packetization scheme, and we refer to the special case $k = 1$ as the *no-merge* packetization scheme. The telehaptic data rate R_k corresponding to the k-merge packetization scheme (in kbps) is given by

$$R_k = D + \frac{R_{oh}}{k}, \tag{5}$$

where D is the telehaptic payload generation rate given by Equation (1) and R_{oh} denotes the overhead rate under the no-merge scheme. Taking $R_{oh} = 536$ kbps, Figure 4 presents the variation of telehaptic overhead rates and packetization delay for different k-merge schemes. Note that these packetization delays correspond to the earliest haptic sample in the packet. Assuming $D = 560$ kbps, we see that on the backward channel the telehaptic transmission rate for the no-merge scheme equals 1,096 kbps, whereas the transmission rate for the 4-merge scheme equals 694 kbps. We observe that there is a substantial scope for losslessly varying the telehaptic transmission rate by controlling the packetization parameter k. Of course, the data rate reduction from increasing k comes at the cost of a higher packetization delay at the source.

The idea behind DPM is to dynamically adapt the packetization parameter k depending on the network conditions. In other words, DPM dynamically switches between different k-merge schemes based on the triggers from the network feedback module. From Figure 4, we note that the overhead reduction becomes insignificant for large values of k, whereas the packetization delay grows linearly in k. Thus, DPM confines the adaptation of k to the range $1 \leq k \leq k_{max}$. In this work, we set $k_{max} = 4$.

DPM is a *step increase, multistep decrease* (SIMD) algorithm. This is a variation of the classical *additive increase, multiplicative decrease* (AIMD) congestion control mechanism of TCP [Chiu and Jain 1989]. Specifically, on receiving the trigger I_C (recall that this trigger signals that the network is getting congested), DPM sets $k = k_{max}$.

[5]The overhead represents an even higher proportion (72.09%) of the telehaptic traffic from the OP to the TOP, since the payload is composed of only haptic data.

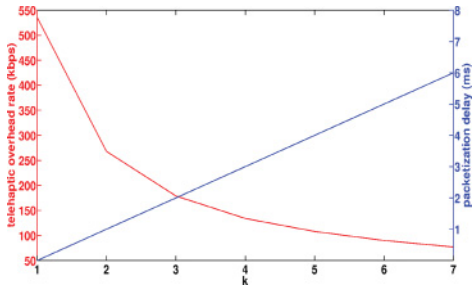


Fig. 4. Telehaptic overhead rate variation for different k-merge packetization schemes, along with the corresponding packetization delay.

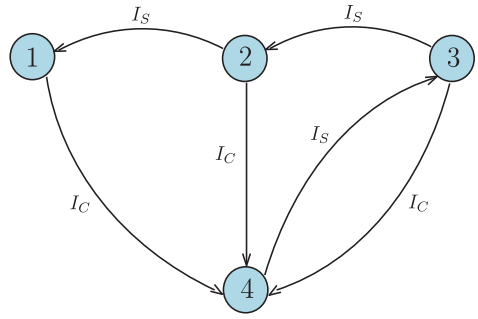


Fig. 5. A state transition diagram representation of the step increase, multistep decrease approach of DPM with $k_{max} = 4$.

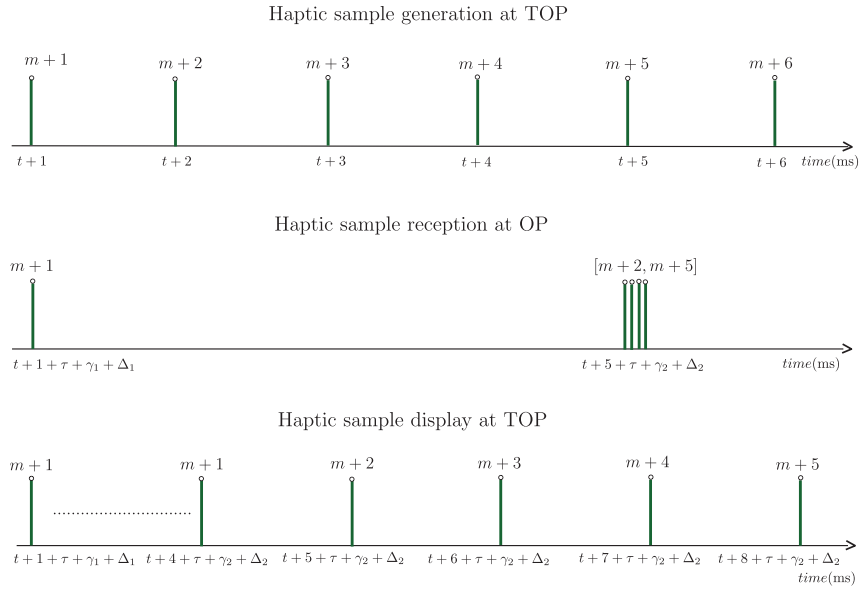


Fig. 6. Timing diagram illustrating haptic sample transmission at TOP, reception, and display at OP using zero-order hold strategy. The samples bunched together indicate simultaneous reception due to the 4-merge packet.

Thus, on sensing congestion in the network, DPM decreases the telehaptic data rate aggressively in order to decongest the network in the shortest possible time. On the other hand, on receiving the trigger I_S (recall that this trigger signals that network delays are steady), DPM decreases k by 1 if $k > 1$. Thus, on sensing that the network is in a steady state, DPM probes if a higher data rate is achievable by decreasing k by one unit. Figure 5 shows a state transition diagram representation of DPM.

Note that DPM's dynamic packet rate adaptation will induce additional jitter in the receiver. To get a sense of the jitter caused by DPM, we perform the following simple analysis, focusing only on haptic jitter (note that the haptic stream has the tightest jitter constraint). It is easy to see that the maximum jitter occurs when switching from $k = 1$ to $k = k_{max} = 4$. Consider the sequence of haptic samples shown in Figure 6. Suppose that initially, $k = 1$. Note that sample number $m+1$ is generated at time $t+1$,

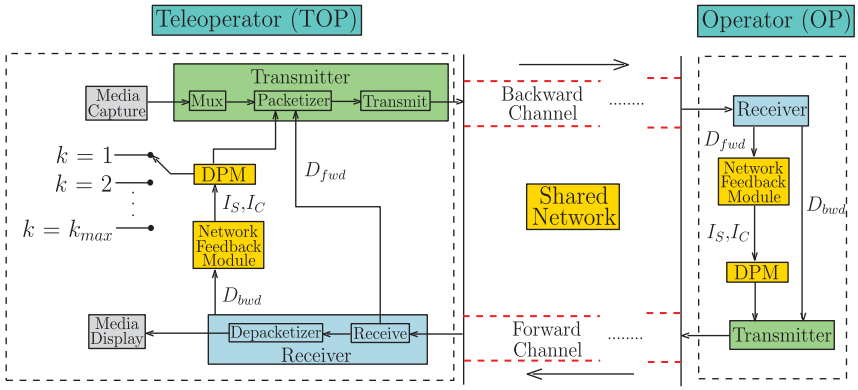


Fig. 7. A block diagram showing the architecture of the proposed telehaptic communication framework. The design at the OP is similar to that of the TOP, and is not shown for brevity.

and is received and displayed at time $t + 1 + \tau + \gamma_1 + \Delta_1$. Here, τ denotes the one-way propagation delay, and Δ_1 and γ_1 denote the queueing delay and transmission delay of the packet containing sample number $m + 1$, respectively. Now, suppose that starting from sample number $m + 2$, we switch from $k = 1$ to $k = 4$. In this case, sample $m + 2$, which is generated at time $t + 2$, will only get transmitted at time $t + 5$ (along with the next three samples), and will get received and displayed at time $t + 5 + \tau + \gamma_2 + \Delta_2$. Here, Δ_2 and γ_2 denote the queueing and transmission delays, respectively, experienced by the packet containing sample $m + 2$. Thus, the jitter of the haptic sample $m + 2$ equals the difference between its actual display time and its expected display time: $(t + 5 + \tau + \gamma_2 + \Delta_2) - (t + 2 + \tau + \gamma_1 + \Delta_1) = 3 + (\Delta_2 - \Delta_1) + (\gamma_2 - \gamma_1)$. Note that $\gamma_2 \leq 4\gamma_1$. Assuming then that Δ_1 and Δ_2 are comparable, we can bound the jitter by $3(1 + \gamma_1)$. Thus, we see that by restricting k to be at most 4 under DPM, we introduce an additional jitter of at most $3(1 + \gamma_1)$ on the haptic stream. Note that the subsequent 4-merge packet (carrying haptic samples $[m + 6, m + 9]$) arrives at $t + 9 + \tau + \gamma_2 + \Delta_2$. We validate the correctness of the jitter analysis through simulation results reported in Table I.

Overview of Protocol Architecture: Figure 7 presents an overview of the proposed telehaptic communication framework. We explain the working with respect to the TOP, whereas similar operations are carried out at the OP as well. On receiving the telehaptic packet at the TOP, the *depacketizer* module decodes the header information. Based on the header contents, the payload is forwarded to the appropriate media display devices. The backward channel delay (D_{bwd}) in the header is supplied to the *network feedback module* for learning the recent changes in the backward channel. Based on the delay analysis, the network feedback module generates triggers (I_S, I_C) appropriately. On arrival of a trigger, the DPM selects k, which is communicated to the *packetizer* for composing the telehaptic packets. The TOP also calculates the end-to-end delay on the forward channel (D_{fwd}) after every packet reception, which is sent to the *packetizer* for inclusion in the packet header that is transmitted to the OP.

3. EXPERIMENTAL DESIGN

In this section, we describe the setup used in our experiments to assess the performance of the telehaptic data transmission scheme proposed in this article. The objective of the experiments is to investigate the ability of DPM to perform congestion control under heavy cross-traffic scenarios. The performance metrics we consider are QoS adherence, Signal-to-Noise Ratio (SNR) of the reconstructed haptic signal at the receiver, and

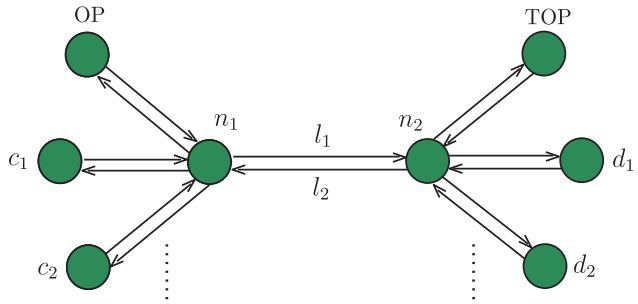


Fig. 8. Single bottleneck dumbbell network topology design for simulation of the telehaptic communication. c_1, c_2, d_1, and d_2 represent the cross-traffic nodes; l_1 and l_2 represent the bottleneck links on the forward and the backward channels, respectively.

the perceptual quality of the displayed haptic-audio-video signal. We first describe the setup used in our simulations, and then describe the setup of the real-time telepottery experiment. The results of these experiments follow in Section 4.

3.1. Simulation Setup

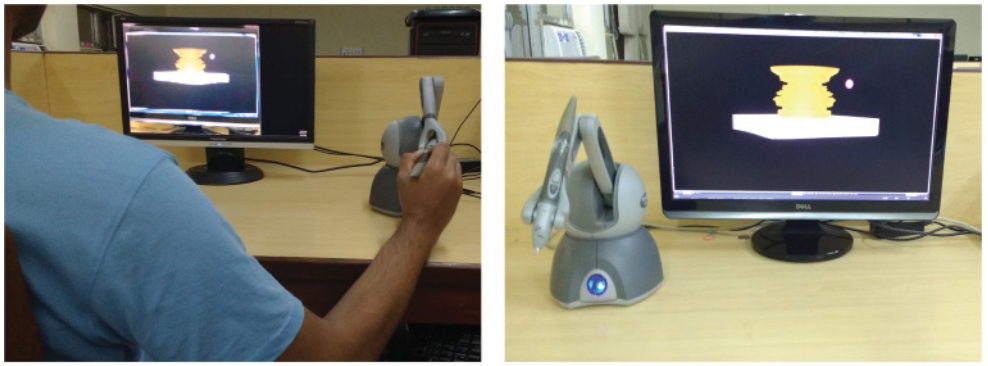
Our simulations are carried out using NS-3, a discrete event network simulator [ns3 2011]. We consider a network with a single bottleneck dumbbell topology connecting the OP and the TOP, as shown in Figure 8. In order to simulate asymmetric network conditions on the forward and the backward channels, we create unidirectional links between the OP and the TOP nodes. All links have identical capacities (denoted by μ) of 1.5 Mbps.[6] To simulate cross-traffic on the forward (backward, respectively) channel, we add source-destination pairs (c_i, d_i) $((d_j, c_j)$, respectively) as indicated in Figure 8. Note that l_1 and l_2 act as the bottleneck links for the telehaptic traffic on the forward and the backward channels, respectively. Thus, queueing delay experienced by the telehaptic application due to network cross- traffic is observable only at the intermediate nodes n_1 and n_2.

For our simulations, we use real haptic traces recorded during a telehaptic activity. Therefore, the haptic payload rates on the forward and the backward channels amount to 192 kbps and 96 kbps [Gokhale et al. 2015], respectively. The TOP generates audio frames with $s_a = 160$ bytes, $f_a = 50$ Hz, and video frames with $s_v = 2{,}000$ bytes, and $f_v = 25$ Hz. This corresponds to payload rates on the backward channel of 64 kbps and 400 kbps for audio and video, respectively.[7] Considering the packet header sizes (see Figure 3), the no-merge data rates on the forward and the backward channels are calculated to be 688 kbps and 1,096 kbps, respectively.

Finally, the propagation delay of each link is set to 5 ms. Hence, the one-way propagation delay (denoted by τ) is 15 ms, which is typically the propagation delay exhibited by a transcontinental link of around 2,000 miles. The buffer size at the ingress of the bottleneck link is set to 15,000 bytes. All nodes follow First-In-First-Out (FIFO) and droptail queueing of packets.

[6] 1.5 Mbps has been picked to represent the typical capacity of a medium speed internet link. However, the nature of our findings remains robust to the channel capacity.

[7] We simulate dummy audio/video streams as per the aforementioned specifications for our simulations, since our QoS measurements are insensitive to the payload itself. Of course, our perceptual experiments (described in Section 3.2) are performed with real audio/video data.



(a) (b)

Fig. 9. Real-time telepottery experimental setup showing (a) human operator and (b) teleoperator.

3.2. Perceptual Experiment Setup

It is important to investigate the qualitative effect of DPM on the human multimedia perception, which is not possible through simulations. For this purpose, we conduct a real-time telepottery experiment in which a human subject interacts with a remote, virtual pottery model on a real network through haptic, audio, and visual feedback, as described in Gokhale et al. [2015]. Figure 9 demonstrates the setup of the telepottery experiment showing the human subject remotely exploring the virtual clay model. The volume preserving pottery model [Chaudhury and Chaudhuri 2014] is rendered at the TOP, which is equipped with a haptic device and a generic webcam. The interaction with the remote scene happens through audio-visual feedback and a separate haptic device for the force feedback. The master-slave relationship between the two haptic devices is implemented using a proportional-derivative controller (see Gokhale et al. [2015] for more details).

The subjects were initially briefed about the concept of force feedback as a few of them were new to the notion of haptics. Later, we explained the telepottery task to them in detail, accompanied by a live demonstration of the task. The telepottery task involves the subject exploring and manipulating a rotating virtual clay model. The task is to design a clay pot. Whenever the haptic interaction point collides with the clay model, the subject hears a filing sound along with the force feedback. There is no benchmarking so far as the shape of the pots is concerned, since the idea behind the experiment is to assess human perception and not the skill. The subject pushes the haptic device stylus so as to establish contact with the clay model and shape it into a pot. The *training* phase involved the participants performing the task to get acquainted with the telepottery setup. During this phase, the participants explored the telepottery model under an expert's guidance until they were confident of performing the task independently. In order to avoid any perceptual degradation due to the network, the training was performed on a very high bandwidth network, under the no-merge packetization scheme.

After the training, the subjects were moved to a *test* setup consisting of a network emulator tool that allows for configuring the network capacity and the propagation delay. Under the emulated network conditions, the subjects independently perform the telepottery task twice: once with no-merge scheme, and once with the proposed DPM scheme. Finally, the subjects were asked to grade the experience of each of the two test experiments, relative to the training, based on three perceptual parameters: *transparency* (the subjects felt as if they were present in the remote virtual environment

and are directly interacting with the objects) [Lawrence 1993], *smoothness* (how smooth or jerky is the feedback) [Isomura et al. 2013], and *overall experience*. The grading of each of the three parameters was based on Degradation Category Rating (DCR) [Hoshino et al. 2011; Suzuki and Katsura 2013; Fujimoto et al. 2008] that assigns a subjective scale to a text descriptor in the following manner:

5 - imperceptible; 4 - slight disturbance, but not annoying; 3 - slightly annoying; 2 - annoying; 1 - very annoying.

For example, the subjects chose 5 if they felt that the degradation in perceptual quality of the test experiments was imperceptible compared to the training phase.

The average training duration was measured to be around 12 minutes, and the average duration for each of the test experiments was around 6 minutes. The subjects had no prior knowledge about the protocol being tested, thereby avoiding grading bias.

3.2.1. System Settings. In the real-time telepottery experiment, we use two Phantom Omni haptic devices. Two desktop computers, each with 4 GB RAM and running Windows 7 operating system are employed. The audio-visual information is captured at the TOP using a Microsoft Lifecam VX-2000 webcam. The TOP transmits uncompressed audio and video frames at the rate of 164 kbps and 9 Mbps (video frames with spatial resolution of 150×100 at 25 Hz), respectively. For these experiments, we increase the channel capacity by 8.7 Mbps compared to the simulation setup to account for the additional audio/video payload.

The network is emulated using a standard network emulator tool called Dummynet [Rizzo 1997]. The training phase of the telepottery experiment is performed on a 100 Mbps network. For the testing phase, the emulated channel capacity and one-way propagation delay are configured to 10.2 Mbps and 15 ms, respectively, for both the forward and the backward channel. In the testing phase, we introduce CBR cross-traffic stream of intensity $R_{cbr} = 400$ kbps on the backward channel. In addition, we introduce a Variable Bit-Rate (VBR) source with intensity $R_{vbr} \in [320, 480]$ kbps with a mean of 400 kbps on the backward channel. The cross-traffic on the forward channel is similar to that on the backward channel.

3.2.2. Human Subjects. The call for participation in the telepottery task was published on noticeboards in the university. All human subjects who took part in the experiment were either students or faculty members at the university. A total of 20 subjects (10 female and 10 male, 18 right-handed and 2 left-handed) participated in the perceptual task. The subjects belonged to the age group of 23 to 52 years, and none of them suffered from any known neurophysiological disorders. Out of the 20 participants, 14 were novice haptic users and the rest were regular users of haptic devices. However, all subjects underwent extensive training prior to the test experiments.

4. EXPERIMENTAL RESULTS

In this section, we present a comprehensive experimental evaluation of DPM. Simulation results are presented in Section 4.1, and the results of our perceptual experiments are presented in Section 4.2. In Section 4.3, we compare the performance of DPM with the state of the art in telehaptic communication protocols.

4.1. Simulation Results

In this section, we present the performance evaluation of DPM via simulations. Specifically, we analyze the interplay between DPM and network-oblivious cross-traffic, highlighting DPM's response to highly congested network conditions. For brevity, we present

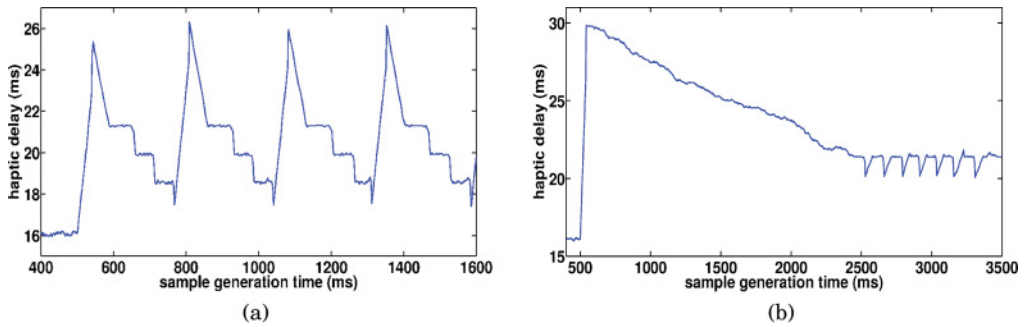


Fig. 10. Haptic delay profile as a result of DPM in the presence of CBR cross-traffic for (a) 260 kbps and (b) 400 kbps.

results corresponding to only the backward channel; the performance of DPM on the forward channel is similar.

The simulation begins at time $t = 0$, at which point the telehaptic stream commences transmission. Starting at $t = 0$, we also maintain the VBR stream on the backward channel with intensity $R_{vbr} \in [320, 480]$ kbps with a mean of 400 kbps.[8] At $t = 500$ ms, we additionally introduce a CBR cross-traffic stream on the backward channel. The intensity of the CBR source R_{cbr} is used as a control parameter to tune the level of congestion on the backward channel. Note that the peak telehaptic data rate on the backward channel equals 1,096 kbps (under no-merge packetization), whereas the minimum data rate equals 694 kbps (with $k = 4$ under DPM). Thus, when $R_{cbr} > 4$ kbps the telehaptic stream has insufficient bandwidth to transmit at its peak rate. (Recall that $\mu = 1.5$ Mbps.) Moreover, $R_{cbr} > 406$ kbps implies that the network is overloaded, since the available capacity is insufficient to even sustain the minimum telehaptic data rate. Thus, the effectiveness of DPM is to be gauged over the range of $R_{cbr} \in [4, 406]$ kbps. In most of our experiments, we set $R_{cbr} = 400$ kbps, which represents a highly congested backward channel. The cross-traffic rates on the forward channel are identical to that on the backward channel. The simulations run for 500 seconds. The throughput, average jitter, and packet loss measurements presented in this section are computed after the CBR cross-traffic is switched on, that is, over the interval $t \in [0.5, 500]$ seconds.

It is important to note that since the proposed protocol operates at the Transport Layer (TL), all delays reported in this section are TL-TL measurements. In other words, we report the latency between the arrival of a haptic/audio/video sample at the TL of the sender and the reception of the same sample at the TL of the receiver. The delays experienced by the OP/TOP in practice (the so-called *glass-to-glass* delays) would include the additional lag introduced by the media devices as well as the encoding/decoding latency.

Temporal Variation of Haptic Delay: We begin by demonstrating the temporal evolution of the delay experienced by the haptic stream under DPM. Figure 10 shows the delay experienced by the haptic samples as a function of the sample generation time, corresponding to $R_{cbr} = 260$ kbps and 400 kbps.

Let us first consider Figure 10(a). For $R_{cbr} = 260$ kbps, the capacity available to the telehaptic stream on the backward channel equals 840 kbps, which is less than the

[8]A Skype video-conferencing connection consumes approximately 400 kbps of bandwidth in each direction. Thus, the VBR cross-traffic can be thought of as a video-conferencing stream contending with the telehaptic stream on the bottleneck link.

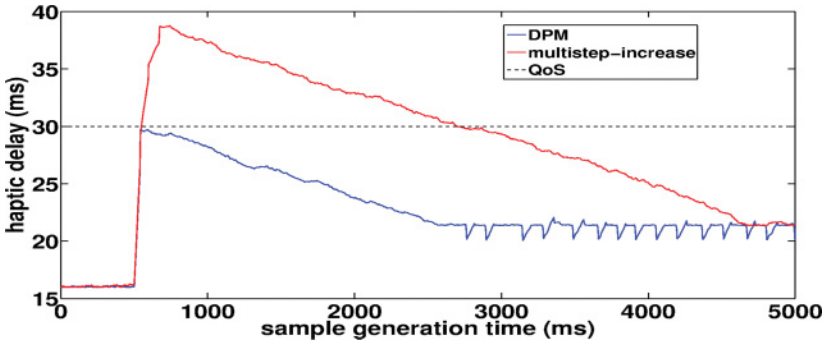


Fig. 11. Haptic delay variation with DPM and multistep-increase approaches, in the presence of $R_{cbr} = 400$ kbps.

no-merge transmission rate of 1,096 kbps, but more than the 2-merge transmission rate of 828 kbps. Once the CBR source turns on at $t = 500$ ms, the telehaptic stream, initially operating at $k = 1$, sees a rapid delay buildup. DPM responds to this buildup by switching to $k = 4$. This aggressive rate reduction allows the network buffers to drain quickly, avoiding a QoS violation. Once DPM sees a steady delay zone, it probes for a higher telehaptic data rate by decreasing k by 1. But when DPM makes the switch from $k = 2$ to $k = 1$, the overall network load once again exceeds the capacity of the bottleneck link. This in turn leads to a delay buildup, and the cycle repeats.

Figure 10(b) has a similar interpretation. For $R_{cbr} = 400$ kbps, the capacity available to the telehaptic stream on the bottleneck link equals 700 kbps, which is less than the 3-merge transmission rate of 739 kbps, but more than the 4-merge transmission rate of 694 kbps. In this case, the switch from $k = 4$ to $k = 3$ causes a delay buildup, forcing DPM to revert to $k = 4$.

In conclusion, we see that DPM adapts its transmission rate depending on the intensity of cross-traffic it experiences. Moreover, against a steady cross-traffic, DPM results in a roughly periodic delay evolution. This is typical of congestion control algorithms (see, e.g., Ha et al. [2008]). Note that even when the backward channel is highly congested (see Figure 10(b)), DPM manages to keep the telehaptic delays below the prescribed QoS limits.[9]

Benefits of Step-Increase in DPM: Recall that DPM responds to network congestion with an aggressive transmission rate reduction (achieved by a *step-increase* in k to k_{max}), as opposed to a gradual transmission rate reduction (which would be achieved by a *multistep-increase* in k). Figure 11 highlights the benefits of employing the step-increase mechanism over a multistep-increase approach for telehaptic data rate reduction. Specifically, we compare the performance of DPM with an algorithm that increases k by one on receiving the congestion trigger I_C (so long as $k < k_{max}$). For this experiment, we set $R_{cbr} = 400$ kbps. Once the CBR stream starts, the telehaptic stream, initially operating at $k = 1$, experiences a rapid delay buildup due to increased queueing in the network. Note that DPM responds with an aggressive rate reduction ($k = 4$), allowing the network buffers to get flushed quickly, avoiding a QoS violation. On the other

[9]It is worth remarking that the haptic delay is dependent on the overall cross-traffic intensity, the link capacities, as well as propagation delays. For haptic QoS compliance, we need to ensure that the maximum haptic delay does not exceed 30 ms. We perform a mathematical analysis for characterizing the maximum haptic delay in Appendix B. This enables us to identify the network settings under which haptic QoS adherence is feasible. Furthermore, we extend this characterization to audio and video in Appendix C, and show that the haptic QoS compliance in general guarantees audio and video QoS compliance.

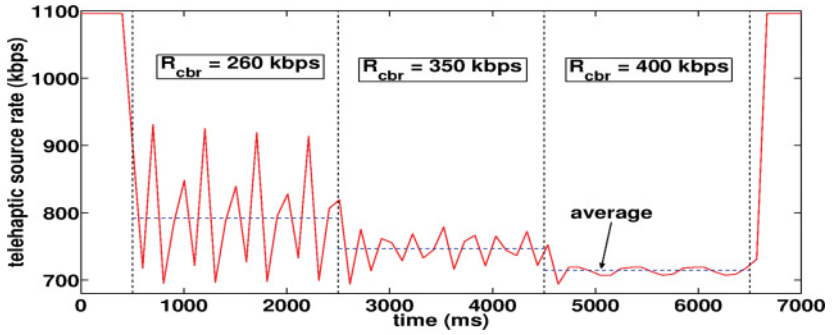


Fig. 12. Telehaptic source rate evolution under time-varying cross-traffic conditions.

Table I. Comparison of the Maximum Telehaptic Delay and Jitter Observed for Different Media for $R_{cbr} = 400$ kbps, Along with the Corresponding QoS Specifications

	Max. Delay (ms)		Max. Jitter (ms)	
	QoS	Observed	QoS	Observed
Haptic	30	29.738	10	3.628
Audio	150	27.952	30	5.372
Video	400	63.629	30	8.255

hand, the multistep-increase approach cuts the transmission rate in stages, requiring three rate adaptations before setting $k = 4$. As a result, network decongestion occurs much later, leading to a violation of the haptic QoS constraint. Thus, we conclude that DPM's SIMD approach is suitable for congestion control for delay-critical telehaptic applications.

Adaptation to Time-Varying Cross-Traffic: In order to test the robustness of DPM to time-varying cross-traffic conditions, we simulate three CBR sources on the backward channel: C_1, C_2, and C_3 with data rates of 260 kbps, 90 kbps, and 50 kbps, respectively. Each of these sources operates over a different interval of time, resulting in an overall cross-traffic scheme shown in Equation (6).

$$R_{cbr} = \begin{cases} 0, & \text{for } 0 < t \leq 500 \text{ ms} \\ 260 \text{ kbps } (C_1), & \text{for } 500 \text{ ms} < t \leq 2500 \text{ ms} \\ 350 \text{ kbps } (C_1 \text{ and } C_2), & \text{for } 2500 \text{ ms} < t \leq 4500 \text{ ms} \\ 400 \text{ kbps } (C_1, C_2 \text{ and } C_3), & \text{for } 4500 \text{ ms} < t \leq 6500 \text{ ms} \\ 0, & \text{for } t > 6500 \text{ ms}. \end{cases} \quad (6)$$

Figure 12 shows the temporal variation of DPM source rate. Until 500 ms, DPM achieves its peak rate since the network is uncongested. After 500 ms, the network is unable to support the peak rate, and DPM automatically lowers the telehaptic data rate to avoid congestion. Note that as R_{cbr} increases, DPM lowers its transmission rate progressively. Once the CBR cross-traffic is completely withdrawn at $t = 6{,}500$ ms, DPM reverts to its peak rate. Thus, we see that DPM exhibits cross-traffic friendliness, and performs a robust congestion control under time-varying cross-traffic settings.

Delay and Jitter Measurements: Table I summarizes the observed telehaptic delay and jitter for haptic, audio, and video streams, respectively, with $R_{cbr} = 400$ kbps. It can be seen that even under heavy cross-traffic conditions, DPM enables the telehaptic

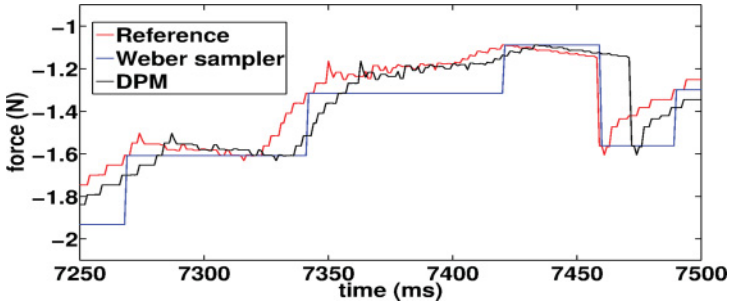


Fig. 13. Graph showing the reconstructed force signals at OP with Weber sampling and DPM for $R_{cbr} = 400$ kbps.

Table II. Comparison of SNR (in dB) in the Case of Weber Sampler and DPM, with $R_{cbr} = 400$ kbps on Backward Channel

	SNR (dB)	Improvement over WS (dB)
Weber sampler (WS)	21.5518	-
DPM	24.0986	2.5468

application to comply with the QoS limits. Note that the measured haptic jitter of 3.628 ms (see Section 2.4 for an analysis) is significantly below the QoS jitter limit.

Haptic Signal Reconstruction: We now study the effects of network cross-traffic, DPM, and data extrapolation on the haptic signal reconstruction at the OP. We compare the reconstructed signal with that corresponding to an adaptively sampled strategy, and measure the improvement in haptic signal display that DPM yields. For this purpose, we use real telehaptic traces captured during the telepottery experiment. Ten pilot telehaptic signals were used in the evaluation of the proposed scheme, with each signal corresponding to a different human subject. For brevity, we present results corresponding to a particular pilot signal. We employ a Weber sampler with a threshold of 12% for adaptively sampling the force samples at the TOP [Hinterseer et al. 2008]. We use the standard zero-order hold strategy for haptic data extrapolation. For this experiment, we set $R_{cbr} = 400$ kbps.

For benchmarking, we make use of a reconstructed signal captured using an ideal (high bandwidth, zero jitter) network; we treat this signal as the reference signal. Figure 13 shows the force signal displayed at the OP under different schemes. As expected, DPM, being a lossless protocol, captures the fine details of the reference signal well. On the other hand, the Weber sampled signal is a piecewise constant approximation of the reference signal. It is to be noted that under the Weber sampling strategy, the *perceptually significant* samples are displayed earlier at the OP as compared to DPM. This is because of the higher packetization delay under DPM. Using SNR as a performance metric to measure the reconstruction error at the OP (against the reference signal), Table II compares the SNR (in dB) measured for the reconstructed haptic signal under different schemes. We see that DPM exhibits a substantial SNR improvement of around 2.5 dB over the Weber sampling strategy. In our experiments, we have found a comparable SNR improvement for other haptic traces.

Throughput-Loss Measurements: Figure 14 compares the performance of DPM and the no-merge scheme,[10] in terms of throughput and packet losses, under various CBR

[10]Recall that the no-merge scheme transmits at the peak telehaptic data rate, oblivious to the state of the network. In the literature, this scheme is also referred to as plain UDP [Postel 1980].

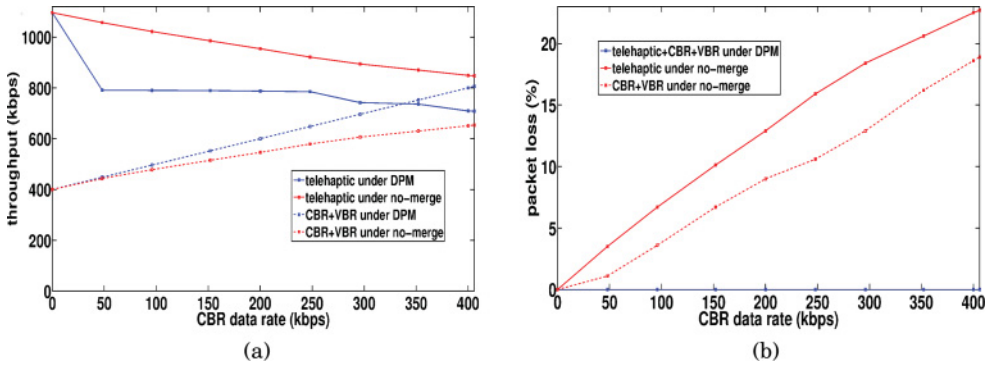


Fig. 14. Telehaptic-CBR traffic interplay demonstrating the improvement of DPM over no-merge in terms of (a) throughput and (b) packet loss.

cross-traffic conditions. The results show that for $R_{cbr} < 4$ kbps, the two schemes exhibit similar behavior since the network can sustain the peak telehaptic data rate. As R_{cbr} increases further, the DPM appropriately lowers the telehaptic data rate resulting in zero packet loss until R_{cbr} approaches 406 kbps. On the other hand, the no-merge scheme demonstrates deteriorated performance when $R_{cbr} > 4$ kbps due to its network obliviousness.

Figure 14(b) shows that the telehaptic and cross-traffic streams sustain severe packet losses with increasing R_{cbr} under the no-merge scheme, whereas DPM avoids packet losses altogether by adapting the telehaptic data rate to the intensity of cross-traffic. We note that DPM is friendly to CBR and VBR cross-traffic. Indeed, the cross-traffic streams see a higher throughput (and zero loss) under DPM as compared to no-merge.

DPM with Hold-Up: Motivated by Figure 10(a), we propose a variant of DPM that seeks to reduce the jitter induced by the frequent rate adaptations. Recall that in the experiment corresponding to Figure 10(a), the maximum data rate for the telehaptic stream that would keep the bottleneck link stable corresponds to $k = 2$. However, when DPM experiences a steady delay at $k = 2$, it switches to $k = 1$, which starts yet another cycle of rate adaptations. In this case, it is clear that if DPM were to hold on to the setting $k = 2$ for a longer period, there would be a reduction in jitter at the receiver. This motivates the following modification of DPM.

DPM with hold-up is identical to DPM, except for the following modification. It remembers the value of k, say $\hat{k}$, that was operating when the previous I_c trigger was received. Subsequently, once $k = \hat{k} + 1$, the algorithm ignores I_S triggers for a *hold-up duration* T_h.

Note that the hold-up modification would work well under steady or slowly varying cross-traffic conditions. Indeed, if one assumes that the cross-traffic is steady, then one may conclude that the previous I_C trigger was actually caused by the rate adaptation $\hat{k} + 1 \rightarrow \hat{k}$. This suggests that $\hat{k} + 1$ is currently the optimal operating point for the algorithm. Thus, once in this state, *DPM with hold-up* puts off attempts to increase its rate further for a period T_h. Of course, this modification is pessimal in that it misses any opportunities for increasing the transmission rate during the hold-up period T_h.

Figure 15 shows the haptic delay variation plots for DPM and DPM with hold-up in the case of $R_{cbr} = 260$ kbps and 400 kbps, respectively. T_h is heuristically chosen to be 500 ms. As expected, under the hold-up modification, the cycles of delay fluctuation occur less frequently. The average jitter for DPM and DPM with hold-up for $R_{cbr} = 400$ kbps are measured as 1.3 μs and 0.93 μs, respectively. This implies a reduction in average jitter of around 29% over DPM. The SNR of the reconstructed signal under

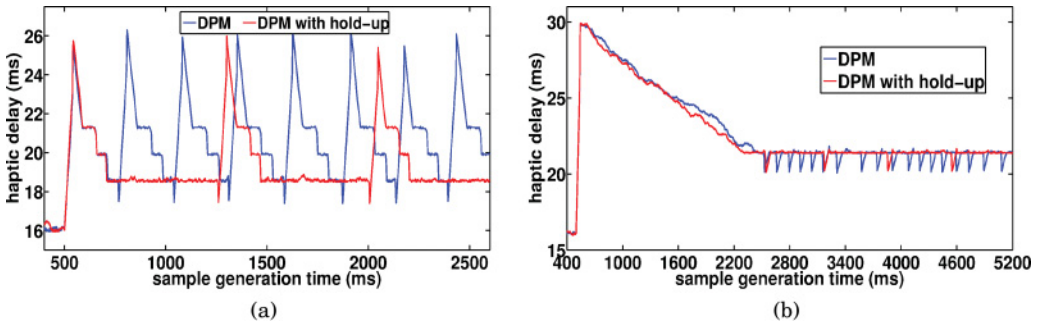


Fig. 15. Haptic delay plots for DPM and DPM with hold-up techniques in the presence of CBR cross-traffic for (a) 260 kbps and (b) 400 kbps.

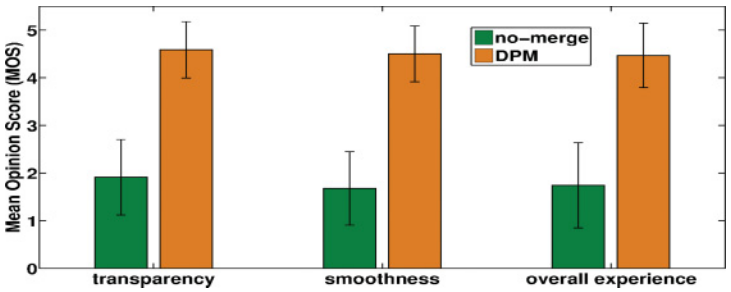


Fig. 16. MOS of the subjective evaluation of the proposed technique on three specific perceptual parameters, averaged over 20 human subjects. The vertical bars denote the standard deviation of the subject grades.

DPM with hold-up is measured to be 24.8332 dB, which is around 0.7 dB higher than the SNR under DPM.

In conclusion, when it is known a priori that the cross-traffic is slowly varying, the hold-up modification provides a modest QoS improvement over DPM.

4.2. Telepottery Subjective Grading

We now move to the qualitative results of the real-time telepottery task. Figure 16 presents the Mean Opinion Score (MOS) of the DCR recorded with 20 human subjects for each of the three perceptual parameters, that is, transparency, smoothness, and overall experience. We observe that the MOS recorded while using the no-merge technique is less than 2, which corresponds to an annoying user experience. In fact, a few subjects found the no-merge experience so disturbing that they hardly made any contact with the clay model. When DPM is employed, the MOS under each of the three perceptual parameters improves substantially (in the neighborhood of 4.5, signifying nearly imperceptible degradation in the user experience).

In order to statistically evaluate the improvement in the perception of DPM over the no-merge scheme, we perform a paired t-test over the subject grades. The test results corresponding to the aforementioned perceptual parameters are as follows: (i) transparency - $t(19) = 10.81$, $p < 0.001$; (ii) smoothness - $t(19) = 13.97$, $p < 0.001$; and (iii) overall experience - $t(19) = 11.72$, $p < 0.001$. This further substantiates our claim that the rate adaptation mechanism of DPM introduces negligible perceivable artifacts.

Thus, we conclude that DPM preserves the immersiveness of the telepottery activity in spite of heavy cross-traffic on the network, thereby resulting in a user-friendly and enjoyable telepottery experience.

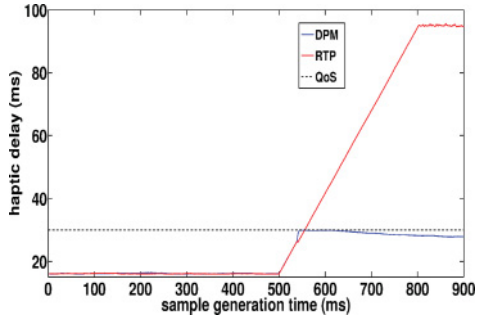


Fig. 17. Early congestion detection and responsiveness of DPM as opposed to sluggish behavior of RTP.

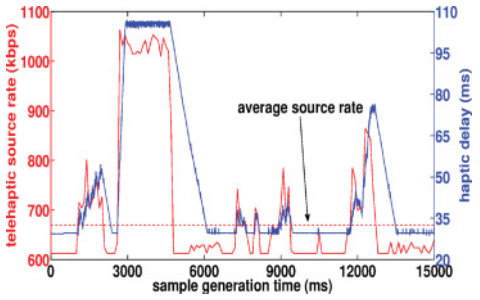


Fig. 18. Telehaptic rate-delay plot of visual-haptic multiplexing on the backward channel.

4.3. Comparison with Existing Telehaptic Communication Techniques

In this section, we compare the performance of DPM with RTP [Schulzrinne et al. 2003] and other recently proposed telehaptic communication protocols.

(1) RTP: We begin by comparing DPM with RTP, which is the predominant protocol for media streaming applications on the Internet. We use the simulation setup from Section 4.1, with $R_{cbr} = 400$ kbps on the backward channel. Figure 17 shows the variation of the end-to-end delay experienced by haptic samples with the sample generation time. Note that once the CBR cross-traffic is introduced at 500 ms, DPM performs a prompt rate adaptation, maintaining end-to-end delays below the QoS deadline of 30 ms. In the same setting, RTP generates its first and the second RTCP reports at 500 ms and 1000 ms, respectively. Since any rate-control mechanism based on RTP would not make a rate adaptation prior to 1000 ms, the haptic delays under any such protocol would keep growing as shown in Figure 17, violating the QoS deadlines. Note that network queues build up on the timescale of tens of milliseconds. Thus, for telehaptic applications, RTP, which generates network feedback reports every 500 ms, is too slow to allow for timely rate-adaptation.

(2) Visual-Haptic Multiplexing: We now evaluate DPM against the visual-haptic multiplexing scheme [Cizmeci et al. 2014], which employs the Weber sampler for force updates on the backward channel. For this evaluation, we use the simulation setup from Section 4.1 with $R_{cbr} = 400$ kbps. Figure 18 shows the source rate evolution and the resulting haptic delays under visual-haptic multiplexing obtained using one of the traces from our real-time telepottery experiment. We see that even though the available capacity on the backward channel (700 kbps) exceeds the average transmission rate on the backward channel (670 kbps), the instantaneous rate fluctuates substantially, resulting in occasional QoS violations (e.g., see the interval from 3000 to 6000 ms). Indeed, during interaction with hard objects, almost every force sample becomes perceptually significant, causing a Weber sampler's instantaneous transmission rate to far exceed its time average. It is also worth noting that packet loss measured between 3000 ms and 6000 ms is around 16%, which could potentially lead to significant perceptual degradation. In contrast, under the same network conditions, the results of Section 4.1 show that DPM meets the QoS constraints and results in zero packet loss.

(3) Network Adaptive Flow Control Algorithm for Haptic Data (NAFCAH): We now compare the performance of NAFCAH [Kokkonis et al. 2015], a protocol that performs RTT-based rate adaptation on the *forward channel*, with DPM. We use the simulation setup from Section 4.1, except that the CBR cross-traffic intensity on the

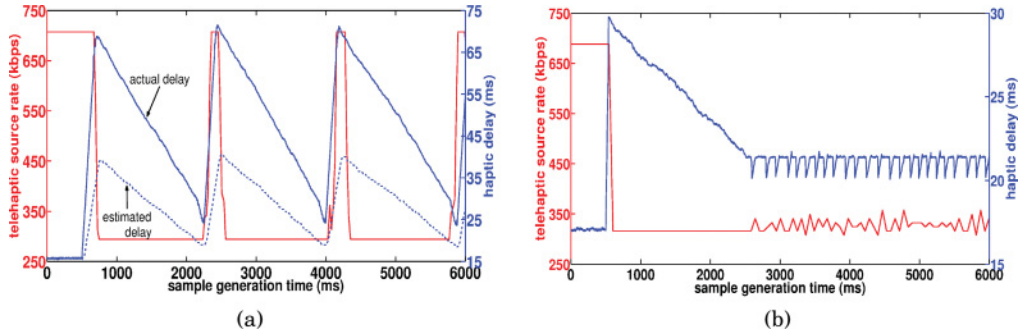


Fig. 19. Telehaptic rate-delay plots for the forward channel with (a) NAFCAH and (b) DPM.

forward channel is increased to 780 kbps; this makes the forward channel highly congested. With the probing packet frequency of NAFCAH set as 100 Hz, Figure 19(a) shows the evolution of the source transmission rate and the delay experienced by the haptic samples under NAFCAH. Note that NAFCAH incurs substantial QoS violations. The reasons for this are twofold: Firstly, once congestion is detected, NAFCAH cuts its transmission rate in stages (i.e., it employs a *multistep-increase* approach). As discussed in Section 4.1, this results in a relatively sluggish congestion control. Secondly, NAFCAH uses RTT measurements to estimate congestion on the forward channel. This leads to incorrect delay estimations under asymmetric network conditions, as shown in Figure 19(a).

In contrast, as seen in Figure 19(b), DPM satisfies the QoS constraints well under the same network conditions, thanks to its aggressive *step-increase* mechanism for rate reduction, and its accurate end-to-end delay estimation mechanism.

5. CONCLUSIONS AND LIMITATIONS

In this article, we presented DPM, a transport layer congestion control protocol for a lossless, real-time telehaptic communication. In order to enable DPM to quickly respond to network variations, we proposed the network feedback module for communicating the end-to-end delays to the transmitters with negligible overhead. Via extensive simulations, we showed that DPM meets the QoS requirements of telehaptic applications even under highly congested network conditions. We also validated DPM's ability to provide a seamless and immersive user experience over a congested network via a real-time telepottery experiment with human subjects. Finally, we showed that DPM outperforms state of the art in telehaptic communication protocols.

While the present article explores the interplay between DPM and network-oblivious UDP traffic, the interplay between DPM and other network-aware cross-traffic streams (predominantly TCP) remains unexplored. Further, it is not clear as to how multiple DPM streams coexisting on a network would share the available bandwidth. Finally, the implications of SNR improvement of our scheme over Weber sampling on the quality of the telehaptic task has not been investigated. We would like to address these issues in a future extension of this article.

APPENDIXES

A. APPLICATION LAYER PACKET STRUCTURE

While the proposed protocol performs a transport layer function, in our implementation, we code the protocol in the application layer leveraging UDP at the transport layer. In this section, we describe the various application layer header fields of a telehaptic

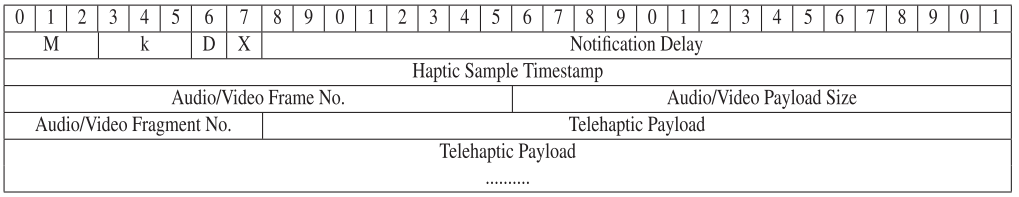


Fig. 20. Telehaptic packet format at the application layer. The top row is numbered bitwise for illustration.

Table III. Detailed Description of the Application Layer Header Structure

Field	Bits	Description
M	3	Indicates the type of media data contained in the payload. 0: haptic, 1: haptic-audio, 2: haptic-video. The additional bit is included to provide support for additional media types.
k	3	Indicates the current value of k used for packetization.
D	1	Delay indicator field. Indicates the transmission status of the delay embedded in the packet header. 0 - fresh transmission, 1 - repetitive transmission.
X	1	Reserved for future enhancements to the protocol.
Notification Delay	24	End-to-end delay inserted by the in-header delay notification mechanism.
Haptic Sample Timestamp	32	Indicates the generation time (in ms) of the haptic sample in the payload in case of $k = 1$. In case of a higher k, this field indicates generation time of the earliest of the k haptic samples in the payload.
Audio/Video Frame No.	16	Indicates frame number of the current audio/video payload being carried by the packet.
Audio/Video Payload Size	16	Indicates the size of the audio/video payload in bytes, carried by the packet.
Audio/Video Fragment No	8	Indicates fragment number of the current audio/video frame being carried by the packet.

packet in our implementation. Figure 20 shows the proposed application layer packet structure for telehaptic communication. The topmost row is shown for convenience, to indicate the bit positions in the packet. The packet structure starts with the field *M*. The haptic header size is 8 bytes, whereas the audio/video headers consume 5 bytes. Since the focus is only on augmenting either audio or video with haptic data, the effective application layer header size is 13 bytes. The audio (video) related headers are included only in the presence of audio (video) payload. Table III describes each of the header fields in detail. The telehaptic payload includes haptic-audio/video payload based on the value indicated in the field *M*. Haptic payload on the forward channel includes position and velocity information of the operator, whereas the backward channel carries force information.

B. CHARACTERIZATION OF MAXIMUM HAPTIC DELAY

In this section, we derive an expression for the maximum end-to-end delay experienced by haptic samples under DPM over a single bottleneck network topology (see Figure 8) with CBR cross-traffic.[11] This analytical characterization enables us to identify the class of network configurations where QoS-compliant telehaptic communication is feasible.

[11]Note that since our protocol operates at the TL, we characterize the maximum TL-TL latency, that is, the maximum latency between the arrival of a haptic sample at the TL of the sender and the reception of the sample at the TL of the receiver.

Let R_{cbr} denote the CBR cross-traffic intensity (in kbps) on the channel under consideration. For simplicity, we assume that the reverse channel is uncongested, so that the packetization rate on the reverse channel equals 1 kHz. Recall that τ and μ denote the one-way propagation delay (in ms) and the bottleneck channel capacity (in kbps), respectively. Define

$$k_{opt} = \min\{k \in \{1, 2, \ldots, k_{max}\} : R_k + R_{cbr} \leq \mu\}.$$

Note that when DPM operates at $k \geq k_{opt}$, the bottleneck link remains uncongested. It then follows that in steady state, the maximum end-to-end delay is experienced during the buffer buildup that results from DPM switching from $k = k_{opt}$ to $k = k_{opt} - 1$.

For simplicity, let us denote the instant when DPM sets $k = k_{opt} - 1$ by $t = 0$. Let d_{inc} denote the generation time of the resulting congestion trigger. Note that d_{inc} is the time required for the delay measurement corresponding to the N^{th} packet transmitted after $t = 0$ to arrive at the transmitter. We can write an expression for d_{inc} as follows.

$$d_{inc} = N(k_{opt} - 1)(1 \text{ ms}) + \frac{N(R_{cbr} + R_{k_{opt}-1} - \mu)}{\mu}(k_{opt} - 1)(1 \text{ ms}) + 2\tau + 1 \text{ ms}.$$

The first term above is the generation time of the N^{th} packet after $t = 0$. The second term is the queueing delay seen by this packet. The third term equals the round-trip propagation delay, and the fourth term (1 ms) represents the time gap between arrival of the N^{th} packet at the receiver and the piggybacking of its delay on the reverse channel.

Since the queue at the ingress of the bottleneck link builds up at the rate of $R_{cbr} + R_{k_{opt}-1} - \mu$ until time d_{inc}, we obtain the following expression for the maximum queue occupancy.

$$q_{inc} = (R_{cbr} + R_{k_{opt}-1} - \mu)d_{inc}.$$

The maximum end-to-end delay would be clearly experienced by the packet that sees a queue occupancy of q_{inc}. This leads us to the following expression for the maximum end-to-end delay:

$$d_{hap} = \tau + \frac{q_{inc}}{\mu} + (k_{opt} - 1)(1 \text{ ms}).$$

The first term above captures the one-way propagation delay, the second term captures the maximum queueing delay, and the last term captures the packetization delay seen by the earliest haptic sample in the packet. Combining the above equations, we get

$$\begin{aligned} d_{hap} &= \tau + (k_{opt} - 1)(1 \text{ ms}) + [N(k_{opt} - 1)(1 \text{ ms}) + 2\tau + 1 \text{ ms}]\left(\frac{R_{cbr} + R_{k_{opt}-1} - \mu}{\mu}\right) \\ &\quad + N(k_{opt} - 1)(1 \text{ ms})\left(\frac{R_{cbr} + R_{k_{opt}-1} - \mu}{\mu}\right)^2. \end{aligned} \tag{7}$$

We have validated the accuracy of the above expression via simulations.

Note that Equation (7) enables us to characterize the set of link capacities, propagation delays, and cross-traffic intensities that satisfy the haptic QoS constraints. In the following section, we relate the maximum end-to-end haptic delay to the maximum end-to-end delay seen by the audio and video streams.

C. CHARACTERIZATION OF THE MAXIMUM AUDIO/VIDEO DELAY

In this section, we derive an upper bound on the maximum end-to-end audio/video delays under DPM over a single bottleneck network topology (see Figure 8) with CBR

cross-traffic.[12] Interestingly, these upper bounds involve the maximum haptic delay d_{hap} characterized in Appendix B. Thus, we are able to relate haptic QoS compliance to QoS compliance for audio and video.

Recall that f_a and f_v represent the (peak) frame rates (in Hz) of audio and video, respectively. Also, s_a and s_v represent the (peak) audio and video frame sizes (in bytes), respectively. Finally, s_m represents the size of the audio/video data (in bytes) in each telehaptic fragment (see Section 2.2).

Note that our media multiplexing framework guarantees that at the instant an audio frame is generated, the previous audio frame has already been multiplexed with the haptic stream. Thus, the multiplexing latency seen by the audio frame equals $\frac{s_a}{s_m}(1\text{ ms})$. There is an additional packetization latency that is at most $(k_{max} - 1)(1\text{ ms})$. Finally, the maximum end-to-end delay experienced by the packet equals d_{hap}. This yields the following upper bound on the (TL-TL) audio delay.

$$d_{aud} \leq d_{hap} + \frac{s_a}{s_m}(1\text{ ms}) + (k_{max} - 1)(1\text{ ms}). \tag{8}$$

Next, we move to the maximum delay experienced by a video frame (TL-TL). For simplicity, we assume that f_a is an integral multiple of f_v. In this case, our multiplexing framework guarantees that by the time a video frame is generated, the previous one has been multiplexed. Thus, the maximum multiplexing delay equals $\frac{1}{f_v}$. Adding to this the maximum packetization delay and the maximum end-to-end delay experienced by a packet, we obtain the following upper bound on the TL-TL video delay.

$$d_{vid} \leq d_{hap} + \frac{1}{f_v} + (k_{max} - 1)(1\text{ ms}). \tag{9}$$

From Equations (8) and (9), we can compute the maximum delay seen by audio/video frames assuming that the QoS constraint on haptic delay is satisfied, that is, $d_{hap} \leq 30$ ms. Consider the settings assumed in our simulations: $f_a = 50$ Hz, $f_v = 25$ Hz, $s_a = 160$ bytes, $s_v = 2{,}000$ bytes. This leads to $s_m = 58$ bytes. It then follows from Equations (8) and (9) that $d_{aud} \leq 35.75$ ms, $d_{vid} \leq 73$ ms. Note that these bounds are well below the audio/video QoS targets. Thus, under the proposed protocol, meeting the (strict) haptic delay constraint in general leads to compliance with the audio/video delay constraint.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for providing their valuable feedback on the paper. We also acknowledge the subjects who participated in the perceptual experiment.

REFERENCES

Hussein Al Osman, Mohamad Eid, Rosa Iglesias, and Abdulmotaleb El Saddik. 2007. Alphan: Application layer protocol for haptic networking. In *Proceedings of the International Workshop on Haptic, Audio and Visual Environments and Games (HAVE)*.

Robert Anderson and Mark Spong. 1989. Bilateral control of teleoperators with time delay. *IEEE Transactions on Automatic Control* 34, 5 (1989), 494–501.

Subhajit Chaudhury and Subhasis Chaudhuri. 2014. Volume preserving haptic pottery. In *Proceedings of Haptics Symposium*.

Dah-Ming Chiu and Raj Jain. 1989. Analysis of the increase/decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems* 17, 1 (1989), 1–14.

[12]Note again that we only consider the maximum TL-TL latency.

Burak Cizmeci, Rahul Chaudhari, Xiao Xu, Nicolas Alt, and Eckehard Steinbach. 2014. A visual-haptic multiplexing scheme for teleoperation over constant-bitrate communication links. In *Haptics: Neuroscience, Devices, Modeling, and Applications*. Springer, 131–138.

Stella Clarke, Gerhard Schillhuber, Michael Zaeh, and Heinz Ulbrich. 2006. Telepresence across delayed networks: A combined prediction and compression approach. In *Proceedings of the International Workshop on Haptic Audio Visual Environments and their Applications (HAVE)*.

Onkar Dabeer and Subhasis Chaudhuri. 2011. Analysis of an adaptive sampler based on Weber's law. *IEEE Transactions on Signal Processing* 59, 4 (2011), 1868–1878.

Mohamad Eid, Jongeun Cha, and Abdulmotaleb El Saddik. 2011. Admux: An adaptive multiplexer for haptic–audio–visual data communication. *IEEE Transactions on Instrumentation and Measurement* 60, 1 (2011), 21–31.

William Ferrell. 1965. Remote manipulation with transmission delay. *IEEE Transactions on Human Factors in Electronics* 1 (1965), 24–32.

Masaki Fujimoto and Yutaka Ishibashi. 2005. Packetization interval of haptic media in networked virtual environments. In *Proceedings of the 4th ACM SIGCOMM Workshop on Network and System Support for Games*.

Takeshi Fujimoto, Yutaka Ishibashi, and Shinji Sugawara. 2008. Influences of inter-stream synchronization error on collaborative work in haptic and visual environments. In *Proceedings of the Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*.

Vineet Gokhale, Subhasis Chaudhuri, and Onkar Dabeer. 2015. HoIP: A point-to-point haptic data communication protocol and its evaluation. In *Proceedings of the 21st National Conference on Communications (NCC)*.

Vineet Gokhale, Onkar Dabeer, and Subhasis Chaudhuri. 2013. HoIP: Haptics over internet protocol. In *Proceedings of the International Symposium on Haptic Audio Visual Environments and Games (HAVE)*.

Vineet Gokhale, Jayakrishnan Nair, and Subhasis Chaudhuri. 2016. Opportunistic adaptive haptic sampling on forward channel in telehaptic communication. In *Proceedings of the Haptics Symposium*.

Sangtae Ha, Injong Rhee, and Lisong Xu. 2008. CUBIC: A new TCP-friendly high-speed TCP variant. *ACM SIGOPS Operating Systems Review* 42, 5 (2008), 64–74.

Peter Hinterseer, Sandra Hirche, Subhasis Chaudhuri, Eckehard Steinbach, and Martin Buss. 2008. Perception-based data reduction and transmission of haptic data in telepresence and teleaction systems. *IEEE Transactions on Signal Processing* 56, 2 (2008), 588–597.

Peter Hinterseer, E. Steinbach, Sandra Hirche, and Martin Buss. 2005. A novel, psychophysically motivated transmission approach for haptic data streams in telepresence and teleaction systems. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Sosuke Hoshino, Yutaka Ishibashi, Norishige Fukushima, and Shinji Sugawara. 2011. QoE assessment in olfactory and haptic media transmission: Influence of inter-stream synchronization error. In *Proceedings of the International Workshop Technical Committee on Communications Quality and Reliability (CQR)*.

Eiichi Isomura, Shuji Tasaka, and Toshiro Nunome. 2013. A multidimensional QoE monitoring system for audiovisual and haptic interactive IP communications. In *Proceedings of the Consumer Communications and Networking Conference (CCNC)*.

Caroline Jay, Mashhuda Glencross, and Roger Hubbold. 2007. Modeling the effects of delayed haptic and visual feedback in a collaborative virtual environment. *ACM Transactions on Computer-Human Interaction (TOCHI)* 14, 2 (2007), 8.

George Kokkonis, Kostas E. Psannis, and Manos Roumeliotis. 2015. Network adaptive flow control algorithm for haptic data over the internet–NAFCAH. *Genetic and Evolutionary Computing*. Springer, 93–102.

Dale A. Lawrence. 1993. Stability and transparency in bilateral teleoperation. *IEEE Transactions on Robotics and Automation* 9, 5 (1993), 624–637.

Seokhee Lee and JongWon Kim. 2007. Haptic event prioritization and network adaptation scheme for collaborative virtual environments. In *Proceedings of the Global Telecommunications Conference*.

Alan Marshall, Kian Meng Yap, and Wai Yu. 2008. Providing QoS for networked peers in distributed haptic virtual environments. *Advances in Multimedia* 2008 (2008).

David L Mills. 1991. Internet time synchronization: The network time protocol. *IEEE Transactions on Communications* 39, 10 (1991), 1482–1493.

Dimitrios Miras, Amela Sadagic, Ben Teitelbaum, Jason Leigh, Magda El Zarki, and Haining Liu. 2002. A survey on network QoS needs of advanced internet applications. *Internet2 QoS Working Group* (2002).

Douglas C. Montgomery. 2007. *Introduction to Statistical Quality Control*. John Wiley & Sons.

ns3. 2011. The network simulator. Retrieved from http://www.nsnam.org/.

Jon Postel. 1980. *User Datagram Protocol*. Technical Report, University of Southern California.

Luigi Rizzo. 1997. Dummynet: A simple approach to the evaluation of network protocols. *ACM SIGCOMM Computer Communication Review* 27, 1 (1997), 31–41.

Nizar Sakr, Nicolas D. Georganas, and Jiying Zhao. 2011. Human perception-based data reduction for haptic communication in six-DoF telepresence systems. *IEEE Transactions on Instrumentation and Measurement* 60, 11 (2011), 3534–3546.

Henning Schulzrinne, Stephen Casner, Ron Frederick, and Van Jacobson. 2003. *RTP: A Transport Protocol for Real-time Applications*. RFC 3550. Retrieved from http://www.rfc-editor.org/info/rfc3550.

Eckehard Steinbach, Sandra Hirche, Marc Ernst, Fernanda Brandi, Rahul Chaudhari, Julius Kammerl, and Iason Vittorias. 2012. Haptic communications. *Proceedings of the IEEE* 100, 4 (2012).

Nobuhiro Suzuki and Seiichiro Katsura. 2013. Evaluation of QoS in haptic communication based on bilateral control. In *Proceedings of the International Conference on Mechatronics (ICM)*.

Tim Szigeti and Christina Hattingh. 2004. *Quality of Service Design Overview*. Cisco, San Jose, CA.

Uras Tos and Tolga Ayav. 2011. Adaptive RTP rate control method. In *Proceedings of the Computer Software and Applications Conference Workshops (COMPSACW)*.

Raul Wirz, Manuel Ferre, Raul Marín, Jorge Barrio, José M. Claver, and Javier Ortego. 2008. Efficient transport protocol for networked haptics applications. In *Haptics: Perception, Devices and Scenarios*. Springer, 3–12.

Received July 2016; revised January 2017; accepted January 2017