Telehaptic Communication over a Shared Network: Protocol Design and Analysis

A thesis submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

by

Vineet Gokhale
(Roll No. 114070002)

Under the guidance of
Prof. Subhasis Chaudhuri
and
Prof. Jayakrishnan Nair

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY BOMBAY
February 2017

Dedicated to

To my beloved parents - Leela and Heramba Gokhale

Thesis Approval

The thesis entitled

Telehaptic Communication over a Shared Network: Protocol Design and Analysis

by

Vineet Gokhale
(Roll No. 114070002)

is approved for the degree of

Doctor of Philosophy

Prof. Rajesh Sundaresan	Prof. Animesh Kumar
(External Examiner)	(Internal Examiner)
Prof. Subhasis Chaudhuri	Prof. Jayakrishnan Nair
(Supervisor)	(Co-Supervisor)

Prof. Ramaswamy Murugavel
(Chairman)

Date: ____________

Place: ____________

Declaration

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: ____________________ ______________________________

Vineet Gokhale

(Roll No. 114070002)

Abstract

The field of *telehaptics* enables humans to control and manipulate remote environments through highly immersive audio-video interfaces, along with the incorporation of force feedback. Telehaptic applications are characterized by the bidirectional nature of communication, wherein the human end transmits position/velocity/torque commands of the human movements to the robot (remote) end, and force/audio/video signals are transmitted in the reverse direction. The inclusion of force feedback has been proven to increase the precision in the control of remote objects to a large extent. Consequently, there is a large improvement in the quality of the telehaptic activity. However, this technology can have a significant impact and outreach only if the communication, between the human and the remote environment, is made possible through the existing network resources, as laying communication cables over long distances purely for telehaptic purposes is impracticable.

While the force feedback brings great value addition to the existing applications, it also presents a variety of fresh challenges. The predominant ones, from the communication standpoint, are the extreme sensitivity of the haptic signal to network irregularities, and maintaining the stability of the haptic control loop. Existing network, like the Internet, is often prone to severe variations in the cross-traffic due to the large number of concurrent users. During periods when the network load overshoots the capacity, congestion occurs. This induces large delays, jitter and packets losses that can have catastrophic effects on the telehaptic applications. Researchers have identified the maximum limits (called *Quality of Service (QoS)*) on each of these parameters that can be tolerated by the telehaptic applications. Strict adherence to these requirements is essential for realizing a seamless telehaptic activity. This motivates us to design a transport layer protocol, tailor-made for telehaptic applications, that shapes the telehaptic data rate to match the available network capacity, and thereby guarantee telehaptic QoS conformance on a shared network.

We begin the design of our transport layer protocol, named *HoIP - Haptics over Internet*

Protocol, by devising a low-latency multiplexing-packetization framework for composing the telehaptic packets comprising of haptic-audio-video data. The framework also incorporates a novel technique that exploits bidirectionality of telehaptic communication for providing a low-overhead, fine-grained feedback for network monitoring. In the next part, we explore two novel telehaptic congestion control mechanisms. Firstly, a *lossy* rate control scheme for the forward channel (human to robot) that is based on the popular haptic data compression technique called *adaptive sampling*. Our rate control scheme opportunistically maximizes the number of haptic samples that are transmitted to the robotic device in order to improve the fidelity of the reconstructed signal. We test the proof of concept through extensive simulations. Secondly, a *lossless* rate control scheme, called *dynamic packetization module (DPM)*, applicable to both forward and backward (robot to human) channels. Via simulations and a real-time telehaptic experiment, we demonstrate that DPM preserves the fidelity of the reconstructed signal, even under heavy cross-traffic conditions.

Finally, we develop an analytical model for capturing the dynamics of interplay between telehaptic traffic and network cross-traffic. Our analysis applies to a broad class of telehaptic protocols including the ones that we have designed. The aim of the analysis is to develop a comprehensive understanding of the impact of network cross-traffic on telehaptic communication. This helps us to identify network configurations where QoS-compliant telehaptic communication is feasible.

Contents

List of Tables

List of Figures

Nomenclature

OH_h	Packet overhead size of HoIP
OH_d	Packet overhead size of data link layer
X	Amplitude of the current haptic sample
X_{prev}	Amplitude of the previous perceptually significant haptic sample
δ	Weber fraction
β	Level crossing threshold
d_{fwd}	End-to-end delay of the forward channel
d_{bwd}	End-to-end delay of the backward channel
$Y(t_{-x})$	Amplitude of x^{th} previous sample received with received timestamp t_{-x}
$Y(t)$	Amplitude of the current sample being displayed currently
$\underline{Y}(t_{-1})$	Lower limit of the linear extrapolator
$\overline{Y}(t_{-1})$	Upper limit of the linear extrapolator
T_o	Time-out interval
R_p	Telehaptic payload rate
R_{oh}	Peak overhead rate due to the network protocol stack
R_f	Overall telehaptic data rate on the forward channel
R_b	Overall telehaptic data rate on the backward channel

α	Weighting factor of the moving average filter
$d(\cdot)$	Instantaneous value of the piggybacked end-to-end delay
$d_{avg}(\cdot)$	Weighted average value of the piggybacked end-to-end delay
I_C	Network congestion trigger
I_S	Steady state trigger
p	Threshold of a static generic adaptive sampler
p_{curr}	Current threshold of the opportunistic adaptive sampler
p_{max}	Maximum permissible value of adaptive sampling threshold
γ	Opportunistic adaptive sampling parameter
K_p	Position gain of the PD controller
K_v	Velocity gain of the PD controller
μ	Capacity of the bottleneck link
τ	One-way propagation delay
R_{cbr}	Data rate of the CBR cross-traffic source
R_{vbr}	Data rate of the VBR cross-traffic source
R_{peak}	Peak data rate of the telehaptic stream
f_h	Peak sampling rate of the haptic signal
s_h	Peak size of a haptic sample
f_a	Peak sampling rate of the audio signal
s_a	Peak size of an audio sample
f_v	Peak sampling rate of the video signal
s_v	Peak size of a video sample

s_f	Peak size of a telehaptic fragment
s_m	Peak size of audio/video data in each telehaptic fragment
R_f	Telehaptic data rate on the forward channel
R_b	Telehaptic data rate on the backward channel
k	Number of telehaptic fragments merged in a packet
k_{max}	Maximum permissible value of k
R_k	Overhead rate due to the $k-$merge scheme
λ	Transmission delay of a telehaptic packet
k_{opt}	Optimal value of k for a given cross-traffic
d_{inc}	Time interval between delay build up and congestion detection
d_{hap}	Upper bound on the haptic delay
d_{aud}	Upper bound on the audio frame delay
d_{vid}	Upper bound on the video frame delay
N	Number of increasing/steady delay measurements for trigger generation
W	Value of the congestion window
S_{tcp}	Size of a TCP packet
W_{min}	Minimum value of the congestion window
B	Size of the queue at the ingress of bottleneck link
Q	Value of the queue occupancy
$Q(i)$	Value of the queue occupancy at the start of i^{th} slot
Q_{min}	Minimum value of the queue occupancy
c	Number of slots in a cycle during congestion avoidance

c_1	Total slots per cycle in which queue drains at the start of congestion avoidance
Q_{init}	Queue occupancy at the start of congestion avoidance
R	CBR traffic rate in the network
$Q_C(i)$	CBR component of queue at the start of i^{th} slot
$Q_T(i)$	TCP component of queue at the start of i^{th} slot
$D_C(i)$	Amount of CBR data drained in the i^{th} slot
$D_T(i)$	Amount of TCP data drained in the i^{th} slot
$RTT(i)$	Round trip time of the probing packet of i^{th} slot
$\Delta Q_C(i)$	Increment in CBR component of the queue between the start of i^{th} and $i+1^{\text{th}}$ slots
T	Time duration of the fast retransmit, fast recovery phase
d_{min}	Lower bound on the packet delay
d_{max}	Upper bound on the packet delay
n	Parameter of cumulative acknowledgement
T_{cbr}	Time interval between two successive CBR packets
x	Number of acknowledgement arrivals between two successive CBR packets
ΔQ	Maximum jump seen by the CBR packets
ν	Approximated packet jitter

Chapter 1

Introduction

The sensation of touch plays a crucial role in proper interpretation of physical objects [83]. Physical properties of an object like hardness, weight, texture, temperature can only be perceived through touch-based exploration of the object. The contact forces exerted by the object on the human body are fed back to the brain in the form of impulses. These impulses are then processed by the brain to create a perception of the object. The term *haptics* deals with the study of sensation and manipulation of physical objects through the modality of touch. The presence of the force feedback mechanism in humans largely simplifies the execution of everyday tasks like lifting an object, driving a car, playing a game of golf, etc.

Haptic technology makes use of this concept of force feedback, and explores the possibility of making remote/virtual objects palpable by augmenting the sensation of touch along with audio and video modalities. This unlocks new directions for creating platforms that enable highly immersive human interactions with the virtual/remote world. For example, through incorporation of the haptic feedback one can feel the intensity of a punch while participating in a virtual-reality boxing game. Research has shown that the addition of haptic modality to the existing audio-visual virtual-reality systems results in an enriched user experience [94], and also significant improvement in the accuracy of the task being performed [13]. A haptic device, such as Geomagic Phantom Omni [4], serves as the touch interface between the human and the virtual/remote world. It is an input-output device that captures the position, velocity and torque corresponding to the human movements in the physical world, and maps these into the virtual/remote world. Based on the nature of the interaction, the device outputs an appropriate force that is fed back to the human user, thus providing a feeling of touching the object.

Figure 1.1 demonstrates a human user performing a haptic activity in the virtual environ-

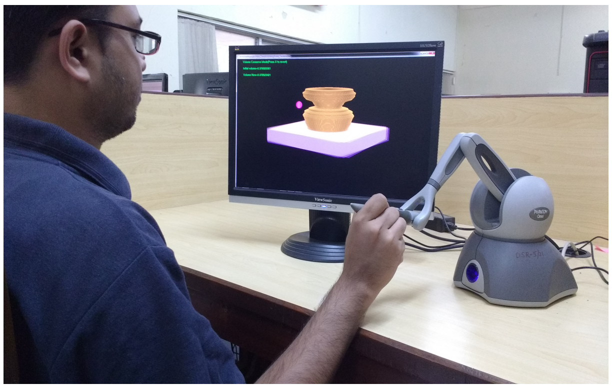


Figure 1.1: Demonstration of a haptic activity in which the user interacts with the virtual environment through Geomagic Phantom Omni haptic device.

ment. The user holds the stylus of the haptic device, and moves it in the physical space. The haptic interaction point (pink ball on the computer screen) follows the trajectory of the stylus based on the user movements. When it touches the virtual object, collision forces are generated which are fed to the user through the haptic device.

The field of haptics technology is broadly classified into three categories: *human haptics, machine haptics,* and *computer haptics*. Human haptics deals with the study of various aspects associated with the human haptic perception. Machine haptics deals with the design and development of high fidelity electro-mechanical devices that provide means to capture the human movements, translate them precisely in the virtual/remote world, and display the forces back to the human user. Computer haptics involves design of rendering algorithms to simulate highly immersive virtual world for a near-realistic interaction, swift computation of the contact forces in real-time, and communicating the haptic signals over a network in the case of a remote haptic interaction.

Haptics technology has paved way for a variety of promising and challenging applications. Common examples include medical training [12], medical diagnostics [86], prosthetic arm [56], digital heritage museums [67, 93], touch-enabled virtual reality gaming [71], online apparel shopping [76]. An exhaustive collection of the potential haptic applications is available in [30]. In this thesis, we focus primarily on the haptic interaction between a human user and a remote environment through a communication medium. This domain of remote interaction through force feedback is commonly referred to as *telehaptics*.

1.1 Telehaptics

The concept of telehaptics is inspired from the invention and widespread impact of the good old television and telephony systems, which enable human users to communicate with remote entities via audio and visual modalities. Telehaptic systems allow the users to carry out exploration and manipulation of geographically distant objects through the modality of touch [55, 77]. The central objective behind the design of such systems is to provide a setup through which users experience a feeling of being present in the remote environment, and directly interacting with the remote objects. Such systems are also referred to as *telepresence and teleaction (TPTA)* systems. Such highly sophisticated systems are a result of large-scale interdisciplinary research spanning the scientific and engineering domains of robotics, control theory, network communications, computer science, and psychophysics. The task being performed by the human user in telehaptics is usually referred to as *teleoperation*. In a TPTA system, multiple users, from distinct locations, may participate simultaneously to perform a collaborative task over a network. Popular example of telehaptic applications include touch-based immersive chat systems [62], telesurgery [10,63], remote disaster management, and distributed touch therapy [15]. A comprehensive survey of telehaptic applications can be found in [50]. For concreteness in exposition, in this thesis we restrict our focus to telehaptic applications involving a human user manipulating a remote environment, generally known as the *point-to-point teleoperation*.

A typical point-to-point teleoperation configuration comprises of a human user known as *operator*, a robotic telemanipulator known as *teleoperator*, a human-system interface (HSI) at the operator's end, and a communication medium for data exchange between the two ends. The teleoperator (TOP) is equipped with high precision actuators and sensors (for example, camera, microphone, and force sensors). The HSI provides the operator (OP) with a sophisticated multi-modal display system (for example, head-mounted display, monitor, headphones, haptic device) in order to create a high quality immersive interface to the remote environment. The OP transmits the current position, velocity and torque commands captured by the haptic device corresponding to the user's physical movements to the TOP. Upon reception, the TOP executes these commands in order to follow the trajectory of the OP as precisely as possible. Essentially, the OP and the TOP share a master-slave relationship between them. Any contact between the TOP and the remote objects generates collision forces that are captured by the sensors at the TOP. They are then transmitted to the OP in addition to the captured auditory and visual signals.

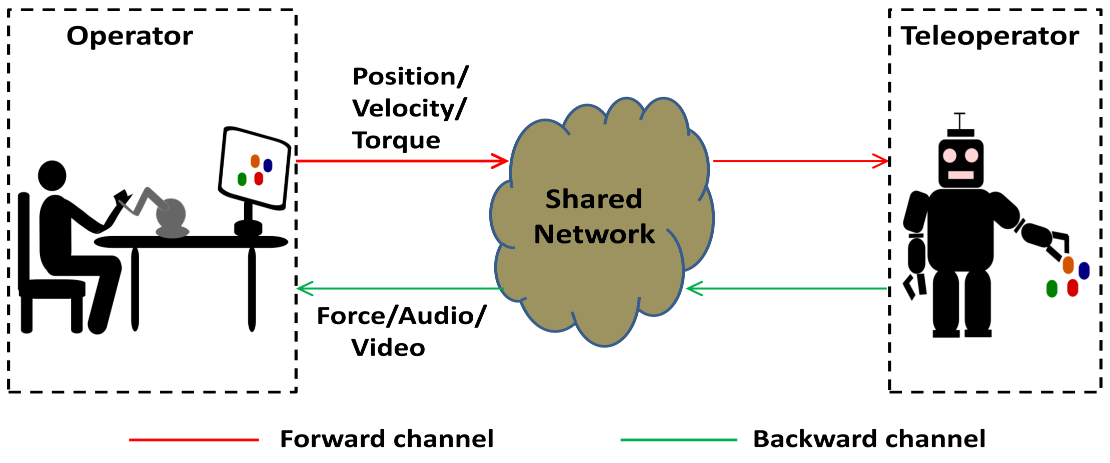


Figure 1.2: Schematic diagram representation of a typical telehaptic communication framework showing the operator and the teleoperator.

Further actions of the OP are driven by the displayed haptic-audio-visual feedback. The communication medium consists of a *forward channel* on which the OP transmits data to the TOP, and a *backward channel* on which the TOP transmits data to the OP. Thus, unlike communication involving only audio and video, the telehaptic communication is inherently bidirectional and asymmetric in nature. Figure 1.2 shows the ingredients of a typical point-to-point telehaptic communication framework.

In the next section, we describe a few challenges involved in fulfilling these telehaptic requirements, and then motivate the need for the design of the *Haptics over Internet Protocol (HoIP)* - a novel transport layer enhancement to the existing network protocol stack for enabling a smooth teleoperation.

1.2 Challenges in Telehaptic Communication

Telehaptic applications can include extremely sensitive and critical operations. Some examples include telemaintenance in a disaster management site, telesurgery in a military site, where either existing shared networks or quickly deployed ad-hoc networks need to be used for communication. Indeed, any real-time communication involving multimedia transmission is sensitive to various factors that are intrinsic to a network like delay, jitter and packet losses. The delay is a consequence of the physical distance between OP and TOP, and the queueing that occurs at the network buffers due to network congestion. Jitter - the variation in the inter-packet arrival delays, is caused due to the fluctuations in the network cross-traffic. Packet losses occur due to overflowing of the buffers during heavily congested network conditions.

The human perception manifests different levels of sensitivities for each of the multimedia types involved in the telehaptic communication viz. haptic, audio, and video. Therefore, the conditions on the network parameters for a smooth teleoperation are extremely media-specific. These conditions are collectively known as *Quality of Service* (QoS). Table 1.1 summarizes the QoS specifications for a telehaptic application [43,64,65,69,100]. For example, a haptic sample that is generated at the OP (respectively, TOP) should be delivered to the TOP (respectively, OP) within 30 ms. Similarly, audio and video signals have different QoS constraints.

	One-way delay (ms)	Jitter (ms)	Packet loss (%)
Haptic	30	10	10
Audio	150	30	1
Video	400	30	1

Table 1.1: Telehaptic QoS specifications for delay, jitter and packet losses.

Clearly, the haptic modality is the most sensitive to the network delay and the jitter. The bidirectionality nature of the telehaptic flows creates a global control loop involving the OP, communication network, and the TOP. For performing a seamless telehaptic activity, maintaining the stability of the control loop is crucial. In general, non-conformance to the above QoS requirements leads to a deteriorated perception of the multimedia by the human user [54]. Specifically, violation of the haptic QoS specifications causes destabilization of the global control loop [31], giving rise to transparency related issues. Such ill-effects of QoS violations could be potentially catastrophic in case of critical operation, like telesurgery. Of course, there are control theory techniques available that preserve the stability of the loop despite the delays being larger than 30 ms. However, in the context of haptics, the delay requirement is also crucial from the human perception standpoint. In other words, haptic delays larger than 30 ms cause deteriorated perception of the remote environment, and hence result in a substandard quality of teleoperation.

The stringent timing deadlines on the haptic data delivery, and the presence of the global control loop distinguish telehaptic applications from conventional audio-video streaming applications. In the literature, a class of audio-video communication protocols have been proposed. Since none of these protocols are tailor-made for haptic data, leveraging them for telehaptic communication would possibly lead to issues related to transparency and perception. For the purpose of illustration, we carry out a detailed performance evaluation of the Real-time Transport Protocol (RTP) - a state of the art multimedia communication protocol, and demonstrate its sluggish behavior from the telehaptics standpoint in Chapter 6. This motivates us to design communication protocols specifically with the requirements of telehaptic QoS in mind.

In order to maintain transparency in the interaction between the human and the remote environment, the standard update rate of a haptic signal is maintained at 1 kHz. Owing to the strict timing deadlines on the delivery of haptic data, the conventional approach is to transmit each haptic sample separately [9, 29]. This technique gives rise to a peak packet rate of 1000 per second.

On a dedicated network (like a network that directly connects the OP and the TOP, and is unaccessible to other traffic flows), the telehaptic application can utilize the network bandwidth completely. Hence, managing such high packet rate, and thereby conforming to the telehaptic QoS requirements may be feasible. In a practical scenario, it is not uncommon for the OP and TOP to be placed thousands of miles apart. Hence, laying dedicated networks for telehaptic communication over such long distances poses several practical challenges, and may not be economically viable. Furthermore, quickly deployed wireless ad-hoc networks, required to be shared amongst multiple traffic flows, for disaster management scenarios are heavily resource constrained. Hence, for a realistic telehaptic communication, one needs to work with existing shared networks, like the Internet.

A shared network simultaneously serves to carry data to and from several users, with each user starting and aborting the transmission session at random instants of time. Therefore, on a shared network the telehaptic flow has to compete with several coexisting flows for utilizing the network resources. The simultaneous utilization of a network by multiple flows, and the randomness involved in their arrivals result in a network load that is both unknown as well as time-varying. This leads to a nondeterministic network behavior. In such scenarios, the telehaptic application becomes extremely vulnerable to the QoS violations. Due to the limited capacity of the shared network, data transmission at the peak rate, as discussed in [9, 29], often causes congestion whose effects are extremely harmful to the telehaptic application.

The state of the art approach to curtail the telehaptic data rate is to perform *adaptive sampling* on the haptic signal [47]. Adaptive sampling exploits the perceptual limitation of the human haptic sensory system, of not perceiving certain haptic samples, to achieve lossy haptic signal compression. The *perceptual significance* of a haptic sample is dependent on the change in its magnitude with respect to a reference magnitude. If this change falls below a predefined threshold, then the current sample is not perceivable by the human users. The communication system can refrain from transmitting such perceptually *insignificant* samples, thereby reducing the telehaptic data rate. The work in [47] demonstrates a substantial reduction of up to 90% in the telehaptic data rate averaged over the entire duration of the session.

Both telehaptic transmission schemes discussed so far, viz. peak rate and adaptive sampling, operate in a *network-oblivious* manner. Irrespective of the network state, the peak rate technique transmits at an excessively high rate, and hence can be problematic during network congestion. The adaptive sampling rate depends on the nature of the haptic signal, thereby making it completely insensitive to the network changes. It is well known that rapid operator movements or interaction with rigid objects can cause every haptic sample to be perceptually significant. This gives rise to bursty haptic traffic. During network congestion, such behavior further aggravates the network condition. On the other hand, adaptive sampling on the forward channel cuts down the fidelity of the velocity signal, and hence could be highly suboptimal since the TOP (a robotic telemanipulator) can execute even the slightest of the signal changes. This could result in an erroneous signal reconstruction at the TOP, and consequently an imprecise teleoperation. To summarize, the network-obliviousness makes the existing telehaptic communication techniques vulnerable to QoS violations. Hence, in order to ensure a seamless telehaptic activity it is crucial to carry out a *network-aware* communication, that tunes the data rate to match the available network capacity. Specifically, during congestion the telehaptic rate needs to be reduced to prevent catastrophic effects on the application. On the other hand, when the network has spare bandwidth the data rate can be increased to transmit a higher resolution telehaptic signal.

Before designing the network-aware protocol, it is crucial to be able to detect network changes in real-time. The time-varying behavior of the network load makes the network estimation more challenging. The congestion is characterized by a persistent build up in the queueing delays. Once the queues are full, the network starts to drop packets. Typically, in the networking literature, network delays and/or packet losses are used as network condition indicators to achieve congestion control [17, 32, 42, 108]. Note that the increasing network delays act as precursor to the packet losses. During network congestion the delays build up in the timescale of milliseconds. Due to the highly sensitive nature of the telehaptic data, using packet losses for network estimation leads to a sluggish rate control. Hence, we resort to a delay-based network state estimation.

This motivates us to design a telehaptic communication protocol that is tuned for handling the requirements of a telehaptic application, while manifesting a network-aware behavior on a shared network. In this thesis, we present Haptics over Internet Protocol (HoIP) - a transport layer enhancement to User Datagram Protocol (UDP) layer, for achieving a seamless telehaptic control of the remote environment.

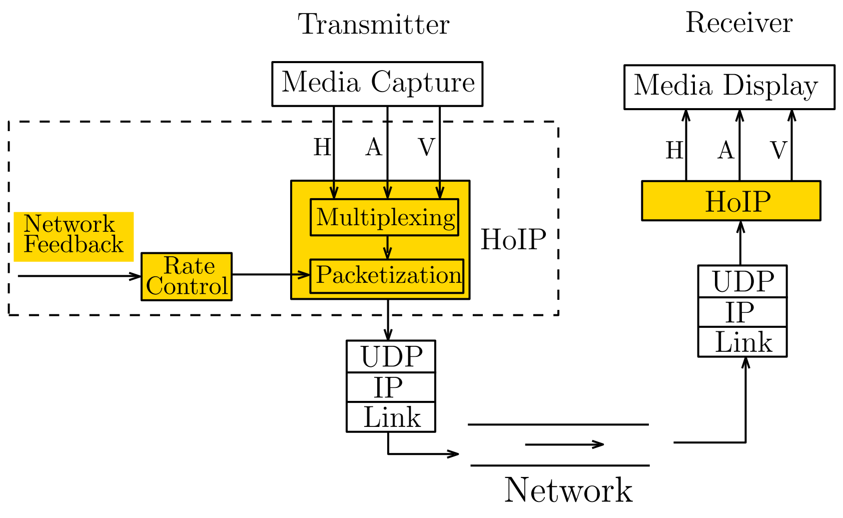


Figure 1.3: Generic block diagram representation of HoIP framework for forward and backward channels. Note that in case of forward channel only haptic media exists. Media notations: H - haptic, A - audio, V - video.

1.3 Contributions of the Thesis

The overall contributions of this thesis can be split into three parts as shown in Figure 1.3.

- In the first part, we design a basic HoIP framework necessary for multiplexing and packetization of the telehaptic data prior to the transmission. We take up the design of this framework separately for the two channels. While Chapter 3 deals with the forward channel, Chapter 5 handles the backward channel functionalities. The framework also incorporates a low-overhead, fine-grained congestion signal for swift network monitoring.

- In the second part, we design two network-aware telehaptic protocols: one lossy (Chapter 4) and the other lossless (Chapter 6) rate control schemes.

- In the third part (Chapter 7), we consider two classes of telehaptic protocols (that includes the ones we design). We identify sufficiency conditions for telehaptic QoS conformance on a shared network.

In the following discussion, we provide a detailed summary of the contributions of each of the chapters in the thesis.

1. **Packetization framework for forward channel:** We start by designing a basic packetization framework for the forward channel telehaptic communication in Chapter 3. The framework renders support for the transmission of haptic data using the standard 1 kHz packetization technique, as well as the adaptive sampling mechanism. Additionally, the framework explores a low overhead, swift network feedback mechanism for enabling efficient network monitoring. Further, it incorporates a *time-out* mechanism for minimizing the reconstruction errors due to packet losses. Through simulations, we demonstrate that the overall processing delay per sample arising due to the framework is substantially low, in the order of sub-millisecond. Therefore, this framework supports processing of the haptic signal at 1 kHz. Results of this work are published in [36].

2. **Lossy rate control using adaptive sampling for forward channel:** As the forward channel directly feeds the high-precision robotic telemanipulator, employing lossy adaptive sampling schemes causes loss of fidelity. Based on this observation, we devise a network-aware, *opportunistic* adaptive sampling technique with a primary objective of utilizing the spare network bandwidth for enhancing the quality of the reconstructed velocity signal at the TOP in Chapter 4. Through simulations, we show that the opportunistic sampler results in a substantial improvement in the quality of the reconstructed velocity signal at the TOP. Further, our findings reveal that this rate control scheme is extremely friendly to constant bit rate (CBR) cross-traffic in the network. Results of this work are published in [38].

3. **Multiplexing-packetization framework for backward channel:** In Chapter 5, we design a multiplexing and packetization frameworks for efficient transmission of haptic-audio-video frames on the backward channel. The multiplexing framework addresses the transmission scheduling of media frames which play a pivotal role in satisfying the QoS of the participating multimedia types. The haptic samples get the highest transmission priority because of the strictest QoS constraints. For resolving the conflict between audio

and video frames, we design two selection rules: one based on *first come, first serve* principle, and the other based on the *QoS-aware* principle. Via simulations, we demonstrate that the latter technique outperforms the former in terms of the media jitter. The packetization framework is a variant of that proposed in Chapter 3. Results of this work are published in [35].

4. **Lossless rate control for forward and backward channel:** We identify a few crucial drawbacks of the adaptive sampling based transmission schemes. In order to overcome these drawbacks, we design *dynamic packetization module* (DPM) - a lossless rate control telehaptic communication scheme. DPM works on the principle of merging k telehaptic fragments in a telehaptic packet. k is used as the control parameter to dynamically adjust the telehaptic rate based on the fluctuating network conditions. Our simulation results suggest that in addition to performing swift congestion control, DPM significantly improves the quality of the reconstructed haptic signal due to its lossless transmission nature. Furthermore, we perform a real-time telehaptic task involving several human subjects. The qualitative results of this evaluation reveals that DPM introduces negligible perceivable artifacts even under heavily congested network scenarios. Results of this work are published in [39].

5. **Interplay between telehaptic flows and network cross-traffic:** In this work, we analyze the interplay between telehaptic flows and network cross-traffic flows by considering two broad classes of telehaptic protocols that subsume the ones that we have proposed in Chapter 4 and Chapter 6. We develop an analytical model for characterizing the haptic delays and jitter in presence of heterogeneous cross-traffic. We derive sufficiency conditions for telehaptic delay and QoS adherence. Additionally, we demonstrate through simulations that smaller sized packets result in near-zero losses. Further, we prove that the haptic delay QoS compliance implies compliance to audio and video delay QoS as well. Results of this work are presented in [37].

Chapter 2

Literature Review

In this chapter, we review the current literature related to the multimedia communication, involving a few non-telehaptic and telehaptic protocols. We begin by discussing certain non-telehaptic protocols that have been deemed appropriate for the telehaptic communication. We describe each of them briefly, and state the reasons for their inapplicability with respect to telehaptic communication. We then review a few recent communication protocols designed specifically for telehaptic applications. The literature review related to the interplay between telehaptic protocols and network cross-traffic is presented as a part of Chapter 7.

2.1 Non-Telehaptic Protocols

We begin our review by inspecting the generic transport layer protocols viz. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) in telehaptic communication. TCP [80] is an adaptive rate control protocol that provides a reliable mode of data delivery to the applications leveraging it. Applications like file transfer, email, web browsing, and even certain streaming applications like YouTube [3], Netflix [1] rely on TCP because of their requirement for lossless transmission. Naturally, such reliability comes at the cost of increased packet delays, as the lost packets need to be retransmitted until their successful arrival at the receiver. As we have already seen in Chapter 1, telehaptic applications prefer timeliness of data over reliability. Thus, the increased delay (owing to reliability) makes TCP unsuitable for teleoperation.

On the other hand, UDP [79] provides a best-effort type of service for packet delivery in order to achieve a low latency communication. To this end, UDP guarantees neither the delivery, nor the orderliness in the delivery of packets. The applications that use UDP need to implement these functionalities, if required. A large number of teleconferencing applications, for example, Skype [2] and Google Hangout [7], are UDP-based. Therefore, due to its low latency nature, UDP is ideally suited for telehaptic communication. However, it has to be noted that UDP flows are inherently network-oblivious in nature. Thus, using plain-UDP for real-time communication could result in severe QoS violations during congested periods. The transport layer protocol that we propose in this thesis are built on top of the UDP layer.

Several transport/application layer protocols have been proposed in the literature for collaborative virtual environments (CVE) or interactive applications. Synchronous Collaboration Transport Protocol (SCTP) is a transport layer protocol proposed in [92] for CVE applications. The three primary objectives of SCTP design are scalability, low latency and reliability. For scalability it makes use of Internet Protocol (IP) multicast. To achieve low latency and reliability, the CVE updates are categorized as *key* and *normal* updates. Key updates are sent reliably (through acknowledgements and retransmissions), whereas normal messages are sent only once. The scalability is achieved through multicasting the updates. The rate control of SCTP is based on the arrival of acknowledgements arising out of the key updates, which are typically sparsely placed in time. This could result in a sluggish congestion control.

Another transport protocol for interactive applications, called Interactive Real-time Protocol (IRTP), is designed in [78]. It is a hybrid scheme that switches between TCP and UDP for transmission. The *crucial packets* need to be reliably transmitted, and hence use TCP. The *real-time packets* need timeliness, and hence use UDP. Acknowledgments are sent corresponding to packet arrivals, irrespective of their type. If an acknowledgement corresponding to a crucial packet is missing, then the protocol resends the lost packet. On the other hand, if an acknowledgement corresponding to a real-time packet is missing, then the protocol makes a packet loss report without resending the lost packet. Congestion control is based on the packet losses. As discussed in the Chapter 1, congestion control based on losses is too slow to cope with the requirements of telehaptic applications.

Some researchers in the haptics community [95] propose leveraging the state of the art streaming media protocol called Real-time Transport Protocol (RTP) [53] for network state estimation in telehaptic communication, and then subsequently perform telehaptic rate adaptation. 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 once in every 500 ms [101]. Each of these reports carries the average of the end-to-end (E2E) delays encountered during that 500 ms window. On a real-world network, network changes can occur over a timescale of tens of milliseconds. Such sparse feedback is insufficient for telehaptic applications, which are extremely sensitive to network changes. Further, the average of the instantaneous E2E delays cannot capture the finer changes in the network. Hence, we believe that RTP does not provide communication solutions to telehaptic challenges. We demonstrate the sluggish behavior of RTP in Chapter 6.

The work in [66] proposes RTP/I - RTP for distributed interactive media. The protocol operation is largely based on RTP [53] (discussed previously). Hence, the drawbacks of RTP apply here as well. This protocol attempts to build a common application level platform for a wide range of distributed interactive media. This results in a large application layer overhead (28 bytes per packet) which is not suited for telehaptic applications.

Thus, we conclude that the state of the art protocols designed for CVE do not cater to the requirements of telehaptic applications.

2.2 Telehaptic Protocols

Several protocols have been proposed specifically for serving the telehaptic applications. These protocols can be broadly classified into two types based on their sensitivity to the network changes: network-oblivious and network-aware. The network-oblivious protocols are not equipped with rate control mechanisms to dynamically adapt the telehaptic transmission rate according to the varying network conditions. The network-aware protocols tune their transmission rates based on the perceived network changes. We discuss all of them briefly in the following.

2.2.1 Network-Oblivious

An initial setup for networked telehaptic communication system is presented in [44]. The authors of this work propose using standard statistical techniques like Differential Pulse Code Modulation (DPCM) or predictive DPCM for encoding the haptic signals to reduce the telehaptic data rate for transmission over a network. Such statistical techniques process blocks of data and hence add significant processing delay to the haptic samples. Hence, they cannot be applied to a delay-sensitive telehaptic application [14].

Smoothed Synchronous Collaboration Transport Protocol (s-SCTP) [27] borrows the core idea of differential reliability from SCTP [92] (discussed previously), except for the inclusion of a buffer at the receiver for smoothing the jitter. Every update is placed in the buffer as soon as it is received. The receiver periodically checks the buffer for the updates that have been sent in each window of a predefined length. In case an update is found, it is forwarded to the application. The authors claim that this minimizes the jitter since the updates are extracted from the buffer at regular intervals of time.

The work in [102] proposes RTNP - Real-Time Network Protocol, for minimizing the delay and jitter suffered by the haptic packets on a network. The idea is to program the transport layer to assign the top priority to packets coming from the telehaptic application. On generation of a telehaptic packet, the transport layer issues an interrupt that cuts the incoming flow from other (non-telehaptic) applications. In this manner the telehaptic packets are packetized as soon as they are generated, leading to low delay and jitter.

ALPHAN - Application Layer Protocol for HAptic Networking, proposed in [9], implements haptic and graphic data communication on the backward channel. The protocol uses the idea of *key* and *non-key* updates for both media types, similar to SCTP [92]. Further, the protocol uses separate transmission buffers for haptic and graphic updates, thereby enabling a critical update to be sent ahead of the others. The protocol transmits separate packets for haptic and graphic updates, resulting in an increased peak telehaptic rate.

A hybrid multicast transport protocol for CVE with networked haptic is presented in [16]. The protocol bunches the receivers into groups based on their geographical locations, and the messages are multicast to these groups. This reduces the number of update messages to be sent. Again, this protocol leverages the concept of differential reliability based on key and normal updates from SCTP [92], as described previously.

The work in [29] proposes AdMux - Adaptive Multiplexer for haptic-audio-visual communication, a statistical multiplexing algorithm for the backward channel transmission. This work develops a mathematical model for scheduling the telehaptic packet transmissions, based on the media type and the network changes. However, it is to be noted that this protocol is adaptive to network variations only in terms of scheduling the packets, and not in terms of the rate control. The transmission is carried out based on the priority assigned to the media types by the multiplexing framework. This scheme tends to give a higher priority to the media type that was transmitted in the previous slot. This could be problematic particularly when audio/video is chosen for transmission, as the haptic sample has to wait in the buffer until it gets its turn. Hence, this scheme may not always guarantee QoS conformance, and this has been reported by the authors themselves.

An elegant approach to curtailing the data rate, called *adaptive sampling*, has been explored extensively by a large number of researchers [14, 18, 25, 26, 45–49, 59, 87, 96, 107, 111, 113]. In principle, adaptive sampling 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 *perceptually insignificant* samples, thereby reducing the telehaptic data rate substantially, without hampering the human perception. The adaptive sampling is applied on velocity signal on the forward channel, and force signal on the backward channel. A variety of adaptive sampling schemes have been proposed to determine the perceptual significance of haptic samples. Among them are two popular forms.

- Weber sampling - flags a haptic sample as perceptually significant if the percentage change in its magnitude with respect to a reference exceeds a predefined threshold; see, for example, [47].
- Level crossing - flags a haptic sample as perceptually significant if the absolute difference in its magnitude with respect to a reference exceeds a predefined threshold [14].

A comprehensive summary of the haptic compression schemes available in the literature is reported in [97]. Several researchers have also considered Weber sampler for usage in a variety of telehaptic applications, like multiuser collaborative telehaptic tasks [89–91], and replay of recorded telemanipulation sessions [19].

Through experimental evaluation, it has been demonstrated that the Weber sampler yields a significant reduction in average data rate of approximately 75% and 90% on the forward and the backward channels, respectively. Essentially, the instantaneous data rates are averaged over the entire duration of the experiment to arrive at these numbers. Level crossing produces a similar order of haptic compression. Despite their considerable compression, adaptive sampling schemes have two major shortcomings.

- The instantaneous transmission rate depends on the type of haptic interaction. For example, rapid user movements produce fast varying velocity signals which gives rise to a large instantaneous data rate, sometimes close to the peak rate (far more than the average rate), as we demonstrate in this thesis. Note that the adaptive sampling is only *signal-aware*, but *network-oblivious*. In such cases, one needs to provision the network for the peak rate for avoiding QoS violations. This does not provide any real economies in terms of the network bandwidth requirement.

- Critical operation, like telesurgery, necessitates accurate replication of the surgeon's hand movements at the teleoperator (TOP). In such scenarios, a minor loss of precision due to adaptive sampling on the forward channel could result in potentially irreparable damage. Also, a TOP, such as a robotic device, could practically sense all haptic samples; in such cases, adaptive sampling discards a lot of useful samples potentially causing loss of fidelity.

The authors in [24] propose a *visual-haptic multiplexing* technique for telehaptic applications using Weber sampling for haptic rate reduction. The multiplexer employs a buffer (of size 15 ms) to intelligently bunch the haptic and video data in a packet. The perceptually significant haptic samples are packetized with video data of worth 1 ms. On the other hand, when a series of perceptually insignificant samples are detected, the multiplexer transmits a large chunk of video frame, not exceeding data of worth 15 ms, into a single packet. This reduces the bandwidth requirement considerably relative to transmitting the video data alone in chunks of worth 1 ms. However, the drawbacks of adaptive sampling mentioned previously apply to this protocol as well. Further, the usage of buffer prior to multiplexing adds a substantial amount of latency that can cause severe haptic delay QoS violations. We present a detailed performance evaluation of this protocol in Chapter 6. The protocol proposed in [73] also depends heavily on the Weber sampling of haptic signals.

The work in [33] is the first to explore the possibility of merging multiple haptic samples in a packet to reduce the telehaptic data rate. This work demonstrates substantial data rate reduction in a lossless manner by bunching a fixed number of haptic samples in every packet. Specifically, the protocol generates a packet once in every 8 ms (i.e. eight haptic samples per packet), irrespective of the network conditions. Note that this implies unnecessary packetization delay even when the network is uncongested. The authors show in a particular setting that such a packetization scheme results in a satisfactory user perception. On the contrary, we demonstrate in this thesis (see Chapter 6) that the packetization intervals greater than 4 ms result primarily larger E2E delays, without any substantial reduction in the telehaptic data rate.

In contrast to the above discussed network-oblivious telehaptic protocols, we propose a network-aware telehaptic protocol in this thesis for a seamless telehaptic activity.

2.2.2 Network-Aware

The literature also provides a few works that have considered network-aware telehaptic communication. In [20], the authors propose STRON - Supermedia TRansport for teleoperations over Overlay Networks. The supermedia refers to the media types that are involved in a teleoperation. The protocol implements forward error correction for reducing the E2E delays. The protocol dynamically adjusts the network routes and encoding redundancy to meet the QoS of supermedia streams.

In [60, 61], the authors propose a network adaptation scheme for merging haptic samples based on the packet losses arising out of congestion. Such a scheme is reactive to network congestion, in the sense that the data rate reduction process is initiated only after the network buffers start to overflow. It should be noted that the packet losses during congestion are always preceded by an increasing delay, as discussed in Chapter 1. Hence, by the time packet losses are detected, the delays would have substantially overshot the haptic QoS deadline. Clearly, such a loss-based congestion control mechanism is not suitable for highly delay-sensitive telehaptic applications. Further, we note that the references [60, 61] do not provide much details about the rate adaptation mechanism itself.

The authors in [109] propose the first delay-sensitive haptic communication protocol named Efficient Transport Protocol (ETP). ETP detects congestion based on the round-trip-time (RTT) measurements. Once congestion is detected, ETP reduces the telehaptic data rate by increasing the interpacket gap, i.e., by downsampling the haptic signal, and vice-versa. However, there are two crucial drawbacks of ETP. First, using RTT may result in improper network state estimation, especially in asymmetric network conditions which is typical of any real-world network. Second, the protocol cuts down the fidelity of the haptic signal (due to downsampling) to control the data rate during congestion. This could result in loss of fidelity which is unfavorable for any critical teleoperation (similar to the adaptive sampling technique).

The authors in [57] propose NAFCAH: Network Adaptive Flow Control Algorithm for Haptic data. NAFCAH performs rate control by adapting the number of haptic samples to be merged into a packet on the forward channel based on the network conditions detected through RTT measurements. Since it uses RTT as the congestion indicator, it has similar drawback as that of ETP. Further, when congestion is detected, NAFCAH decreases its transmission rate gradually. This results in an extremely sluggish response to congestion causing slow flushing of the network buffers. This increases the risk of QoS violations. We demonstrate the performance implications of the design principles of NAFCAH in Chapter 6.

To summarize, the literature provides network-aware telehaptic protocols that either wrongly estimate the network state or are sluggish in response or cut down the fidelity of the haptic signal for rate controlling the congestion. Moreover, to the best of our knowledge no telehaptic protocol evaluation has considered the traffic setting that resembles the traffic in a real-world network. In this thesis, we attempt to address these issues through a novel transport layer protocol for telehaptic communication called *Haptics over Internet Protocol (HoIP)*, and its analysis in a real-world network traffic setting.

Chapter 3

A Packetization Framework for Forward Channel

3.1 Introduction

The forward and the backward channels in a telehaptic communication carry asymmetric telehaptic data, as described in Chapter 1. While the forward channel transmits haptic data only, the backward channel deals with heterogeneous data types viz. haptic, audio and video. Hence, we take up the design of Haptics over Internet Protocol (HoIP) for the two channels separately. We begin by designing the packetization framework of HoIP for forward channel in this chap-

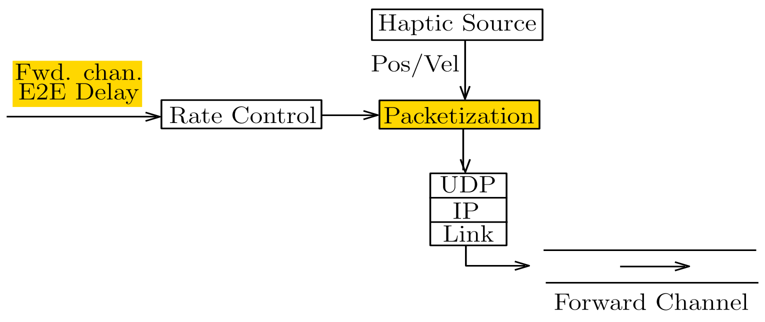


Figure 3.1: Architecture of Haptics over Internet Protocol (HoIP) for the forward channel, focusing on the packetization aspect which forms the subject of this chapter.

ter, as highlighted in Figure 3.1. This protocol mainly deals with packetization and feeding the congestion signal to the rate control scheme. The salient features of the protocol are the following:

- large flexibility for using a wide variety of lossless and lossy haptic sampling strategies at the operator (OP)

- extrapolation techniques for haptic signal reconstruction at the OP
- provision of a swift, low overhead, fine-grained network congestion signal at the teleoperator (TOP) for network monitoring at the OP
- time-out mechanism for reducing the errors due to packet losses in the network at the TOP
- plug-and-play, device independent platform for remote communication of telehaptic signals.

It is to be noted that the protocol defines the end-to-end treatment of the haptic signal at the OP, i.e., starting from sampling the velocity signal to reconstruction of the force signal. This also includes a specific functionality (incorporation of network congestion signal) to be implemented at the TOP, as we will describe in detail in this chapter.

HoIP is designed as a transport layer enhancement to User Datagram Protocol (UDP) in order to handle highly delay-sensitive haptic media. In this chapter, we present the architecture of the protocol in detail. It is also worth mentioning that the treatment of the haptic signal on the backward channel remains similar, as we will see in Chapter 5. We describe each of the HoIP features in detail. We present the HoIP header structure that conveys the metadata information to the TOP. Through simulations, we evaluate the HoIP performance in terms of its processing delays. We show that the worst case HoIP latency is less than 0.6 ms.

The remainder of the chapter is organized as follows. In Section 3.2, we provide a detailed description of the salient features of HoIP, and present the high level architecture of the protocol framework. In Section 3.3, we describe the performance evaluation of the protocol, and present the findings of the evaluation. We state our conclusions in Section 3.4.

3.2 Architecture

In this section, we describe the salient features of the protocol, and present its high level architecture.

3.2.1 HoIP in Protocol Stack

The proposed protocol is a transport layer enhancement to the UDP layer, and uses the underlying network protocol stack, as shown in Figure 3.2. The application layer runs routines required for handling haptic, audio and video capturing and display devices. In this chapter, we will refer to them collectively as *media interface*. HoIP is built independent of the media interface. Such decoupling of HoIP from the media interface increases the interoperability between them, and makes HoIP robust to the type of underlying telehaptic application. UDP is an unreliable mode of communication, and hence guarantees neither the successful delivery nor the in-order reception of the telehaptic packets. Hence, HoIP needs a means to monitor the arrival of haptic samples and validate their usefulness, since a delayed packet is often a useless packet in the context of telehaptic communication. In Section 3.2.6, we describe in detail the packaging of the necessary HoIP metadata required to identify the useful packets in a UDP-based telehaptic stream.

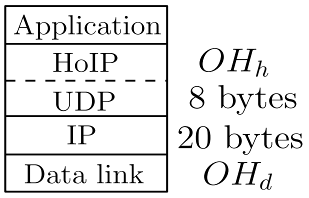


Figure 3.2: Network protocol layers along with their corresponding packet header sizes. OH_h and OH_d denote the HoIP and data link headers sizes, respectively. The application layer generates and displays the telehaptic media, hence there is no header associated with it.

3.2.2 Haptic Sampling Strategies

The initial operation involved in the processing of haptic signal is collecting the samples from the device drivers and forwarding them for packetization. This stage is crucial since it governs the overall telehaptic data rate. The conventional approach [9, 29] is to transmit all the generated haptic samples in order to perform lossless reconstruction at the receiver. Owing to the strict delay constraint, the haptic sample is transmitted immediately after its generation. This results in a peak telehaptic rate of 1000 packets/sec. Due to its fast packetization and *lossless* nature, this approach results in the transmission of a high fidelity signal to the teleoperator (TOP), leading to accurate replication of the human actions. While in areas served with high speed internet connections the resultant peak rate is manageable, many haptic applications are expected to operate in networks with limited bandwidth. Telesurgery in a disaster zone using quickly deployed networks is an example. Hence, such peak rate transmission approach may not be always feasible.

A well known approach for cutting down the telehaptic rate is to carry out haptic subsampling after packetization. This is achieved by transmitting only the perceptually significant features of the haptic signal - a scheme popularly known as *adaptive sampling* [14, 25, 26, 45, 47, 87, 107]. The idea is to filter out the samples that convey marginal or zero haptic information to the human operator. Only the significant haptic samples are transmitted, thereby creating a significant scope for reduction in transmission rate. This method provides a way to carry out telehaptic transmission in a *lossy* manner. The work in [45] has demonstrated that on the forward channel roughly 75% of the haptic samples are perceptually insignificant. The communication system can refrain from sending such samples. The perceptually significant velocity sample is transmitted along with the corresponding position vector. However, each of the perceptually significant samples is packetized separately.

Indeed, the choice of the sampling strategy is highly subjective, and is dependent on a few factors, such as the type of telehaptic interaction and the available network capacity. However, since we aim to design a generic, application-independent protocol we provide support for a wide variety of lossless, as well as lossy haptic sampling schemes. The user has the option of choosing the sampling technique before initiating the telehaptic session. The opening command prompt interface provides the necessary guidelines for the same.

Based on the literature, we implement the following two adaptive samplers in our protocol, and we also leave room for implementing other samplers in future. Let X_{prev} denote the immediate previous significant sample level (reference sample), and X denote the current sample level.

1. The *Weber sampler* flags X as perceptually significant only if
$$\left|\frac{X - X_{prev}}{X_{prev}}\right| \geq \delta. \tag{3.1}$$
2. The *level crossings sampler* flags X as significant only if
$$|X - X_{prev}| \geq \beta. \tag{3.2}$$

Here δ is the Weber threshold, and β is the level crossing threshold. As discussed in [45], the adaptive sampling is applied to the velocity signal on the forward channel, and to the force signal on the backward channel. It is worth noting that the adaptive sampling thresholds can vary significantly across users depending upon their level of perception [14]. Additionally, the user's perception level is a time varying entity, which is heavily dependent on various factors like the mood, fatigue level, and so on. Therefore, adaptive sampling threshold is a complex function of time and the haptic user. Hence, working with a fixed threshold [14, 25, 26, 45, 47, 87, 107] would result in a pessimal approach for efficiently capturing the significant haptic samples. Additionally, we discover that there is a substantial scope of performing dynamic telehaptic rate control simply by varying these thresholds in a network-aware manner (this forms the central theme of Chapter 4 of this thesis).

While we implement the above two adaptive sampling techniques, HoIP has enough flexibility to support the implementation of additional samplers, if needed.

3.2.3 In-header Delay Notification

As discussed in Chapter 1, due to their extreme sensitivity towards the network changes, the telehaptic applications need a swift, fine-grained network feedback mechanism. This helps in adapting the telehaptic communication to match the changing network state. Note that in order to perform congestion control on the forward channel, the OP must be aware of the forward channel end-to-end (E2E) delays. We exploit the bidirectional nature of the telehaptic traffic to convey the forward channel E2E delays to the OP, without transmitting specialized reports (unlike RTP). The idea is to piggyback the latest measured forward channel E2E delay on the outgoing packets of the backward channel. We term this technique as the *in-header delay notification* mechanism. This is demonstrated in Figure 3.3. d_{fwd} is measured at the TOP based on the time of packet arrival at the TOP. The in-header delay notification mechanism is more effective than existing network feedback techniques for three major reasons:

- The high packet rate of the telehaptic transmission enables the TOP to transmit a fine-grained forward channel E2E delay. As an example, a packet rate of 1 kHz means that the network changes can be measured with a resolution of 1 ms, and conveyed instantly. This enables the OP to swiftly adapt the telehaptic rate to the changing network conditions.

- The OP can perform rate adaptation purely based on d_{fwd}, rather than RTT or average delay estimates.

- Transmitting network feedback signal as a part of the packet header introduces a negligible packet overhead of 3 bytes per packet.

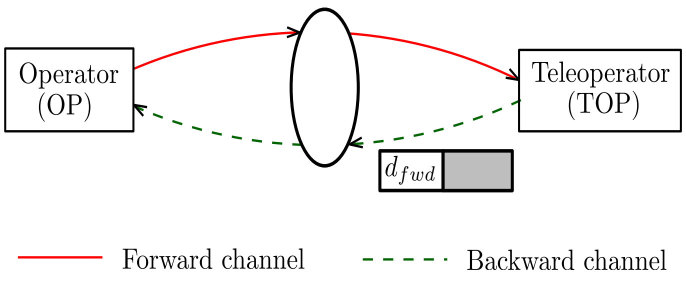


Figure 3.3: Demonstration of the in-header delay notification scheme for network feedback. d_{fwd} represents the forward channel E2E delay being piggybacked on the backward channel packets.

3.2.4 Haptic Signal Reconstruction

For the display of force signal, the OP needs to reconstruct a continuous-time haptic signal based on the discrete packets received from the TOP. The network cross-traffic introduces non-uniformity in the packet arrival process. In order to maintain the haptic display rate at 1 kHz, the OP needs to extrapolate the haptic signal based on the previous information, until a new packet is received. For this purpose, we implement two signal reconstruction techniques in our protocol. Let the signal level of the most recent samples received be $Y(t_{-1}), Y(t_{-2})$ with corresponding received time stamps t_{-1}, t_{-2}. Let $Y(t_1)$ be the signal level of the next sample to be received at time t_1. We implement the following extrapolators in HoIP.

1. Zero-order hold extrapolation: The signal displayed at the OP at time t,

$$Y(t) = Y(t_{-1}) \text{ for } t_{-1} \le t < t_1.$$

2. Linear extrapolation: The signal displayed at the OP depends on the sampling scheme used at the TOP.

   For Weber sampler, define

$$\underline{Y}(t_{-1}) = (1 - \delta \cdot \operatorname{sgn}(Y(t_{-1})))Y(t_{-1}), \quad \overline{Y}(t_{-1}) = (1 + \delta \cdot \operatorname{sgn}(Y(t_{-1})))Y(t_{-1}.$$

   where $\operatorname{sgn}(\cdot)$ denotes the signum function.

For level crossings, define

$$\underline{Y}(t_{-1}) = Y(t_{-1}) - \beta, \quad \overline{Y}(t_{-1}) = Y(t_{-1}) + \beta.$$

Let $L(a, b; x)$ be the limiter function such that

$$L(a, b; x) = \begin{cases} x, \text{ if } a \leq x \leq b \\ a, \text{ if } x < a \\ b, \text{ if } x > b \end{cases}$$

then

$$Y(t) = L\left(\underline{Y}(t_{-1}), \overline{Y}(t_{-1}); Y(t_{-1}) + \frac{(t - t_{-1})}{(t_{-1} - t_{-2})}(Y(t_{-1}) - (Y(t_{-2}))\right),$$

for $t_{-1} \leq t < t_1$.

The role of the limiter is to ensure that we use linear interpolation only to the extent that it does not change the perception as per the rule used for adaptive sampling. This ensures that if the reconstructed signal is passed through the same adaptive sampler, then we get the same samples as on the true signal.

While we implemented the above two extrapolators, HoIP has enough flexibility to support the implementation of additional extrapolators, if needed. Since, the reconstructed signal has a direct dependence on the adaptive sampling threshold used at the source, and also due to the fact that the thresholds are time-varying entities, we make provision for sending the threshold used at the TOP to the OP in the packet header (see field *Threshold* in Figure 3.4). This feature can also be utilized by the rate control techniques that use threshold as the control parameter (for example, see the rate control scheme in Chapter 4). In this thesis, we focus on the reconstruction of haptic signal only. However, one can employ standard reconstruction techniques for audio and video signals.

3.2.5 Time-out Mechanism for Adaptive Sampling

Suppose that the transmission is being carried out using an adaptive sampling scheme. Recall that using UDP communication inherently implies that the packet losses that are never recovered. Such packet losses make the sample references used by the extrapolator ($Y(t_{-1})$ and $Y(t_{-2})$)

at the OP highly unreliable. The packet losses can occur due to network errors or queue overflows. During rapid signal variations, the packets are transmitted in quick successions due to the large number of perceptually significant samples. Hence, it is more likely that the impact of packet losses are nullified by the successfully delivered packets which are transmitted shortly after the lost packets. On the other hand, if the haptic signal changes slowly, then the packets transmissions are sparse in time. In such conditions, the packet losses have a significant impact on the reconstructed signal, since they deprive the OP of the latest signal updates. This increases the drift between the original and the reconstructed signal.

One possible approach to minimize the error in the reconstructed signal is by ensuring that the time separation between successive packets does not exceed a certain threshold. In order to achieve this, we devise a *time-out mechanism* in which the TOP starts a timer after the latest packet transmission. If the time elapsed after the latest transmission exceeds the time-out interval, denoted by T_o, then the TOP reads a haptic sample directly from the device drivers and transmits it, irrespective of its perceptual significance. After the transmission, the timer is restarted. It should be noted that this gives rise to an additional telehaptic data rate relative to the standard adaptive sampling rate. The additional rate depends on the value of T_o.

The time-out mechanism has an added advantage that relates to the telehaptic rate control. Recall that the in-header delay notification mechanism, the TOP piggybacks d_{fwd} into the outgoing packets on the backward channel. Naturally, the forward channel state is sensed accurately when the packet rates are high both on the forward channel (leading to higher frequency of d_{fwd} measurements) as well as on the backward channel (leading to higher frequency of d_{fwd} transmissions). In other words, transmitting more packets on the backward channel conveys network feedback in a swift manner. This creates an opportunity to perform precise network estimation, and consequently an efficient rate control on the forward channel. We elaborate more on this aspect in Chapter 4.

3.2.6 Packet Structure

In order for the protocol to function properly, we need to transmit the haptic metadata to the TOP so that the haptic samples are decoded correctly. Typically, the metadata is conveyed in the form of packet headers. The packet structure of HoIP for the forward channel is as shown in Figure 3.4. Each row in the frame is composed of 32 bits. The top row indicates bit positions of the fields. The bits are numbered from 0 to 9 for the purpose of illustration. The HoIP packet starts from the second row. The packet consists of 10 bytes of header, followed by the haptic payload. We explain the meanings of each of the header fields in the following.

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Type		S			k		DI	Threshold																Notification Delay							
Notification Delay																Timestamp															
Timestamp																Haptic Payload															
....																															

Figure 3.4: Packet structure of HoIP on the forward channel.

- *Type* (2 bits): Indicates the type of haptic payload being carried by the packet. The typical forward channel payload types include position, velocity, rotation and rotational velocity. However, the payload types carried by the packet is extremely application-specific. For example, the torque information in some tasks may not have any importance at all. In others, the torque information may carry intermittent significance in which case it needs to be transmitted only when it is relevant.

- *S* (3 bits): Indicates the haptic sampling scheme employed for the current telehaptic session. 0: lossless (1 kHz), 1: Weber sampling, 2: level crossing. The other bits can be used for extending the protocol to accommodate other sampling schemes in future. Transmitting this field in every packet provides the flexibility of switching between different sampling schemes (if required) during the haptic interaction.

- k (2 bits): Indicates the number of haptic samples included in the current packet. This field provides support for a lossless, rate control telehaptic scheme, where the number of haptic samples bunched in a packet is used as the control parameter. We describe this technique in detail in Chapter 6.

- *DI* (1 bit): Denotes the *delay indicator* field. When the OP transmits multiple packets in between two packet arrivals, it piggybacks duplicate copies of d_{bwd}. In order to ensure that the TOP is aware of the delay type, the protocol uses *DI* to indicate the status of the delay piggybacked in the packet header. 0: the piggybacked delay is being transmitted for the first time, 1: the piggybacked delay has been transmitted in the previous packet. This field is used to prevent duplicate copies of the delay from impacting the network estimation process, as described in Chapters 4 and 6.

- *Threshold* (16 bits): Indicates the current threshold of adaptive sampling strategy employed. This field is included in the packet header if and only if the field S is non-zero. This threshold will be used by the TOP for the signal reconstruction.
- *Notification Delay* (24 bits): Indicates the latest backward channel E2E delay as measured by the OP. This field forms the basis of the rate adaptation on the backward channel by TOP, to be discussed in Chapter 6.
- *Timestamp* (32 bits): Indicates the generation time (in ms) of the haptic sample being carried in the payload. This field, along with the received timestamp at the TOP, is used to calculate the forward channel E2E delay, as discussed in Chapter 4. Further, this field can be used for packet reordering and scheduling the display of the haptic samples.

Note that the packet header structure on the backward channel is similar, except for addition of a few fields corresponding to audio and video frames, as we will see in Chapter 5.

3.2.7 Implementation Details

In order to prevent accumulation of samples at the source, the latency due to the protocol needs to be small so that the transmission/display of a haptic sample is complete before the subsequent sample is generated/received. Hence, the protocol latency per sample should be kept below 1 ms. The processing of the haptic samples involves a series of operations at the sender and the receiver. The sender operations include reading the samples from the media interface, subjecting them to the chosen sampling scheme, and appropriately packetizing the samples for transmission. The receiver operations include decoding the header, testing for the sample validity, extrapolating the signal, and finally feeding the signal to the media interface.

Even though HoIP is a transport layer enhancement, we code the protocol in the application layer for testing the proof of concept. However, it should be noted that this does not change the protocol behavior or the nature of our results. In order to cope with the stringent processing delay criteria (< 1 ms), we design a multithreaded architecture using C++ since it is most suited for real-time applications due to its faster execution of multithreaded programs. The different functionalities of the protocol are handled by multiple threads. In this section, we describe the individual components employed in the implementation of HoIP at the OP viz. *named pipes, sender* and *receiver modules*. While the named pipes are used for data exchange with the media interface, the sender and the receiver modules are associated with the transmission and reception of telehaptic packets, respectively. Figure 3.5 shows the high level architecture of HoIP at the OP. We implement the same protocol structure at the TOP as well for our simulations.

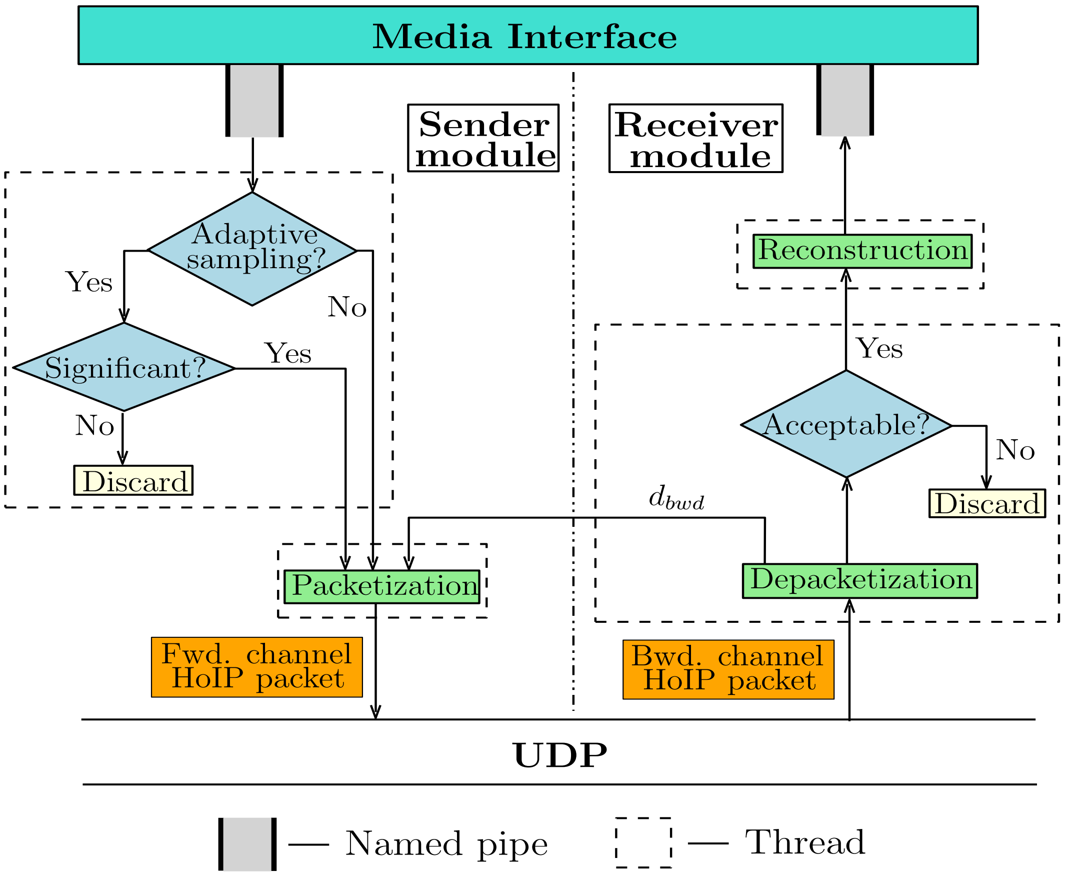


Figure 3.5: Architecture of HoIP for the forward channel telehaptic transmission showing the implementation of each of the features.

The implementation comprises primarily of three building blocks.

- **Named pipe**: A named pipe, described in [81], is characterized by *first in, first out* (FIFO) data flow, and is well suited for scenarios which require data sharing across applications running in the same machine. The named pipe acts as the bridge between the media interface and HoIP. For simplicity, we use two half-duplex named pipes to achieve bidirectional data flow between the two applications: one for reading the position-velocity samples from the interface for transmission, and the other for feeding the media frames into the interface for display purpose. The usage of the named pipes decouples our protocol from the media interface, enabling the two applications to be developed and upgraded asynchronously. Further, it makes the protocol easily compatible with the telehaptic application, and creates a *plug and play* type communication scheme, since the media interfacing software should only establish a connection with the pipe to start the communication. This also makes our protocol independent of the media capture and display devices being employed.

- **Sender module**: This module is responsible for reading the haptic samples from the media interface, subjecting them to the appropriate sampling scheme, packetizing the samples accordingly, and finally forwarding the HoIP packets to the lower layer for transmission. These operations are managed by multiple threads for minimizing the packetization delay, as shown in Figure 3.5. As soon as the haptic samples are extracted from the sender-side pipe, they are subjected to the chosen sampling mechanism described in Section 3.2.2. If the lossless scheme is chosen, then the samples are forwarded directly to the *packetization* block. If an adaptive sampling scheme is chosen, the perceptually significant samples are identified based on Equation (3.1) or (3.2). These samples are then sent to the packetization block for augmenting the HoIP header. On the other hand, the perceptually insignificant samples are useless as far as the transmission is concerned, and hence are discarded. The packetization block receives the samples to be transmitted, adds the packet header (described in Section 3.2.6), and sends the HoIP packets to the UDP layer. Each of the threads are connected through FIFO queues.

- **Receiver module**: This module is responsible for depacketization of the received packets, checking for their acceptability, reconstruction of the signal, and loading the samples into the pipe. Upon reception of packets, the *depacketization* block extracts the haptic, audio and video samples and their corresponding metadata. Later, the samples are scrutinized for their acceptability based on the generation and arrival timestamps. Specifically, the samples are unacceptable if they arrive later than the deadline or if previous samples are already displayed. The unacceptable samples are discarded. The acceptable samples are then forwarded to the *reconstruction block*, where they are utilized to predict the amplitudes of the samples that did not make it to the OP, as described in Section 3.2.4. The reconstruction block then loads the extrapolated samples into the pipe at the uniform rate of 1 kHz, which are then supplied to the media interface for the display purpose. The backward channel E2E delay (d_{bwd}) is computed on packet arrivals, and is conveyed to the packetization module, irrespective of the sample validity. This delay corresponds to the *Notification Delay* field of the packet header shown in Figure 3.4. It is worth noting that d_{bwd} will be utilized by the TOP for rate adaptation on the backward channel. This will be discussed in detail in Chapter 6.

For measuring the E2E delays, we assume that the OP and the TOP are time-synchronized throughout the telehaptic session. This is usually achieved by using the state of the art clock synchronization protocol called Network Time Protocol (NTP) [68]. Therefore, d_{bwd} can be computed as the difference between the generation timestamp (field *Timestamp* in Figure 3.4) and the arrival timestamp. It is to be noted that in this architecture even though the processing of a sample is carried out in a serial manner, the usage of multiple threads allows multiple samples to be processed at once.

3.3 Performance Evaluation

In this section, we describe the performance evaluation of the protocol, and present the findings of our experiments. The objective of this exercise is to get a sense of the protocol's suitability for time-critical telehaptic activity. Our experiments are conducted using NS3 - a discrete event network simulator [74]. We use haptic traces recorded during a real-world telehaptic activity. We begin by reporting the latency introduced by the sender and receiver modules of the protocol. Further, we present the comparison between the reconstructed haptic signals at the TOP due to the usage of Weber sampler and the reconstruction techniques mentioned in Section 3.2.4.

3.3.1 Setup

In this section, we describe the telehaptic data rate and the network settings used for our simulations.

Telehaptic Data Rate: Considering the packet header structure (described in Section 3.2.6), we see that for the peak rate transmission $OH_h = 8$ bytes since the packet does not include the field *Threshold* (2 bytes). Using Ethernet on the data link layer leads to $OH_d = 26$ bytes, and hence an overall packet header of size 62 bytes. This causes the instantaneous telehaptic overhead rate R_{oh} to take the peak value of 496 kbps. We consider a single point of contact device, like Geomagic Phantom Omni [4]. The position and velocity are 3D vectors, and each of the coordinates is encoded using four bytes. This gives rise to a payload rate of $R_p = 192$ kbps. Therefore, the overall telehaptic data rate ($R_p + R_{oh}$) on the forward channel is $R_f = 688$ kbps. On the other hand, using adaptive sampling techniques results in $R_{oh} \in$ [0, 496] kbps depending on the number of perceptually significant samples transmitted during that instant. In this case, $R_f \in$ [0, 688] kbps.

Network Settings: Since the protocol latency is robust to network settings, we consider a simple network topology wherein the OP and TOP are directly connected to each others, as shown in Figure 3.6. Since we are interested in the performance of the protocol alone, we consider a network with high bandwidth and negligible propagation delay. Due to the above network setting, the delay between sample generation and display is attributed solely to HoIP. The sender module latency is measured by the time between the generation and the injection of a packet into the UDP layer at the OP. The receiver module latency is measured by taking the difference between the display of a force sample at the OP and its corresponding arrival at the receiver module.

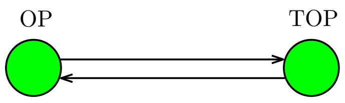
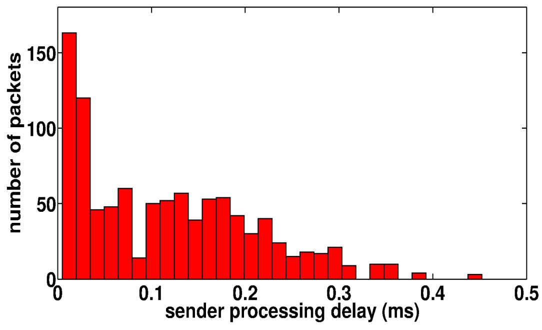


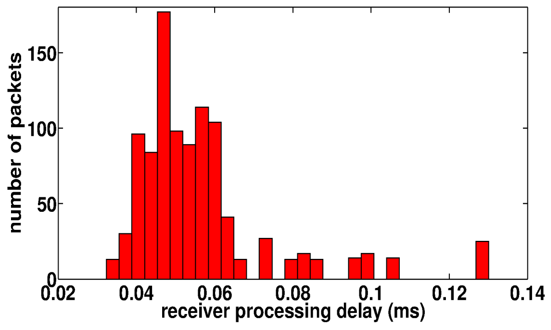
Figure 3.6: A simple network topology used for the simulations where the links have very high bandwidth and zero propagation delay.

3.3.2 Results

Protocol latency: We begin our evaluation by measuring the latency introduced by the protocol. Figures 3.7a and 3.7b show the histograms of the latency corresponding to the sender and the receiver modules, respectively, measured during the peak rate transmission. It can be observed that even the worst-case aggregate latency suffered by the haptic samples at the sender and receiver modules is less than 0.6 ms. This implies that HoIP is quite successful in satisfying the processing delay criteria (< 1 ms). It may also be noted that in both cases, the delay distributions do not show any signs of long tails suggesting that the proposed protocol would rarely violate the processing delay criteria. Note that even though we show the histogram only for



(a)



(b)

Figure 3.7: Histogram of the protocol latency due to the (a) sender module (b) receiver module.

1000 packets, the measured latency is consistent throughout. For brevity, we report the latency for peak rate transmission only. However, we observe through simulations that the latencies in case of adaptive sampling schemes are similar.

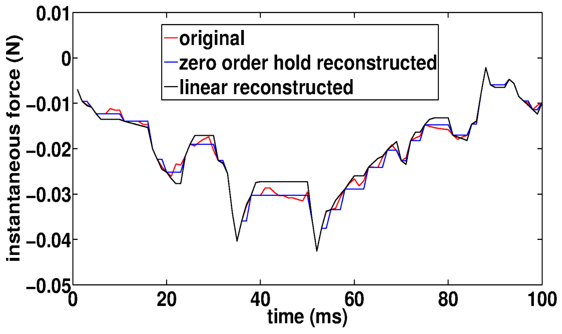


Figure 3.8: Plots of original and reconstructed force signal at the OP for Weber sampling with a typical $\delta = 0.1$.

Reconstructed Signal: We now move to the signal reconstruction at the OP as a consequence of employing adaptive sampling and signal reconstruction strategies. For brevity, we show the plots for one of the force traces. We subject this trace to Weber sampling with a typical threshold of $\delta = 0.1$ at the OP. We heuristically set the time-out interval $T_o = 100$ ms at the TOP. For the purpose of demonstration, we implement the extrapolators discussed in Section 3.2.4 at the OP. In separate experiments, we perform reconstruction at the OP using the zero-order hold and linear extrapolation techniques. We plot the original force signal along with the reconstructed signals in Figure 3.8. It can be seen that when the original signal changes slowly (around 40 - 50 ms), the Weber sampler filters these variations since they are not perceivable by the human user. In this region, the zero order hold technique produces a piecewise constant signal. On the other hand, the linear extrapolator predicts the current signal level that follows the trend exhibited by the past samples until the limits ($\underline{Y}(t_{-1})$ or $\overline{Y}(t_{-1})$) are reached. During rapid signal variations (around 85 - 100 ms), the reconstructed signal matches the original signal. This is because the Weber sampler detects every sample as perceptually significant.

3.4 Conclusions

In this chapter, we described the packetization framework of Haptics over Internet Protocol (HoIP) for forward channel telehaptic communication. We demonstrated that the protocol renders support for a wide variety of lossless as well as lossy haptic signal transmission schemes. We described the in-header delay notification mechanism for providing a low overhead, fine-grained, swift network feedback mechanism, thus rendering support for any rate adaptive feature to be incorporated. The time-out mechanism minimizes the possibility of signal drift at the receiver due to packet losses. The decoupling of the protocol from the haptic interface provides a plug and play type communication scheme that is independent and robust to nature of telehaptic application and the media devices being employed. This feature allows both the haptic interface and our protocol to be developed and upgraded asynchronously. Through simulations, we showed that indeed HoIP is successful in processing the haptic samples within 0.6 ms, thus adhering to the 1 ms criteria. In summary, we presented a flexible, low latency application layer protocol that can be leveraged by telehaptic applications for enhancing the state of the art in telehaptic communication.

Chapter 4

Lossy Telehaptic Rate Control on Forward Channel

4.1 Introduction

In the previous chapter, we devised the basic multiplexing and packetization framework of Haptics over Internet Protocol (HoIP) for the forward channel. The framework also extends support for network monitoring by providing a network feedback signal to the transmitter corresponding to the channel on which it transmits data. We now move to the telehaptic rate control aspect of HoIP that makes use of the packetization framework of Chapter 3. Figure 4.1 shows the HoIP design for the forward channel highlighting the rate control scheme, which forms the central theme of this chapter.

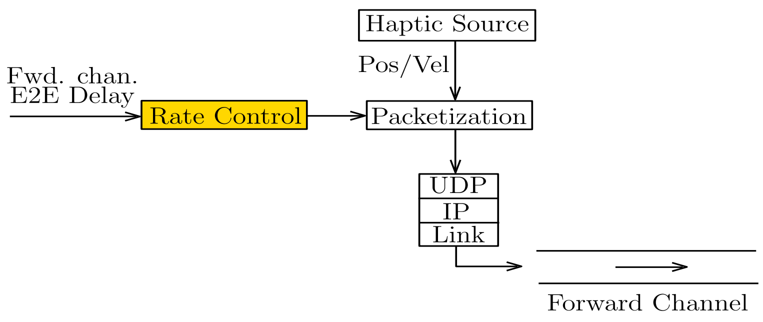


Figure 4.1: Architecture of Haptics over Internet Protocol (HoIP) for the forward channel, focusing on the rate control aspect which forms the subject of this chapter.

As discussed in Chapter 1, the transmission of haptic data at the peak rate (1000 packets/sec) is viable only when the network is uncongested. During congested periods, this approach induces large E2E delays and packet losses, leading to severe QoS violations. The direct consequence of this is the instability of telehaptic control loop, and the deteriorated perception giving rise to transparency issues. To counter these issues, researchers resorted to haptic data compression schemes, predominantly the deadband approach like adaptive sampling. The work in [47] demonstrated that the usage of perception-based deadband technique for adaptively sampling the haptic signal substantially reduces the data rate on the forward and the backward channels. The deadband scheme is applied on the velocity and force updates on the forward and the backward channels, respectively.

In critical applications like telesurgery and telemaintenance, the TOP is a robotic device that receives position-velocity updates to replicate the actions of the human as precisely as possible. In presence of high-precision actuators, the TOP can execute even the slightest of position-velocity changes causing every haptic sample to be practically significant. Thus, in such cases, the usage of a compression technique that is based on human perceptual limitations on the forward channel is suboptimal. In other words, the full precision of the TOP is never exploited leading to an imprecise teleoperation. This suboptimality is particularly significant when the network is uncongested, since transmitting haptic data at the peak rate would lead to a better reproduction of the human actions by the TOP. However, it is important to state that this argument is valid only for the forward channel. The deadband approach works well in the backward channel due to the human operator on the receiving end.

To summarize, the literature proposes two extreme techniques for telehaptic communication over a network. On one hand, transmitting haptic data at the peak rate works well when the network is uncongested, but is problematic when the network is congested. On the other hand, adaptive sampling works well when the network is congested, but is suboptimal when the network is uncongested, especially in applications where the TOP is a robotic telemanipulator. The key issue with both the aforementioned approaches is that they are insensitive to the state of congestion of the network. This motivates us to design *network-aware* schemes for telehaptic communication that adapt their transmission rate depending on the level of congestion in the network. We seek to devise an *opportunistic* scheme that detects the changing network state, and accordingly tunes the telehaptic data rate. In other words, the transmission rate due to the scheme should span the range between that of adaptive sampling (during congestion) and the peak rate sampling (during uncongested conditions) depending on the network state. For convenience, we term the standard, *network-oblivious* adaptive sampler as the *static adaptive* *sampler*.

In this chapter, we introduce a network-aware, opportunistic algorithm for adaptively sampling the velocity updates at the OP. In addition to satisfying the haptic QoS specifications, the goal of the opportunistic scheme is to maximize the transmission rate, subject to network capacity constraints. The assumption here is that at any point in time the network supports the instantaneous rate defined by the static adaptive sampler. By continuously monitoring the E2E delays on the forward channel, we estimate the corresponding congestion level. The idea behind the *opportunistic adaptive sampling* algorithm is to tune the adaptive sampling threshold to the current congestion level in the forward channel. The method is opportunistic as it constantly looks for an opportunity to pump more haptic packets without violating the QoS constraints.

To validate our approach, we record real-world haptic signals using a real-time telepottery experiment with several human subjects. We conduct extensive simulations on the recorded signals for performance evaluation of the algorithm, using a discrete event network simulator NS3 [74]. Our experiments demonstrate a substantial improvement in the packet rate as well as the quality of the reconstructed velocity signal at the TOP, compared to the static adaptive sampling. Additionally, we analyze the interplay between the telehaptic traffic and other network sources, and conclude that the opportunistic algorithm is friendly to exogenous cross-traffic flows, in terms of resource sharing.

The remainder of the chapter is organized as follows. In Section 4.2, we describe the working of the proposed opportunistic adaptive sampling mechanism. Section 4.3 describes the experimental setup designed for the evaluation of the proposed method. In Section 4.4 we present the findings of our evaluation. We state our conclusions in Section 4.5.

4.2 The Communication Framework

In this section, we describe in detail the working principle of the opportunistic adaptive sampling method. Figure 4.2 presents a schematic representation of the communication framework showing the opportunistic sampling module. We adopt the basic communication framework presented in Chapter 3, and build upon that to implement the opportunistic rate control module.

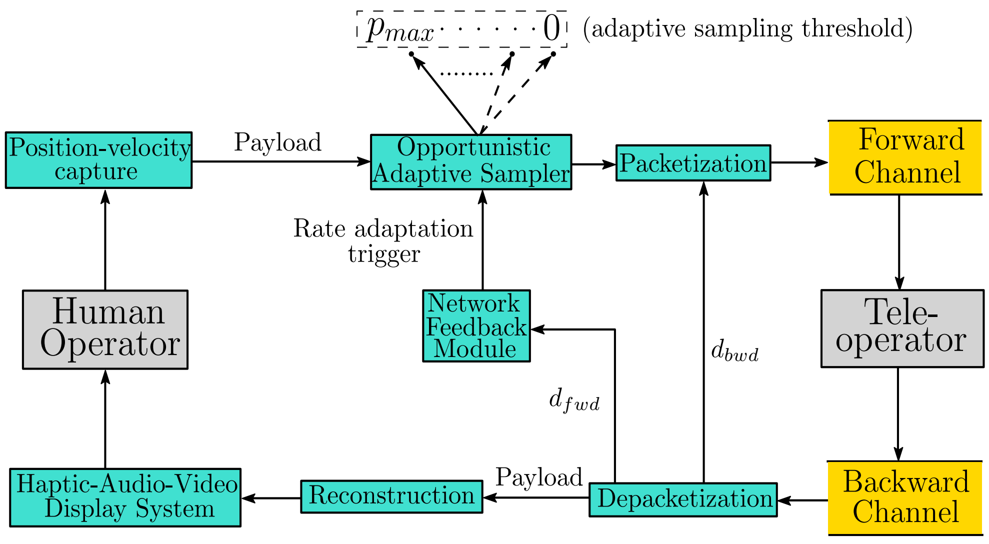


Figure 4.2: Block diagram representation of the communication framework showing the opportunistic sampling framework for the forward channel. d_{fwd} and d_{bwd} denote the forward and the backward channel E2E delays, respectively.

4.2.1 Network Analyzer

In Chapter 3, we mentioned the possibility of performing rate control by using the E2E delay encountered by the packets as the congestion indicator. To this end, we presented the in-header delay notification mechanism designed for swiftly conveying the network state through piggybacking the E2E delays on the packet headers in Chapter 3. In this section, we present a simple method to analyze the received piggybacked forward channel delays at the OP, and detect the current congestion level so as to enable congestion control.

Delay Analysis: The forward channel E2E delays, which are decoded from the packet headers, are fed to the *network analyzer* module, which carries out network monitoring. The trends followed by the E2E delays precisely suggest network congestion or underutilization. The E2E delays build up continuously during congestion, and stay roughly steady during uncongested network conditions. In order to filter out measurement noise, we use an exponentially weighted moving average filter on the E2E delays defined by

$$d_{avg}(m) = (1 - w) * d_{avg}(m - 1) + w * d(m), \tag{4.1}$$

where $0 < w < 1$. Here, $d(m)$ and $d_{avg}(m)$ denote the m^{th} E2E delay and m^{th} weighted average delay measurement, respectively.

Suppose that the TOP carries out peak rate transmission. In this scenario, duplicate copies of forward channel E2E delays reach the OP. This can happen if the TOP makes multiple packet transmissions between successive receptions. To avoid the same delay measurement from resulting in multiple updates in Equation (4.1), we utilize the packet header field named *DI* referring to delay indicator, introduced in Chapter 3. This field is set to 1 in case of a packet carrying a duplicate delay, and 0 in case of a packet carrying an original delay. Based on the value in this field, the OP can decide whether or not to run the update equation.

Generation of Rate Adaptation Triggers: Based on the trend observed in the computed average delays, the network analyzer generates two triggers which steer the rate adaptation process. The trigger I_C signals that the channel is getting congested; this causes the opportunistic sampling algorithm to reduce the telehaptic data rate, if possible. The trigger I_S signals that the channel delays are steady; this causes the opportunistic sampling algorithm to probe if the channel has spare capacity by increasing the telehaptic data rate, if possible. The rate adaptation mechanism is described in Section 4.2.2.

We use the following simple rule for trigger generation. If $d_{avg}(\cdot)$ shows N continuous increasing measurements, the congestion trigger I_C is generated. The steady state trigger I_S is generated if the most recent N entries in $d_{avg}(\cdot)$ satisfy two conditions:

- they exhibit neither an increasing nor a decreasing trend
- the latter $N-1$ entries stay within a specified tolerance interval (say 10%) around the first.

The filter described by Equation (4.1) is reset after each update. The choice of N plays a crucial role in the rate adaptation. A small value of N makes the network state estimation process more prone to noisy measurements of the E2E delays. On the other hand, a large value of N results in a sluggish response to the network changes. In our simulations, we empirically set $N = 8$. It is also worth mentioning that since the generation of triggers is based on a trend of the E2E delays, the above network state estimation scheme remains robust to the time synchronization errors of Network Time Protocol (NTP).

4.2.2 Opportunistic Adaptive Sampler

The objective of the *opportunistic adaptive sampler* is to improve the fidelity of the transmitted haptic signal whenever the network supports. The key idea is to increase the number of transmitted haptic samples so as to utilize the spare network bandwidth during uncongested conditions. This leads to a more precise replication of the user action relative to the case of the static adaptive sampler. The opportunistic sampler uses the adaptive sampling threshold as the control parameter to vary the fidelity of the haptic signal. Recall that this threshold defines the width of the perceptual deadband. A larger threshold corresponds to a wider deadband, resulting in a lower data rate, and vice-versa. The opportunistic sampler exploits this principle and dynamically tunes the threshold as a function of the current level of congestion in the network.

Let us denote the threshold of a generic adaptive sampler by p. Let p_{max} denote the maximum permissible value of p, beyond which the human perception is impaired. Depending on the congestion level, the opportunistic sampler varies p over the range $[0, p_{max}]$. Note that $p = 0$ corresponds to the peak rate, and $p = p_{max}$ corresponds to the minimal rate given by the static adaptive sampler. The opportunistic adaptive sampler updates the current adaptive sampling threshold, denoted by p_{curr}, each time a trigger is generated. When I_C is generated, p_{curr} needs to be increased so as to carry out congestion control. When I_S is generated, p_{curr} needs to be reduced so that the additional haptic packets are transmitted only to the extent of not overwhelming the network.

For efficient rate control, we resort to the classical additive-increase/multiplicative-decrease (AIMD) approach [23] proposed in the network congestion control literature. According to the AIMD principle, during network congestion the data rate is reduced in a multiplicative fashion to result in a swift congestion control. During uncongested conditions, the data rate is increased in an additive (linear) manner to prevent abruptly overshooting the channel capacity. We vary p_{curr} as shown in Figure 4.3, so that the telehaptic rate control is in line with the AIMD principle.

If I_C is generated, then the opportunistic adaptive sampler sets $p_{curr} = p_{max}$ in a single step. In other words, when congestion is detected, the algorithm aggressively cuts its data rate to the lowest level permissible. Indeed, the trigger I_C suggests that the buffers at the intermediate routers on the forward channel are fast filling. Thus, a rapid transmission rate reduction enables the network buffers to get flushed quickly resulting in congestion control in the shortest possible time. Such behavior is advisable on a shared network since the aggressive back off strategy minimizes the chances of haptic QoS violations. Further, the quick congestion control makes the haptic source friendly to the exogenous cross-traffic sources.

If I_S is generated, then the opportunistic adaptive sampler reduces p_{curr} by γ, so long as

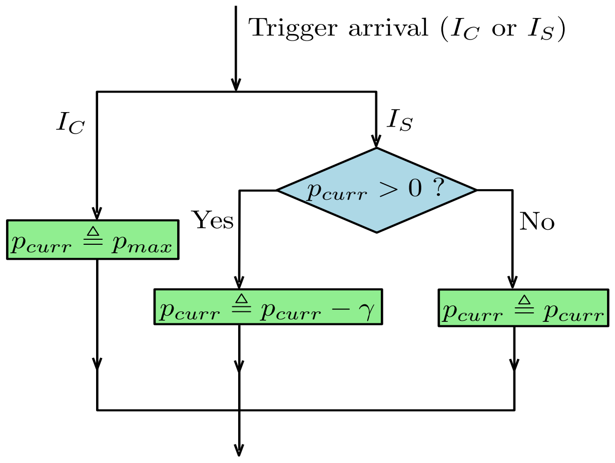

Trigger arrival (I_C or I_S)
I_C
I_S
$p_{curr} > 0$?
Yes
No
$p_{curr} \triangleq p_{max}$
$p_{curr} \triangleq p_{curr} - \gamma$
$p_{curr} \triangleq p_{curr}$

Figure 4.3: Flowchart representation of the opportunistic adaptive sampling scheme.

$p_{curr} > 0$. Here, γ is the step size, with p_{max} being an integral multiple of γ. The algorithm takes I_S as a signal that the network is uncongested, and thus attempts to increase the haptic data rate. The increase in the haptic data rate is performed in a controlled, multistep manner. A premature, abrupt scaling up of the data rate is more likely to cause persistent congestion, thereby posing a potential threat to haptic QoS.

To summarize, during congestion the opportunistic sampler immediately reverts to the static adaptive sampler with the minimum possible data rate. While the network is uncongested, the opportunistic sampler cautiously advances towards the peak rate transmission. It is during the uncongested periods that the opportunistic sampler gains over the static sampler. In the proposed scheme, p_{max} and γ are the algorithm parameters. At the start of the telehaptic session, the opportunistic sampler operates at $p_{curr} = p_{max}$ as the network condition is unknown, and it is safer to start the transmission conservatively.

Leveraging time-out strategy: In the previous chapter, we introduced the concept of time-out mechanism as a strategy to reduce the signal drift at the receiver. We also mentioned that it also helps in performing efficient rate control. In this section, we explain this aspect in detail.

We saw earlier that the triggers are generated when the OP receives at least N original E2E delay measurements piggybacked on the backward channel. This means that the OP can learn the forward channel condition only when it transmits new packets, as new delays are computed only on arrival of packets at the TOP. During intervals when the human user carries out very slow hand movements, the velocity signal is a slowly varying function of time. In such scenarios, the adaptive sampling at threshold p_{curr} might discard several successive haptic samples.

This gives rise to a long period without any packet transmissions from the OP. As a consequence, the OP is starved of feedback on delays on the forward channel for a long time, thus temporarily pausing the adaptive sampling threshold adjustment. This might result in severe network underutilization. Hence, the time-out strategy serves to minimize the network underutilization by ensuring that in the worst-case at least one packet is transmitted in the duration T_o.

4.3 Experimental Setup

In this section, we describe the setup of a real-time telepottery experiment that we design for assessing the performance of the opportunistic adaptive sampling scheme.

4.3.1 Haptic Data Generation

For the purpose of generating a realistic haptic dataset, we adopt the haptic pottery model developed in [21]. The model generates a volume conserving, standalone rotating clay structure. In [21], the authors develop a realistic deformation model for pottery in which the human user interacts with a rotating clay model via haptic and visual display. The user pushes the stylus of the device in order to make contact with the virtual clay to design a virtual pottery structure. Note that the human user and the clay object are closely coupled in terms of their physical locations in [21].

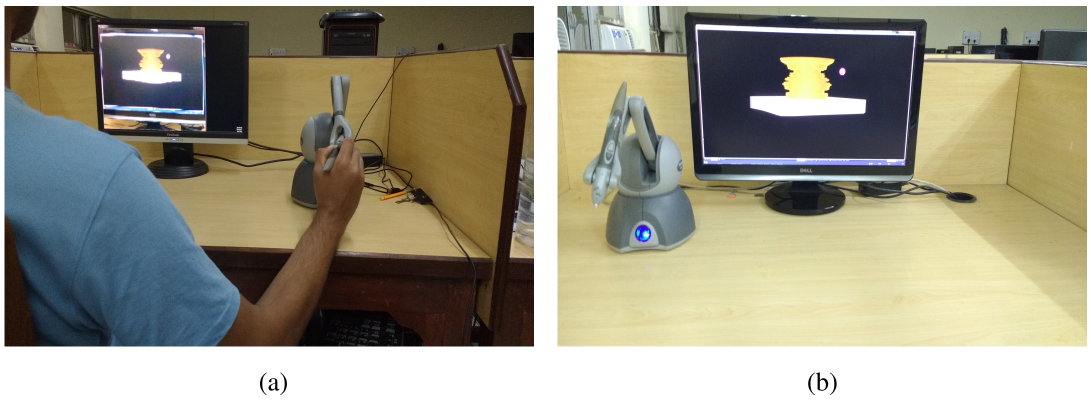

(a) (b)

Figure 4.4: Real-time telepottery experimental setup showing (a) human operator (b) teleoperator. The smaller window in the computer screen at the OP is the visual feedback.

For the purpose of emulating a telehaptic activity, we decouple the human user and the clay model to design a real-time telepottery model in which a human subject interacts with a remote, virtual pottery model on a real network. In addition to haptic and video, we also introduce audio feedback from the TOP end. The volume preserving pottery model [21] is rendered at the TOP, which is equipped with a haptic device and a generic webcam. Figure 4.4 shows the human subject at the OP and the haptic device at the TOP. The subject receives haptic-audio-video feedback from the TOP, and manipulates the virtual clay model remotely.

The subjects were initially briefed about the concept of force feedback as few of them were new to the notion of haptics. Later, we explained them the telepottery task 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. The subject pushes the haptic device stylus so as to establish contact with the clay model and shape it into a pot. Initially, participants perform the telepottery task to get acquainted with the telepottery setup. During this phase, the participants explored the telepottery model under an experts guidance until they were confident of performing the task independently which is the data collection phase. During this phase, we record the velocity traces generated by each of the subjects over the telepottery interaction.

We use Geomagic Phantom Omni haptic devices [4] with six degrees of freedom (DoF) input and three DoF output capability, at the OP and the TOP. The audio-visual information is captured at the TOP using a Microsoft Lifecam VX-2000 webcam. An open source, cross platform haptic rendering software named HAPI [5], which is written in C++, is used as the haptic interface program. Both end systems, running on Windows 7 OS, are connected through a 100 Mbps LAN. The human user at the OP pushes the stylus of the device in order to establish contact with the clay model. The user movements are captured using device position and velocity coordinates, and are communicated to the TOP on the forward channel using our protocol. At the TOP, a proportional-derivative (PD) controller (described next) is designed to drive the haptic device to follow the user trajectory.

Teleoperator setup: In order to engineer the master-slave relationship between the two haptic devices, we implement a PD controller at the TOP that drives the haptic device along the path of the user trajectory. The PD controller is chosen over other standard controllers for achieving a swift TOP response. Let $P(t_{-1})$ and $V(t_{-1})$ be the latest received position and velocity of the OP, respectively, and let $P(t)$ and $V(t)$ denote the current position and velocity of the TOP. The force input $F(t)$ to the TOP-side haptic device in order to minimize the error in positions of the two devices is given by

$$F(t) = K_p \cdot (P(t_{-1}) - P(t)) + K_v \cdot (V(t_{-1}) - V(t)) \tag{4.2}$$

where K_p and K_v denote the position and velocity gains of the controller, respectively. We empirically choose $K_p = 25$ N/m and $K_v = 2$ Ns/m in our experiments.

It is important to state the distinction between $F(t)$ and the force feedback that is sent to the OP. $F(t)$ represents the force required for driving the TOP-side haptic device along the path traversed by the OP-side haptic device. On the other hand, the force feedback, is generated when the interaction point at the TOP collides with the clay model. These forces are read from the haptic interface and sent to the OP.

Human subjects: Eight human subjects, six males and two females, aged between twenty three and fifty two years participated in the telepottery experiment, and none of them suffered from any known neurophysiological disorders. Out of them, three were regular users of the haptic devices. Nevertheless, ample time was provided to the subjects prior to data collection to get accustomed to the telepottery setup.

4.3.2 NS3 Simulation Testbed

The network simulation tools provide an easy, flexible platform for carrying out performance evaluations of the network protocols than a real-network. For our evaluations, we resort to a popular, discrete event network simulator tool called NS3 [74]. We consider a single bottleneck dumbbell topology, shown in Figure 4.5, that is generally used in the networking literature for studying the protocol behavior in presence of exogenous cross-traffic sources. In order to simulate asymmetric networking conditions on the forward and the backward channels, we create unidirectional links between the OP and the TOP node. For concreteness in exposition, we investigate the network scenario in which only the forward channel is congested. l_1 acts as the bottleneck link on the forward channel. Therefore, increasing queueing delays due to congestion are observable at the intermediate node n_1. The velocity traces recorded during the telepottery experiment are utilized in NS3 simulations for performance evaluations of the opportunistic adaptive sampling scheme. For brevity, we report the results corresponding to one of the traces. However, we note that the nature of our findings is similar for others as well.

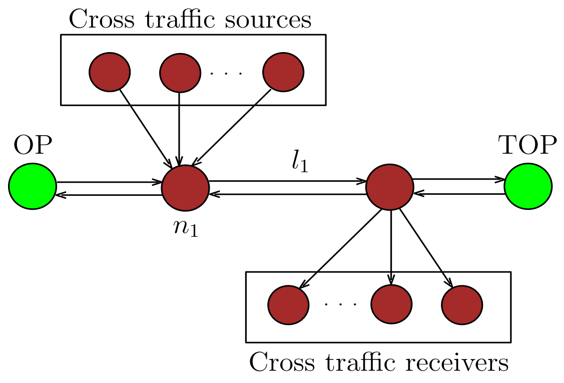


Figure 4.5: Single bottleneck dumbbell topology designed for investigating the performance of the opportunistic sampler. l_1 - bottleneck link on the forward channel; n_1 - intermediate node.

The unidirectional links have a capacity (denoted by μ) of 1.5 Mbps. This represents the typical capacity of a medium speed internet link. However, it is worth stating that the nature of our findings remain unchanged across different channel capacities. The propagation delay of each link is configured to 5 ms. Hence, the one-way propagation delay between the OP and the TOP is 15 ms. This is typically the propagation delay exhibited by a link of length around 3200 kms. The queue size at n_1 is set to 15000 bytes.

For testing the proof-of-concept, we work with the popular Weber sampler (discussed in Chapter 3). It is worth stating that the opportunistic algorithm can be applied to any adaptive sampling mechanism in general. For our simulations, we choose a realistic value of p_{max} = 30% [47], and γ = 2%. However, it should be noted that our findings remain robust to changes in these parameters. We work with the default time-out interval T_o = 20 ms, as discussed in the Chapter 3. We compare our opportunistic scheme with the static Weber sampler operating at a fixed maximum threshold of $p =$ 30%. For the purpose of illustration, we choose the zero-order hold strategy for haptic signal reconstruction at the TOP. Another algorithm parameter is the weight used in the moving average filter w in Equation (4.1). We set w = 0.2, as recommended in the widely cited reference [70].

Feasible Range of CBR Traffic: Let R_{cbr} denote the CBR cross-traffic intensity on the forward channel. We evaluate the proposed sampler in the CBR cross-traffic range such that (i) peak rate transmission is infeasible, and (ii) the network always has enough bandwidth to support the instantaneous rate of the static sampler.

Taking the packet overheads into account, the peak transmission rate $R_{peak} =$ 696 kbps.

For satisfying condition (i) we need to ensure that $R_{peak} + R_{cbr} > \mu$. Thus, $R_{cbr} = 804$ kbps is the largest possible rate such that the peak rate transmission is feasible. Hence, 804 kbps acts as the lower bound of R_{cbr}.

In order to find the upper bound for R_{cbr}, we choose the velocity trace corresponding to Subject 1. We observe that the time-averaged data rate of the velocity trace is 221 kbps, whereas the instantaneous rate fluctuates in the range 0 - 501 kbps. In order to satisfy condition (ii), we have to ensure that the network has at least 501 kbps available for the telehaptic stream. Therefore, $R_{cbr} = 999$ kbps is the maximum CBR cross-traffic intensity for our simulations. Hence, we evaluate the proposed protocol in the CBR range $R_{cbr} \in [804, 999]$ kbps.

4.4 Results

In this section, we evaluate the performance of the opportunistic sampler in terms of SNR improvement at the TOP over the static sampler, and the haptic QoS compliance. Additionally, we assess the interplay between the exogenous traffic streams and the telehaptic stream due to the proposed scheme, and report the friendliness of the opportunistic algorithm to the concurrent cross-traffic in the network. Initially, we investigate the interplay with the network-oblivious CBR cross-traffic. Later, we move to studying the interplay with the commonly used network-aware, rate-adaptive protocol called Transmission Control Protocol (TCP).

4.4.1 Interplay with CBR Cross-Traffic

In this section, we report the behavior of the opportunistic scheme in presence of multiple CBR cross-traffic streams.

SNR-Throughput Measurements: We now evaluate the performance by measuring the signal-to-noise ratio (SNR) of the reconstructed signal, and the throughput of the telehaptic stream. We begin by reporting the SNR improvement for all the recorded velocity traces. Each of the recorded velocity signals are initially transmitted through an uncongested network with a capacity of 100 Mbps, and a one-way propagation delay of 15 ms. The corresponding reconstructed velocity signals at the TOP are treated as benchmark, for each specific human subject. Note that due to the high channel capacity and uncongested conditions, the reconstructed signal turns out to be a time-shifted version of the original signal, with the offset being 15 ms (one-way propagation delay).

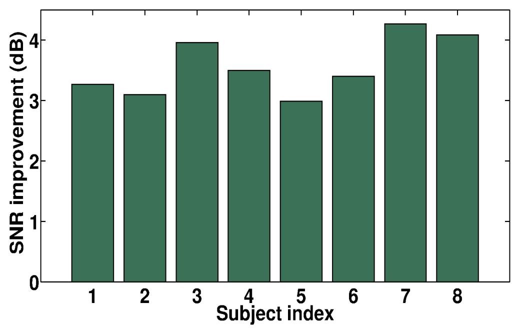


Figure 4.6: SNR improvement of opportunistic sampler over the static sampling measured for the velocity updates across the subjects.

First we discuss the SNR improvement achieved by the proposed opportunistic sampler over the static sampler across the recorded real-world velocity traces. For these simulations, we fix $R_{cbr} = 900$ kbps to demonstrate the improvement for a mid-range CBR traffic. Our results are presented in Figure 4.6. Note that the opportunistic sampler achieves an average SNR improvement of 3.57 dB over the static sampler, across the traces. The substantial improvement is because the static method underutilizes the network, and loses out on the possibility of transmitting a higher resolution haptic signal. Our proposed method opportunistically senses the available bandwidth in the network and transmits at a higher haptic data rate. In the subsequent discussion, we will see that this increase in the rate results in an E2E delay that is QoS-compliant.

Next we study the gain in telehaptic throughput and the corresponding SNR improvement of the proposed method over the static sampler for a fixed velocity trace, over the full range of feasible CBR cross-traffic intensity. Throughout this section, we use the velocity trace corresponding to Subject 1.

Figure 4.7 demonstrates the correlation between the telehaptic throughput and SNR improvement of the opportunistic sampler over the static sampler. The time-averaged haptic data rate due to the static sampler for Subject 1 turns out to be 221 kbps. It can be observed that for $R_{cbr} \leq 804$ kbps, the opportunistic scheme achieves the peak haptic data rate of 696 kbps, thereby resulting in a near error-less signal reconstruction. In this case, the SNR improvement is very large (around 5.9 dB) as the error between the benchmark signal and the reconstructed signal due to the opportunistic sampling is negligible.

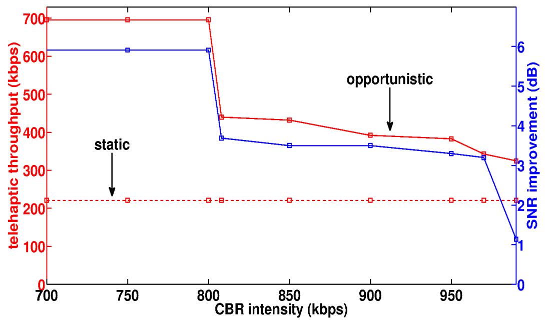


Figure 4.7: Telehaptic throughput adaptation along with the corresponding SNR improvement over the static sampling for a real-world velocity signal.

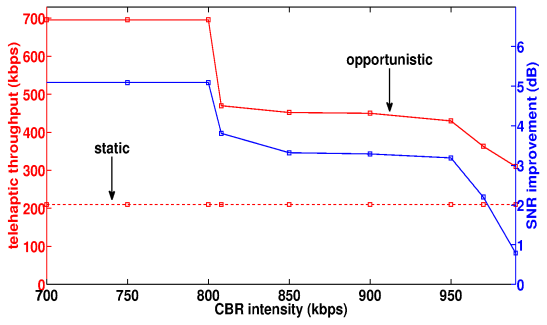


Figure 4.8: Telehaptic throughput adaptation along with the corresponding SNR improvement over static sampler for a synthetic velocity signal.

As R_{cbr} is increased from 804 to 999 kbps, the opportunistic scheme gradually reduces its transmission rate, thus ensuring that the concurrent flows are not throttled. Consequently, the SNR improvement also reduces gradually with increasing R_{cbr}. We note that the CBR cross-traffic flows attain full throughput and sustain no losses over this range of R_{cbr}, though this is not apparent from the figure. Even for the upper threshold of $R_{cbr} =$ 999 kbps, the opportunistic sampler manages to transmit significant amount of extra haptic packets, thereby resulting in an SNR improvement of 1.2 dB.

		End-to-end delay (ms)	Jitter (ms)
QoS		30	10
Expt.	Max	26.995	0.636
	Avg	19.645	0.165

Table 4.1: QoS performance of the proposed opportunistic scheme with $R_{cbr} =$ 990 kbps.

To demonstrate that our opportunistic scheme meets the telehaptic QoS requirements, Table 4.1 gives a comparison of the experimental observations and the corresponding QoS limits of the E2E delays and jitter, with $R_{cbr} =$ 990 kbps. Note that the haptic flow does not lose any packets. Thus, we conclude that even under very high cross-traffic intensity (near the higher end of the range), the opportunistic scheme adheres well to the telehaptic QoS specifications. We also verify that the QoS requirements are satisfied even for $R_{cbr} =$ 999 kbps.

Finally, we evaluate the performance of the opportunistic sampler for a synthetic velocity trace generated using a standard statistical model, in order to provide a task or subject independent analysis of its performance. For this purpose, we generate a white Gaussian signal, smoothed using a simple moving average filter with a cutoff frequency of 100 Hz. We feed this signal in place of the velocity trace generated by the OP, and test the performance of the proposed sampler over a wide range of mean and standard deviation of the input Gaussian signal. For brevity, we report the observations for mean and standard deviation of 0.3 cm/s and 0.3 cm/s, respectively. The chosen values are realistic, since rapid hand movements rarely occur in a practical telehaptic scenario. Figure 4.8 presents the telehaptic throughput and the corresponding SNR improvement of the proposed sampler as a function of R_{cbr}. We note that the results are quite similar to those from the subject trace as shown in Figure 4.7. This demonstrates that the performance enhancement of the opportunistic adaptive sampler is robust with respect to the haptic source.

Temporal Behavior: We now move to depicting the temporal variation of the E2E delays and the chosen instantaneous sampling threshold p_{curr}. For this experiment, we set $R_{cbr} = 990$ kbps to study the worst-case behavior. The sample-wise E2E delay encountered by the velocity updates, along with the corresponding p_{curr} is presented in Figure 4.9. It can be observed that when the forward channel is uncongested (for example, 800 ms- 1500 ms), the E2E delays exhibit a steady behavior, and the opportunistic algorithm reduces p_{curr} based on the threshold update scheme presented in Section 4.2.2. It is clear from the graph that when the network is congested (for example, around 1600 ms), the algorithm performs a quick congestion control through the aggressive fall back approach without significantly building up the delay. The switch to p_{max} is deferred after the start of a delay build-up primarily due to the propagation delays of the forward and the backward channels. The periods where $p_{curr} < p_{max}$ are when the opportunistic scheme transmits at a higher rate than the static sampler, resulting in a better velocity signal reconstruction compared to the static sampler, as discussed earlier.

Until this point, we considered a steady cross-traffic throughout a simulation run. Next we investigate the adaptation of the opportunistic sampler to time-varying cross-traffic. We present the temporal variation in the haptic packet rate, in a piecewise-CBR cross-traffic scenario. We simulate two CBR cross-traffic sources on the forward channel: C_1 with a data rate of 850 kbps, and C_2 with a data rate of 100 kbps. We use different combinations of the two CBR sources to design a time-varying cross-traffic scheme as shown in Equation (4.3).

$$R_{cbr} = \begin{cases} 0, & \text{for } 0 < t \le 3s \\ 850 \text{ kbps } (C_1), & \text{for } 3 < t \le 5s \\ 950 \text{ kbps } (C_1 \text{ and } C_2), & \text{for } 5 < t \le 7s \\ 100 \text{ kbps } (C_2), & \text{for } t > 7s \end{cases} \tag{4.3}$$

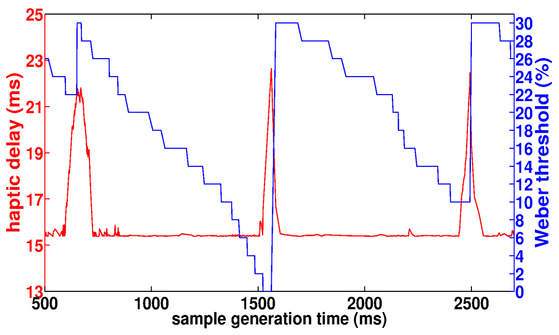


Figure 4.9: E2E delay encountered by the haptic samples, along with the corresponding adaptively chosen sampling thresholds. Here the cross-traffic is given by $R_{cbr} = 990$ kbps.

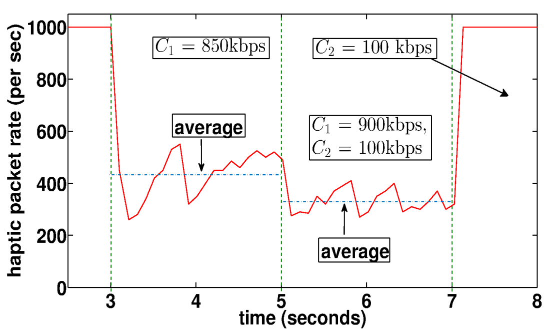


Figure 4.10: Temporal behavior of the haptic packet rates due to the opportunistic algorithm under piecewise CBR cross-traffic conditions.

Our results are shown in Figure 4.10. Until 3 seconds, the telehaptic source transmits at 1000 packets/sec as there are no concurrent flows on the forward channel. After 3 seconds, the available capacity on the forward channel is insufficient to sustain the peak haptic data rate. Accordingly, the opportunistic scheme dynamically lowers the telehaptic packet rate. The reduction in the packet rate due to CBR source C_1 alone is clearly observable between 3 and 5 seconds. At 5 seconds, the additional traffic due to the CBR source C_2 results in a further reduction in the packet rate. C_1 aborts transmission at 7 seconds, after which point the telehaptic source resumes transmission at 1000 packets/sec, as the overall load on the network is less than the capacity.

4.4.2 Interplay with TCP Cross-Traffic

In this section, we anlayse the interplay between the opportunistic scheme and the TCP cross-traffic. For this study, we consider a commonly deployed version of TCP known as NewReno [32]. The TCP sources have an infinite backlog of data, meaning that they never run out of data to transmit. As before, we use the velocity trace generated by Subject 1. Figure 4.11 presents

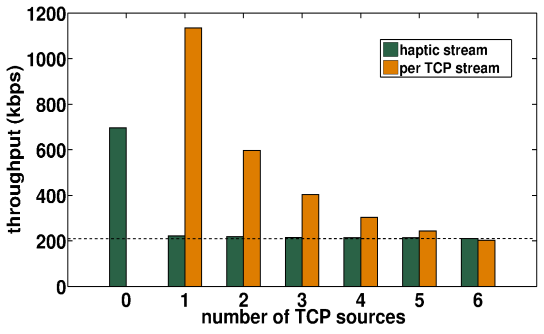


Figure 4.11: Demonstration of TCP-friendliness of the opportunistic algorithm.

the telehaptic and TCP throughput per stream, for a varying number of TCP NewReno streams. In absence of TCP flows, the telehaptic source gains its peak throughput of 696 kbps. It is clear from the graph that in presence of TCP sources, the telehaptic source attains a throughput of 221 kbps which corresponds to the static sampler with p_{max} = 30%. Thus, the opportunistic sampler backs off to the minimal rate when coexisting with TCP sources. Therefore, we conclude that the proposed opportunistic adaptive sampling scheme is highly TCP-friendly. However, we observe severe violations of haptic delay QoS. The presence of multiple TCP sources also increases packet losses in the network. Therefore the telehaptic throughput gradually reduces with increasing number of TCP sources, even though the telehaptic source rate remains at the minimum rate of 221 kbps.

The observed behavior of the opportunistic scheme is because of the rate adaptation mechanism of the NewReno protocol. This protocol uses packet loss as the congestion signal, and employs the standard AIMD principle for rate adaptation. This causes continuous variations in the E2E delays, thus depriving the opportunistic sampler of an opportunity to increase its rate. Also, the inherent nature of the NewReno protocol causes packet losses, thereby causing the network to drop haptic packets as well. We perform a detailed analysis of the impact of coexisting TCP flows on telehaptic QoS in Chapter 7, where we analyze the interplay between TCP NewReno and a broad class of telehaptic protocols.

4.5 Conclusions

In this chapter, we explored a possible solution for enhancing the quality of telehaptic communication based on an opportunistic network utilization approach on the forward channel, where the end receiver is a high-precision robotic telemanipulator. We demonstrated that the proposed algorithm enhances the telehaptic throughput by a fairly considerable extent through timely detection of the network state and appropriate tuning of the data rate, without overwhelming the underlying shared network. The results presented show haptic QoS adherence even under heavy cross-traffic scenarios. The SNR measurements reveal that the opportunistic adaptive sampler outperforms the static adaptive sampler in terms of the quality of signal reconstruction at the teleoperator. We also demonstrated the cross-traffic friendliness behavior of the opportunistic scheme. However, the network should be provisioned for the maximum rate of the static adaptive sampler, which may be close to the peak telehaptic rate. In order to overcome this drawback, one potential approach is to ensure that the instantaneous telehaptic rate is independent of the underlying signal profile. We explore this aspect in Chapter 6.

Chapter 5

A Multiplexing-Packetization Framework for Backward Channel

5.1 Introduction

In Chapter 3, we presented the protocol design for telehaptic communication on the forward channel. Since the forward channel deals with the haptic modality alone, scheduling the packet transmissions is rather simple. On the other hand, the backward channel carries heterogeneous media types viz. haptic, audio and video. However, future telehaptic applications need not be restricted to the usage of the aforementioned media types, and could potentially explore the possibility of augmenting other sensory technologies, for example, olfaction [72]. This poses an additional challenge of prudently scheduling the media frames for transmission, as the frame delay and jitter are heavily dependent on the time instants of their transmission.

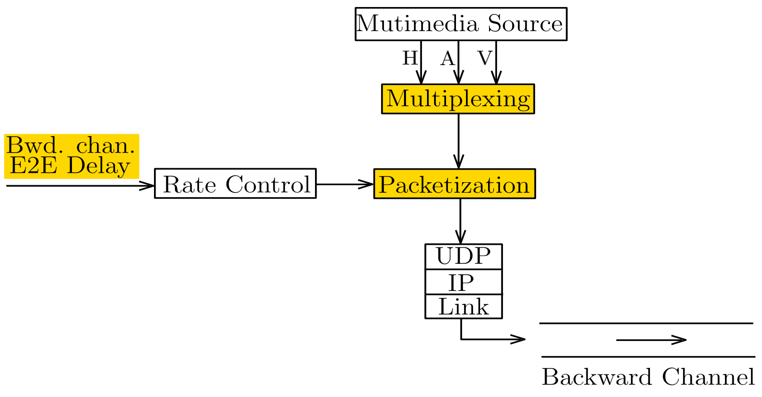


Figure 5.1: Architecture of Haptics over Internet Protocol (HoIP) for the backward channel, focusing on the multiplexing and packetization aspects which form the subject of this chapter.

As discussed in Chapter 1, each of the involved media types have non-identical QoS specifications. The transmission protocol needs to ensure that the QoS requirements of all the participating media types are simultaneously satisfied. Hence, in addition to the protocol functionalities discussed in Chapter 3, the backward channel protocol should also address the problem of efficiently multiplexing the telehaptic data in order to achieve a holistic QoS compliance. In this chapter, we address the problem of multiplexing of the haptic-audio-video data, and also define the packetization process on the backward channel, as highlighted in Figure 5.1

Recently, an elegant protocol for telehaptic multiplexing on the backward channel was introduced in [24]. The protocol, known as *visual-haptic multiplexing*, designs a packet scheduling algorithm for telehaptic communication involving haptic and video data. The protocol employs Weber sampling for compressing the haptic signal. The protocol intelligently packs the perceptually significant haptic samples with video data of a pre-defined size, resulting in a payload of worth 1 ms. When a series of perceptually insignificant haptic samples are generated, the protocol injects a large video chunk, of worth not exceeding 15 ms, into the network. By design, the protocol takes no more than two media types into consideration. Hence, the problem of multiplexing in case of multiple media types (apart from haptic) is not addressed by this protocol. Therefore, this protocol may not be useful especially in telehaptic transmission that involve haptic, audio and video transmissions. We seek to fill this gap through the design of a generic multiplexer that can efficiently handle more than two media types.

In this chapter, we design a multiplexing cum packetization framework for telehaptic communication on the backward channel. The multiplexing framework is a substantial refinement of the design proposed in [24]. The protocol transmits packets at the standard haptic sampling rate of 1 kHz. Each packet carries a haptic sample along with audio/video data. We describe a basic framework for composing fragments in a telehaptic communication. Later, we design two multiplexing rules for scheduling the audio/video multiplexing. While the *first come, first serve* (FCFS) rule selects the transmission frame based on the generation timestamps, the *QoS-aware* scheme multiplexes the most QoS-sensitive media ahead of the others. We report the performance evaluation of the proposed multiplexing rules using the network simulator tool NS3 [74], and demonstrate that the QoS-aware rule outperforms the FCFS rule in terms of audio jitter, and is therefore well suited for delay-critical telehaptic applications.

The remainder of the chapter is organized as follows. In Section 5.2, we describe the backward channel HoIP framework. Section 5.3 describes the experimental setup designed for the evaluation of the proposed method and the findings of our experiments. We state our conclusions in Section 5.4.

5.2 Architecture

In this protocol, we adopt several features from the forward channel protocol architecture, described in Chapter 3. These features include

- in-header delay notification mechanism
- signal reconstruction techniques
- the packet structure (discussed next in Section 5.2.2).

The novelty of this chapter lies in the design of the multiplexer strategies. In this section, we present two multiplexer frameworks for usage in telehaptic communication viz. first come first serve (FCFS) and QoS-aware. We then move to describing the relevant packet header structure developed for transmitting the telehaptic payload on the backward channel.

5.2.1 Media Multiplexing Framework

In this section, we describe the design of our media multiplexing framework in detail. 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 [24] 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 considerably long time. This blocks the transmission of subsequent haptic (or any media in general) samples, thereby adding to their E2E delay substantially. As an example, transmitting an entire video frame of size 2 kB in a packet results in a large transmission delay of 16 ms on a channel with a bandwidth of 1 Mbps. Note that the transmission delay refers to the amount of time required for the packet to be injected into the network. Hence, proper monitoring of the media frames is essential to ensure that they are delivered to the operator (OP) before the respective deadlines. The media multiplexing framework proposed in this chapter is a substantial improvement of that in [24], in the sense that the proposed schemes carefully handle multiple media transmissions in a telehaptic communication, which is not addressed in [24].

Recall that out of the three telehaptic media types, haptic media is the most sensitive to delay. The idea behind the multiplexing principle is to assign highest transmission priority to the haptic samples, so that the arrival of other media frames do not deprive the haptic samples of the network resources. For this purpose, we set the clock rate of the multiplexer to 1 kHz, the typical sampling rate of haptic signals. Hence, the multiplexer works in synchronization with the haptic sample generation process. This implies that the haptic samples are registered at the multiplexer module as soon as they get generated. Note that we will use the terms samples and frames interchangeably throughout this thesis.

Each time a haptic sample is generated, the multiplexing framework generates a *telehaptic fragment* that needs to be packetized and dispatched. Hence, a telehaptic fragment is generated once in every millisecond. The telehaptic fragment is composed of the latest generated haptic sample, and a chunk of the audio/video data that was not multiplexed before. The size of this audio/video chunk needs to be carefully chosen as it directly influences the media delay and jitter. In order to maintain equilibrium, we must ensure that the telehaptic payload generation rate is equal to the multiplexing rate. Hence, the size of the audio/video chunk that is included in a telehaptic fragment is subject to the equilibrium criteria, and hence depends on the audio/video payload rates. The equilibrium criteria ensures that the audio/video frames are injected into the network at the same rate as they are generated, thereby minimizing the frame accumulation in the buffer. The multiplexing framework is shown in Figure 5.2.

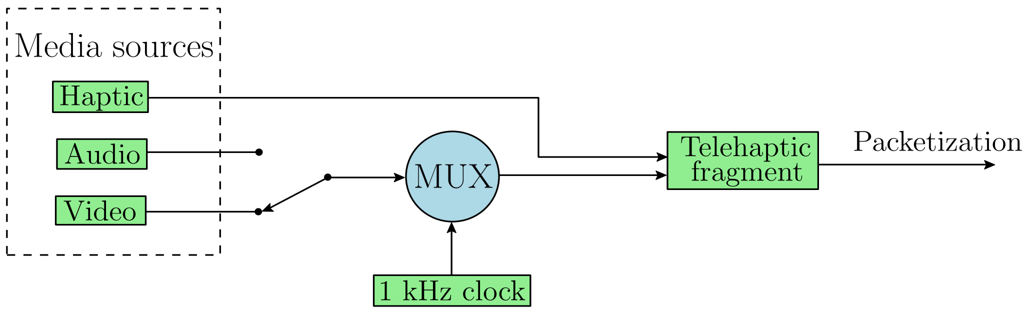


Figure 5.2: Block diagram representation of the basic framework designed for composing the telehaptic fragments on the backward channel.

Let f_h represent the haptic sampling rate in Hz, and s_h represent the size of a haptic sample in bytes. Let f_a and f_v represent the peak frame rates of audio and video data, respectively, in Hz. Let s_a and s_v represent the peak size of an audio and video sample, respectively, in bytes. Let R_p denote the peak telehaptic payload generation rate in kbps. Therefore, R_p can be expressed in terms of the individual media parameters as

$$R_p = (f_h s_h + f_a s_a + f_v s_v) \cdot (8/1000). \tag{5.1}$$

Let s_f denote the peak size of the telehaptic fragment in bytes. In order to satisfy the equilibrium criteria, s_f is given by

$$s_f = \frac{R_p}{f_h} \cdot (1000/8). \tag{5.2}$$

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

$$s_m = s_f - s_h. \tag{5.3}$$

The technique of composing the telehaptic fragments is demonstrated in Figure 5.3. Let us assume that the audio/video frame being multiplexed consists of l segments, each of size s_m. For the purpose of illustration, we name the media segments as $M_1,\ M_2,\ ...\ M_l$, and the haptic samples as $H_1,\ H_2,\ ...\ H_l$. In the i^{th} multiplexing cycle, where $i \in [1, l]$, the multiplexer packs the segment M_i with the haptic sample H_i to form a telehaptic fragment. This telehaptic fragment is then packetized for transmission, as explained later in Section 5.2.2. Once the multiplexing of a frame completes, the next frame is chosen according to the schemes described next.

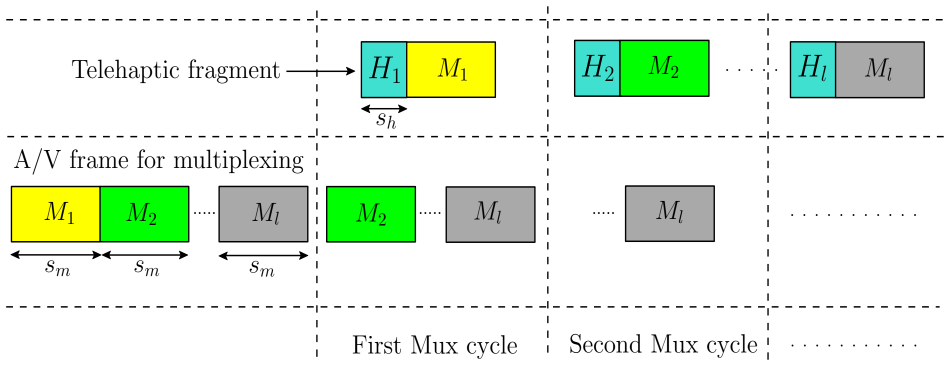


Figure 5.3: Generation of telehaptic fragments by packing the haptic samples with audio or video segments.

When either an audio or a video frame is available, the multiplexer simply extracts s_m bytes of data and bunches it with the latest haptic sample. On the other hand, when both audio and video frames contend for multiplexing, then the multiplexer has to choose one of them for inclusion in the telehaptic fragment. This results in the deferment of transmission of the other frame until it gets its turn, causing additional delay. Hence, the multiplexer needs to resolve this conflict efficiently in order to ensure that no media QoS violation occurs. In the following discussion, we present two rules that propose different conflict resolution techniques.

1. **First come, first serve rule:** Our first approach for conflict resolution is to perform multiplexing depending on the time instant of the frame generation. The multiplexing principle used here is *first come, first serve* (FCFS) based. According to the FCFS rule, the media frame that is generated first is assigned the highest multiplexing priority ahead of all others. Similar to the case of haptic samples, each of the audio/video frames is associated with a timestamp that indicates the clock value at which the frame is generated. The media frame with the highest priority is then used as payload in the telehaptic fragment as per Equation (5.3). Once a media frame is chosen, the FCFS scheme is insensitive to the frame generation process until it is multiplexed completely. This means that once a frame is selected, it is multiplexed in an uninterrupted manner until its completion, thereby being oblivious to the subsequently generated frames. To summarize, the multiplexer packs s_m bytes from the audio/video frame that is generated earlier in time in each telehaptic fragment along with the latest haptic sample. While the FCFS rule is simple to design, in that the multiplexer need not have any knowledge of the participating media types and their respective QoS, it is oblivious to the media QoS requirements. This network obliviousness may increase the risk of QoS violation, thereby causing catastrophic effects in a delay-critical telehaptic activity.

2. **QoS-aware rule:** In order to ensure that the multiplexer actions are driven by the media QoS requirements, we design a hierarchical multiplexing rule called *QoS-aware multiplexing*. The multiplexer assigns the transmission priorities to the media types based on their QoS specifications. The media with the most stringent QoS criteria receives the highest priority. During a conflict, the media frame with the highest transmission priority is multiplexed ahead of the others. At any point in time, the arrival of a higher priority media causes an preemption of the ongoing frame multiplexing. The multiplexer then switches to the newly generated frame until the partially multiplexed frame gets the high-

est priority. In our case, clearly audio QoS constraints are more stringent than those of video. Hence, the audio frames are multiplexed ahead of the video frames. To summarize, the multiplexer packs s_m bytes of audio/video data in each telehaptic fragment along with the latest haptic sample, giving strict multiplexing priority to audio over video.

We evaluate the above multiplexing schemes through simulations and report the findings in Section 5.3.

5.2.2 Packet Structure

In this section, we describe the HoIP packet structure designed for transmitting the telehaptic payload on the backward channel. The packet structure of HoIP for the backward channel is shown in Figure 5.3. Each row in the frame is composed of 32 bits. The top row indicates bit positions of the fields. The bits are numbered from 0 to 9 for the purpose of illustration. The HoIP packet starts from the second row. We explain the meanings of each of the header fields in the following.

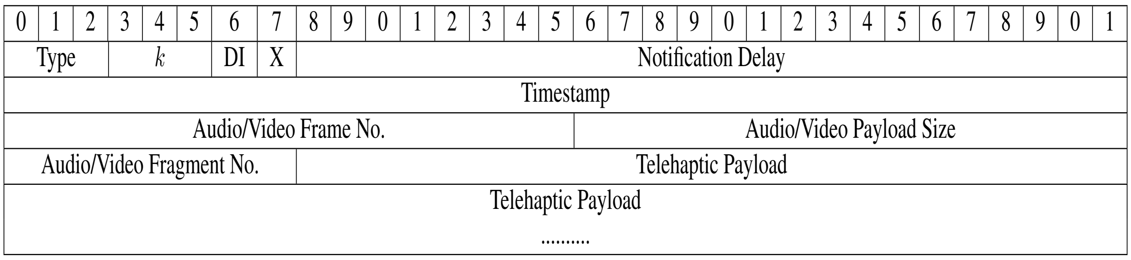


Figure 5.4: Packet structure of HoIP on the backward channel.

- *Type* (3 bits): Indicates the type of telehaptic payload carried by the packet. The typical telehaptic payload consists of haptic, audio and video data. However, as per the protocol design the packets carry either haptic-audio or haptic-video payload. At times, when the audio/video frames are not available the packets carry haptic sample only. The payload type is encoded in the following manner. 0: haptic; 1: haptic-audio; 2 - haptic-video. The other bits are reserved for augmenting other media to the telehaptic application in future.
- k (3 bits): Indicates the number of haptic samples included in the current packet. This field provides support for another type of lossless, rate adaptive transmission scheme, where the rate control is performed by merging multiple haptic samples in a packet. We describe this technique in detail in Chapter 6.

- *DI* (1 bit): Denotes the *delay indicator* field. This is used to indicate the status of the delay embedded in the packet header (explained below). 0: new delay, 1: duplicate delay. This field is the backward channel counterpart of the *DI* field introduced in Chapter 3.
- *X* (1 bit): This bit is reserved for future enhancements to the protocol.
- *Notification Delay* (24 bits): Indicates the latest forward channel E2E delay as measured by the TOP. This delay forms the basis of the rate adaptation on the forward channel by OP, as discussed in Chapter 4.
- *Timestamp* (32 bits): Indicates the generation time (in ms) of the earliest haptic sample in the payload. This field, along with the received timestamp at the teleoperator (TOP), is used to calculate the backward channel E2E delay, as discussed in Chapter 4. Further, this field can be used for packet reordering and scheduling the display of the haptic samples.
- *Audio/video frame number* (16 bits): Indicates the sequence number of the audio/video frame whose fragment is being carried by the current packet. This will be used by the receiver to reconstruct the frame from the discrete fragments.
- *Audio/video payload size* (16 bits): Indicates the size of the audio/video payload being carried by the packet in bytes.
- *Audio/video fragment number* (8 bits): Indicates the sequence number of the audio/video fragment belonging to the frame number that is being carried by the current packet. This will be utilized by the receiver to reconstruct the frame from the discrete fragments.

It is worth remarking that since a packet carries either haptic-audio or haptic-video payload, the effective size of the HoIP header per packet on the backward channel is 13 bytes.

5.2.3 Communication Framework

Figure 5.5 shows the high-level architecture of the communication framework featuring the multiplexer and the packetization modules. Recall that the opportunistic sampling scheme at the OP, presented in Chapter 4, relies heavily on the forward channel delays that are piggybacked on the backward channel. The piggybacking of the measured forward channel delay (d_{fwd}) is also shown in Figure 5.5. d_{fwd} is conveyed to the *packetization module*, which inserts it into the packet header for transmission on the backward channel. The network analyzer module,

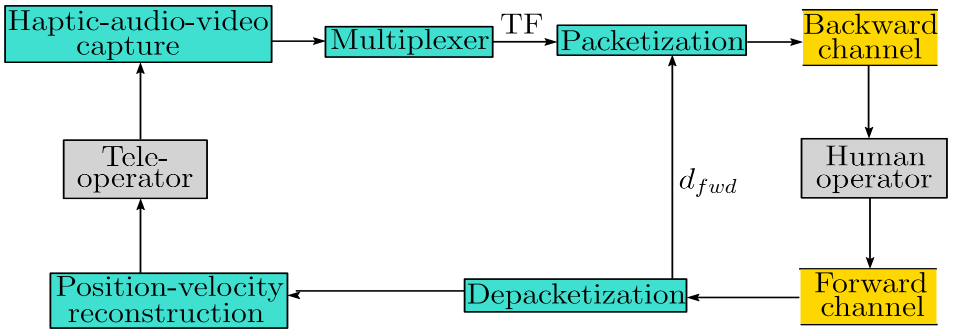


Figure 5.5: High-level architecture of the communication framework on the backward channel showing the multiplexer module. TF denotes telehaptic fragment.

inbuilt in the opportunistic sampling scheme, utilizes this delay measurement for estimation of the forward channel state.

5.3 Performance Evaluation

We now move to investigating the performance of the proposed backward channel multiplexing cum packetization protocol. The objective of this evaluation is to assess the performances of the proposed multiplexing rules, and get a sense of the haptic/audio/video delay and jitter introduced by them. It is worth remarking that the sender and receiver module delays of this protocol are similar to that measured in Chapter 3. Therefore, we report the delays due to multiplexing only. In the following, we describe the testbed used for the performance evaluation, and present the findings of our experiments.

5.3.1 Setup

We begin by mentioning the telehaptic data rates, followed by the network configurations that we use for the experiments in order to assess the performance of the proposed multiplexing framework. For simplicity, we work with dummy haptic/audio/video traces in our simulations, since the multiplexing is robust to the payload itself.

Telehaptic Data Rate: We work with the haptic data rate that corresponds to Geomagic Phantom Omni [4] which is a single point of contact device. Every sample of the 3D force signal requires 12 bytes (i.e. $s_h = 12$ bytes). Considering the sampling rate of 1 kHz, the haptic payload rate amounts to 96 kbps. Since the multiplexing depends heavily on the nature of audio/video frame generation, we consider two separate cases in our evaluation:

- **Case 1:** The audio payload rate is set to 64 kbps (G.711 PCM standards), packetized in blocks of size 20 ms as in the conventional VoIP systems [40]. Hence, $f_a = 50$ Hz and $s_a = 160$ bytes. The video payload rate is set to 400 kbps (H.264 encoder standards) generated at a frame rate of 25 Hz. Hence, $f_v = 25$ Hz and $s_v = 2000$ bytes. This results in $R_p = 560$ kbps, and consequently $s_m = 58$ bytes. Note that in this case f_a is an integral multiple of f_v.

- **Case 2:** In this case, we change the audio and video payload rates slightly to 72 kbps and 408 kbps, respectively. In order to achieve this, we set $f_a = 30$ Hz, $s_a = 300$ bytes, $f_v = 25$ Hz and $s_v = 2040$ bytes. This results in $R_p = 576$ kbps, and consequently $s_m = 60$ bytes. Note that in this case f_a is not an integral multiple of f_v.

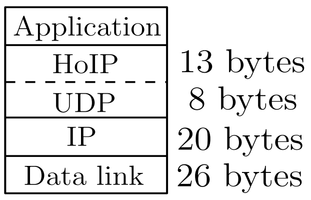


Figure 5.6: Representation of network stack model, along with the corresponding header size at each layer.

We reuse the network protocol stack diagram in Figure 5.6 featuring HoIP, described in Chapter 3. Adding the header due to each layer, we arrive at a net packet overhead of 67 bytes. Since the telehaptic packets are transmitted at the rate of 1000 per second, the resulting overhead rate $R_{oh} = 536$ kbps. Therefore, the overall telehaptic data rates due to our multiplexing schemes on the backward channel ($R_b = R_p + R_{oh}$) for Cases (1) and (2) are 1.096 Mbps and 1.112 Mbps, respectively.

Network Settings: We consider the scenario where the network bandwidth $\mu > R_b$. Of course, a real world network might be bandwidth constrained, and need not always support the peak telehaptic data rate. We will address the problem of network congestion control in Chapter 6, where the rate control scheme leverages the multiplexing schemes proposed in this chapter. For our evaluations, we consider the single bottleneck network topology shown in Chapter 4. We set

$\mu = 1.5$ Mbps, and remove the cross-traffic sources in the network. The one-way propagation delay (τ) is set to 15 ms.

5.3.2 Experimental Results

In this section, we report the multiplexing delay and jitter (both measured at the output of the multiplexer) encountered by the haptic, audio and video frames.

FCFS Multiplexing: Figures 5.7 and 5.8 show the haptic/audio/video delays due to FCFS multiplexing for the two cases described in Section 5.3.1. It can be seen that the delay QoS of all of the media types are comfortably within their respective limits. In both cases, the haptic samples encounter zero jitter due to their high multiplexing priority.

- **Case 1.** While the video frame encounters steady delays, the audio frame delays exhibit large fluctuations, incurring a large jitter of approximately 17 ms, as shown in Figure 5.7.
- **Case 2.** As shown in Figure 5.8, we measure a substantial audio jitter of 29 ms. The video frames also incur a marginal jitter of 1 ms.

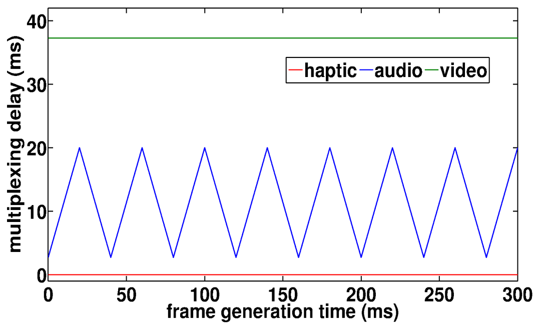


Figure 5.7: Multiplexing delays encountered by the haptic, audio and video frames under the FCFS rule for Case (1).

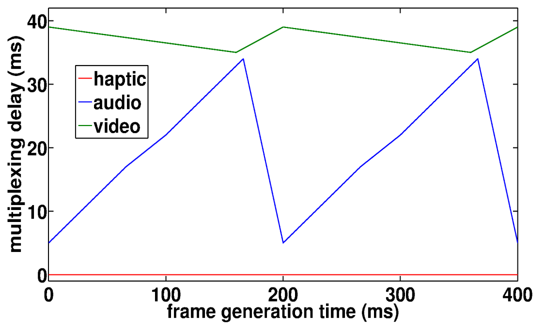


Figure 5.8: Multiplexing delays encountered by the haptic, audio and video frames under the FCFS rule for Case (2).

The reason behind the large audio jitter of the FCFS scheme is related to the frame rate and the frame size of audio relative to that of video. Typically, the video source generates large, but less frequent frames compared to the audio source. Therefore, the video frames consume a majority of the multiplexer cycles. Since the multiplexing is uninterrupted once a video frame is selected, the subsequent audio frames are queued at the input of the multiplexer until their turn. It is to be noted that amongst the queued up audio frames, the earlier frames suffer larger delays compared to the subsequent frames. This introduces a substantial audio jitter, sometimes even approaching the QoS limit (30 ms), making the audio media highly vulnerable to QoS violations.

Due to the significant audio jitter, the FCFS scheme is not a practical multiplexing solution for telehaptic applications.

QoS-Aware Multiplexing: Figures 5.9 and 5.10 show the haptic/audio/video delays due to QoS-aware multiplexing for the two cases described in Section 5.3.1. It can be seen that the delay QoS of all of the media types are comfortably within their respective limits. Similar to the FCFS scheme, the haptic samples encounter zero jitter in this scheme as well.

- **Case 1.** In Figure 5.9, we see that the audio jitter is completely eliminated due to multiplexer's sensitivity to audio frame generation. However, it is to be noted that because of the interruption to the video multiplexing the video delays increase slightly relative to the FCFS scheme (Figure 5.7).

- **Case 2.** As shown in Figure 5.10, the audio jitter is zero in this case as well. The video delay is slightly higher than the FCFS scheme (5.8), whereas the jitter is comparable.

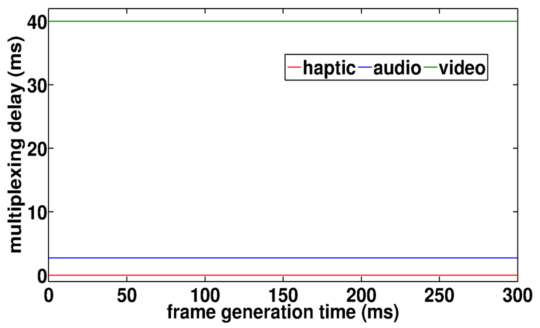


Figure 5.9: Multiplexing delays encountered by the haptic, audio and video frames under the QoS-aware rule for Case (1).

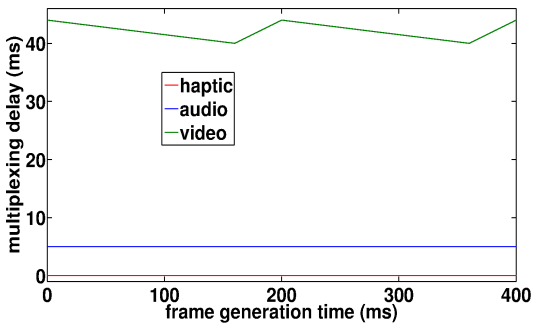


Figure 5.10: Multiplexing delays encountered by the haptic, audio and video frames under the QoS-aware rule for Case (2).

Indeed, the multiplexer's sensitivity towards audio frame generations works well by intelligently pushing the audio jitter (of FCFS) into the video delay. This is acceptable, since the video frames are more delay tolerant. Since the network also introduces additional jitter, the complete elimination of the audio jitter leaves a large room for the network jitter to be accommodated. This makes the telehaptic application more robust to the network irregularities. Therefore, a QoS-aware multiplexing scheme is more suitable for telehaptic applications than the FCFS based multiplexing scheme.

Tables 5.1 and 5.2 compare the haptic/audio/video delay and jitter under FCFS and QoS-aware multiplexing schemes for each of the two cases, highlighting the improvement of QoS-aware scheme over the FCFS scheme.

	Max. delay (ms)			Max. jitter (ms)		
	Hap.	Aud.	Vid.	Hap.	Aud.	Vid.
FCFS	0	20	37	0	17	0
QoS-aware	0	3	40	0	0	0

Table 5.1: Comparison of haptic/audio/video delays and jitter under FCFS and QoS-aware multiplexing schemes for Case (1).

	Max. delay (ms)			Max. jitter (ms)		
	Hap.	Aud.	Vid.	Hap.	Aud.	Vid.
FCFS	0	34	39	0	29	1
QoS-aware	0	5	44	0	0	1

Table 5.2: Comparison of haptic/audio/video delays and jitter under FCFS and QoS-aware multiplexing schemes for Case (2).

5.4 Conclusions

In this chapter, we described the design of multiplexing cum packetization module for backward channel telehaptic communication. We explained the rationale behind the design of the two multiplexing frameworks. While the FCFS-based scheme assigns multiplexing priority based on the frame generation timestamps, the QoS-aware scheme assigns multiplexing priority such that the media with a tighter QoS criteria receives a higher priority. Via simulations, we assess the performance of the proposed multiplexers in terms of delay and jitter of the media types. The results revealed that the QoS-oblivious nature of the FCFS scheme results in a large audio jitter, making the audio frames highly susceptible to jitter QoS violations. On the other hand, the QoS-aware scheme completely eliminates the audio jitter due to its sensitivity to QoS specifications. Hence, the QoS-aware multiplexing scheme is more suited for a delay-critical telehaptic application.

Chapter 6

Lossless Telehaptic Rate Control on Forward and Backward Channel

6.1 Introduction

In Chapter 4, we described an adaptive sampling based rate control scheme for network-aware telehaptic communication on the forward channel. In this chapter, we discover certain shortcomings of any adapting sampling based telehaptic communication scheme in general. In order to overcome these, we design a more refined rate control scheme that can be applied to both forward and backward channel, as shown in Figure 6.1.

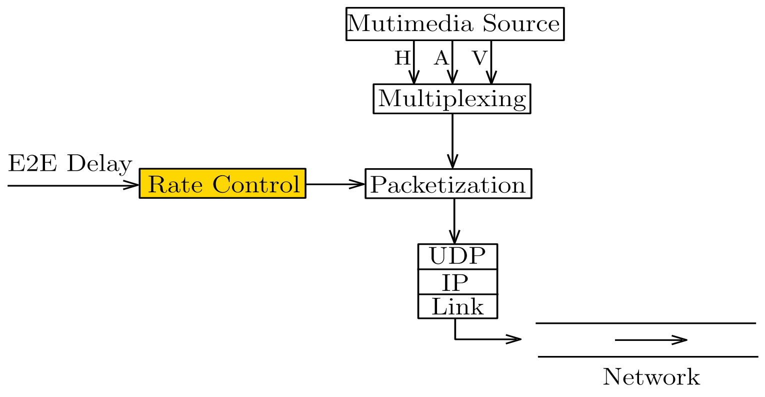


Figure 6.1: Architecture of Haptics over Internet Protocol (HoIP) focusing on the rate control aspect which forms the subject of this chapter. Note that the rate control proposed in this chapter is applicable to both forward and backward channel. The figure depicts HoIP on backward channel solely for the purpose of illustration.

We observe that by design, the instantaneous rate of any adaptive sampling based scheme on the forward channel is purely a function of the velocity of the operator's movements. Similarly, the instantaneous rate of the adaptive sampling scheme on the backward channel is a function of the physical properties of the remote objects. For example, rigid objects are known to generate fast varying force signals during the contact period. Therefore, the adaptive sampling scheme may not be an effective solution to telehaptic transmission on shared networks due to the following major drawbacks.

- It does not provide a fine-grained control over the instantaneous rates, and hence does not guarantee congestion control when the network is not provisioned for the peak rate.
- The scheme is inherently lossy in nature. This could be problematic on the forward channel due to robotic teleoperator (TOP) at the receiving end, as it is capable of executing even the slightest of the signal changes.

In the following discussion, we elaborate on the above drawbacks of an adaptive sampling based telehaptic communication, and motivate the need for a lossless, network-aware scheme with a fine-grained control over the instantaneous rate.

Suppose that the human user in a telehaptic activity carries out rapid hand movements. This produces a velocity signal that changes rapidly over time. Applying adaptive sampling on such a signal results in a large number of perceptually significant samples over that duration. The direct consequence of this is a large instantaneous rate (often reaching the peak rate) on the forward channel, as seen in Chapter 4. Similarly, haptic interaction with rigid objects causes a high instantaneous rate on the backward channel. The application of Weber sampler for haptic data, explored and evaluated in [45], reveals that only 25% of the velocity samples, and 10% of the force samples are perceptually significant. Discarding the perceptually insignificant samples achieves a substantial reduction in rate of around 75% and 90% on the forward and the backward channels, respectively.

The crucial observation we make here is that the effective rate reported in [45] is in a *long term average* sense. In other words, according to [45] the effective rate of the adaptive sampler is the instantaneous rate time-averaged over the entire duration of the experiment. Due to this, the large number of perceptually insignificant samples occurring during slow movements or soft object interaction nullify the effect of the intermittent overshoots in the instantaneous rate resulting in a remarkably low average data rate, as shown below. Figure 6.2 demonstrates the large swing of the instantaneous haptic data rate on the backward channel due to Weber sampler with $\delta = 0.1$, applied on a force signal recorded during the real-time telepottery activity. For the purpose of illustration, we ignore the corresponding audio and the video data.

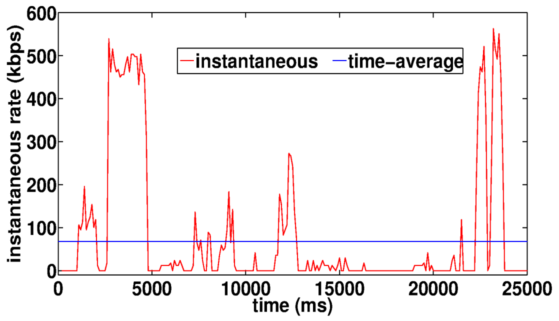


Figure 6.2: Plot of instantaneous rate on the backward channel with a Weber sampler threshold of $\delta = 0.1$. It can be seen that the plot shows large fluctuations with the instantaneous rate approaching the peak rate (592 kbps) occasionally.

It can be seen that in spite of the exceedingly low time-averaged data rate (around 70 kbps), the instantaneous rate varies between zero and the near peak rate (570 kbps). While this technique provides an elegant way to curtail the average transmission rate, it has no precise control over the instantaneous rate. Therefore in order to ensure QoS compliance, the shared network needs to be provisioned for the peak telehaptic rate throughout the telehaptic session. Hence, the Weber sampler (or in general, any adaptive sampling scheme) may not always provide real economies with respect to network bandwidth requirement. Furthermore, on the forward channel the usage of adaptive sampling may cause a loss of fidelity, as the TOP is typically a high precision robotic device that can execute even the slightest of the velocity changes.

To summarize, adaptive sampling, a lossy compression technique, may not be able to always guarantee telehaptic QoS adherence on a shared network, unless the network is provisioned for the peak instantaneous rate. This motivates the need for a *lossless*, *network-aware* rate adaptation technique with a fine-grained control over the instantaneous rate.

Under the peak rate transmission on the backward channel, the overhead due to packet headers from various layers of the network protocol stack (shown in Figure 6.3) accounts for almost half of the total telehaptic traffic. The packet overhead contributes an even larger proportion of the telehaptic traffic on the forward channel because of the presence of haptic data only. The large overhead is a direct consequence of the high packetization rate. In other words, the overhead rate is an increasing function of the packetization rate. Hence, there is a substantial scope for reducing the overhead rate by simply reducing the packetization rate. This observation leads us to the design of *Dynamic Packetization Module (DPM)* - a transport layer solution for telehaptic congestion control akin to Transmission Control Protocol (TCP) for elastic internet traffic.

In this chapter, we present DPM, a lossless transmission scheme that opportunistically varies the telehaptic data rate through tuning the packetization rate. Specifically, in congested networks, DPM dynamically merges successive telehaptic fragments into a single packet, thus adapting the overall transmission rate depending on the available network capacity. Under uncongested conditions, the proposed DPM prudently reverts to peak rate transmission, thereby maximizing the network utilization and minimizing the packetization delay. The fine-grained rate control, and the lossless nature of DPM overcomes the drawbacks of the adaptive sampling strategy. We test the proof-of-concept through extensive simulations, which reveal that DPM performs an efficient congestion control even under heavily congested network conditions, thereby guaranteeing telehaptic QoS conformance. The perceptual tests reveal that the proposed DPM provides a seamless telehaptic user experience even in a congested network. Further, we carry out mathematical analysis for characterization of the maximum haptic, audio and video delays under DPM based communication. Further, we compare DPM with other recently proposed telehaptic communication protocols, and demonstrate that DPM outperforms these protocols with respect to QoS compliance. Finally, we evaluate the impact of the variation in packetization rate on human perception via real-time telepottery experiment involving several human participants.

The rest of the chapter is organized as follows. Section 6.2 describes the working principle of DPM in detail. In Section 6.3, we discuss the setup for simulations and real-time telehaptic experiment designed for performance evaluation of the proposed DPM. We present our findings in Section 6.4, and analytically characterize the audio/video delays in Section 6.5. We state our conclusions in Section 6.6.

6.2 Dynamic Packetization Module

In this section, we describe the design of dynamic packetization module (DPM) in detail with respect to the backward channel. However, the design of DPM for forward channel is similar. Further, we perform a simple mathematical analysis for haptic jitter characterization of DPM.

6.2.1 Design

For convenience, we reuse the network protocol stack diagram (shown in Figure 6.3) that was presented in Chapter 3. Assuming Ethernet on the data link layer, the overall overhead per packet due to the link layer (OH_D = 26 bytes), IP, and UDP headers equals 54 bytes. Adding to this the HoIP overhead of 13 bytes (recall from Chapter 5), we arrive at a net overhead of 67 bytes/packet. If we transmit each telehaptic fragment as a separate packet (this corresponds to the peak packetization rate), this amounts to an overall overhead rate of 536 kbps. For a standard TOP payload rate (see Chapter 5) of 560 kbps (haptic - 96 kbps, audio - 64 kbps, video - 400 kbps), the overhead constitutes a substantial proportion (48.9%) of the telehaptic traffic. The overhead represents an even higher proportion (72.09%) of the telehaptic traffic on the backward channel, since the payload is composed of only haptic data.

Application	
HoIP	13 bytes
UDP	8 bytes
IP	20 bytes
Data link	26 bytes

Figure 6.3: Representation of TCP/IP stack model, along with the corresponding header sizes at each layer.

We use the QoS-aware multiplexing framework described in Chapter 5. Recall that the multiplexer packs s_m bytes of audio/video data with a haptic sample (of size s_h bytes) to form a telehaptic fragment once in every 1 ms, giving strict priority to audio over video. 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 (in kbps) corresponding to the k-merge packetization scheme is given by

$$R_k = R_p + \frac{R_{oh}}{k}, \tag{6.1}$$

where R_p is the telehaptic payload generation rate, and R_{oh} denotes the overhead rate under the no-merge scheme, both in kbps. Taking R_{oh} = 536 kbps, Figure 6.4 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

$R_p = 560$ kbps, we see that on the backward channel the telehaptic transmission rate for the no-merge scheme equals 1096 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.

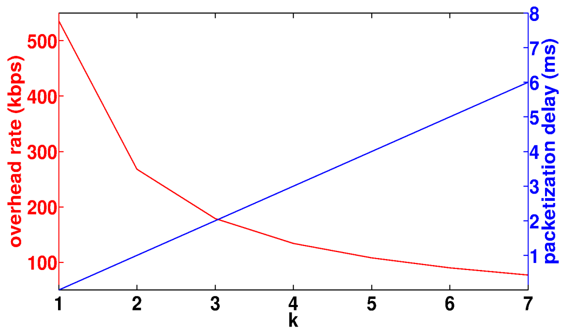


Figure 6.4: Telehaptic overhead rate variation for different k-merge packetization schemes, along with the corresponding packetization delay, demonstrating marginal rate reduction beyond $k = 4$.

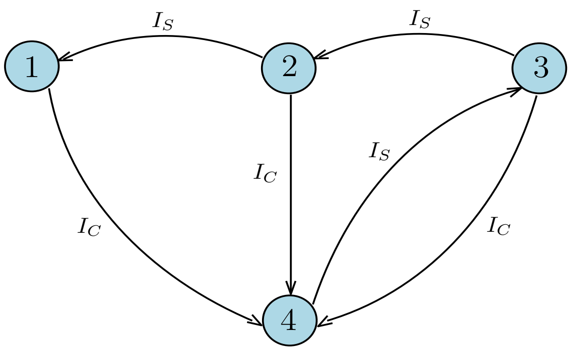


Figure 6.5: A finite state transition diagram representation of the step increase, multistep decrease mechanism of DPM with $k_{max} = 4$. I_C and I_S denote the congestion and the steady state triggers, respectively.

The idea behind DPM is to dynamically adapt the packetization parameter k depending on the current network conditions. In other words, DPM dynamically switches between different k-merge schemes based on the triggers from the network feedback module (described in Chapter 4). From Figure 6.4, 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 [23]. Figure 6.5 shows a finite state transition diagram representation of DPM. On receiving the trigger I_C (recall from Chapter 4 that this trigger signals that the network is getting congested), DPM sets $k = k_{max}$. Thus, on sensing congestion in the network, DPM decreases the telehaptic data rate abruptly. Such an aggressive rate reduction causes the queues to be flushed out quickly, thereby helping in decongesting the network in the shortest possible time. On the other hand, on receiving the trigger I_S (recall from Chapter 4 that this trigger signals that the network is uncongested), 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. Such prudent network probing avoids persistent congestion that is more likely to be induced by an aggressive rate increase.

6.2.2 Haptic Jitter Characterization

DPM's dynamic packet rate adaptation introduces additional jitter at the receiver. To get a sense of the jitter caused by DPM, we carry out the following simple analysis, focusing only on haptic jitter as 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 6.6. Suppose that initially, $k = 1$. Note that sample $m+1$ is generated

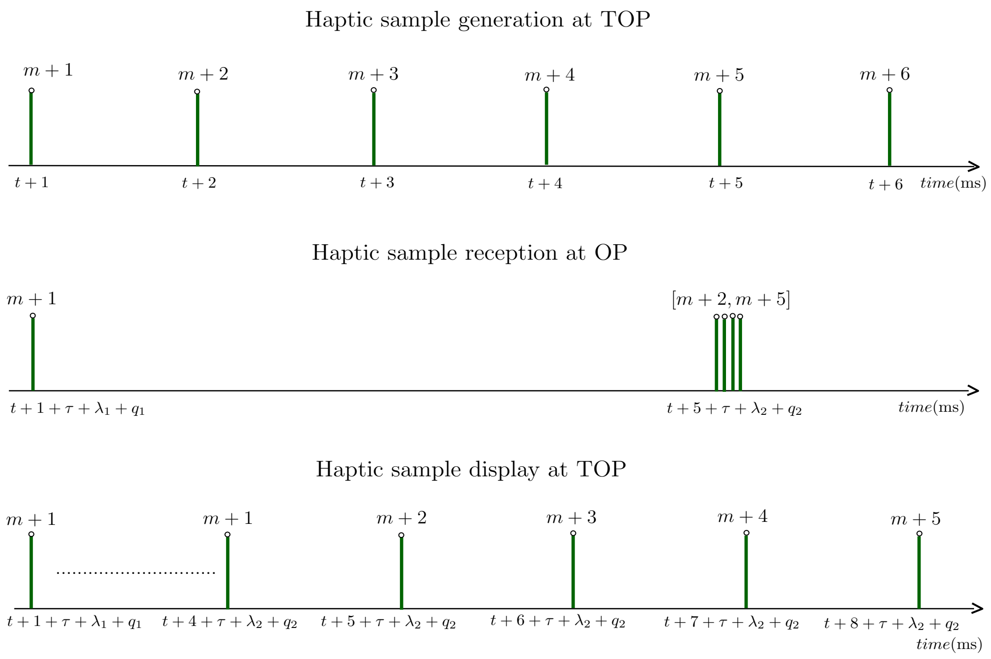


Figure 6.6: Timing diagram illustrating haptic sample transmission at TOP, reception and display at OP using zero-order hold strategy. The samples bunched together indicate simultaneous reception due to the 4-merge packet.

at time $t + 1$, and is received and displayed at time $t + 1 + \tau + \lambda_1 + q_1$. Here, τ denotes the one way propagation delay, and q_1 and λ_1 denote the queueing delay and the transmission delay encountered by the packet containing sample $m + 1$, respectively. Now, suppose that starting from sample $m + 2$, 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 $t + 5 + \tau + \lambda_2 + q_2$. Here, q_2 and λ_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:

$$(t + 5 + \tau + \lambda_2 + q_2) - (t + 2 + \tau + \lambda_1 + q_1) = 3 + (q_2 - q_1) + (\lambda_2 - \lambda_1).$$

Note that $\lambda_2 \leq 4\lambda_1$. Assuming then that q_1 and q_2 are comparable, we can bound the jitter by $3(1 + \lambda_1)$. Thus, we see that by restricting k to be at most 4 under DPM, we introduce an additional jitter of at most $3(1 + \lambda_1)$ on the haptic stream. Note that the subsequent 4-merge packet (carrying haptic samples $[m + 6, m + 9]$) arrives at $t + 9 + \tau + \lambda_2 + q_2$. This means that the subsequent 4-merge packets face no additional jitter. We validate the correctness of the upper bound expression for the jitter through simulations, the findings of which are reported in Section 6.4.1.

6.2.3 The Communication Framework

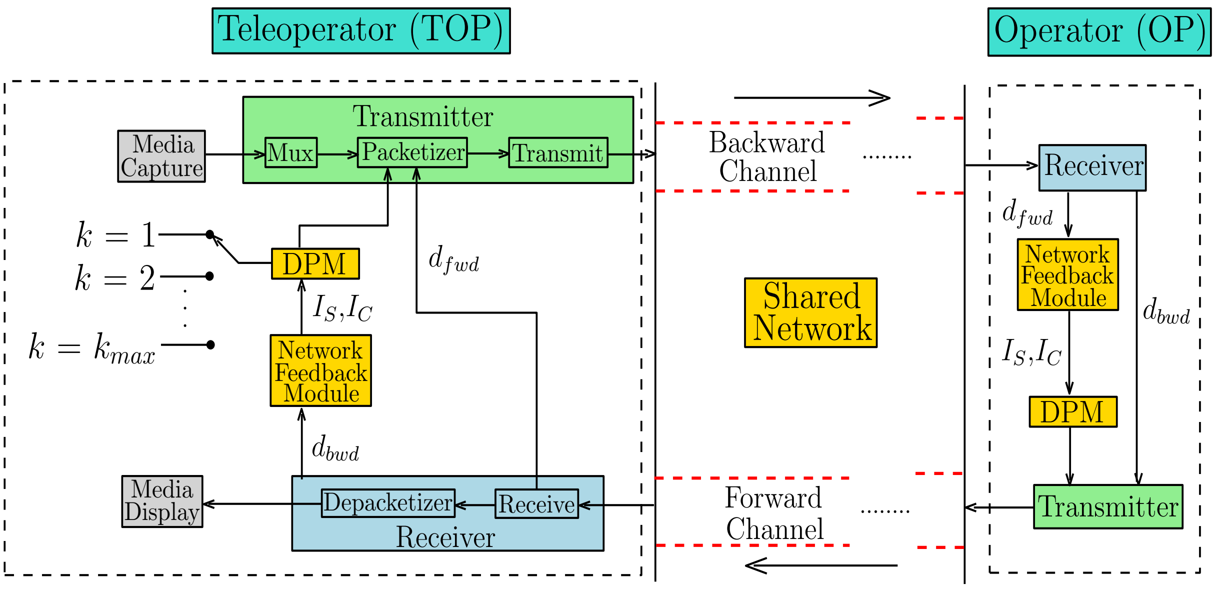


Figure 6.7: A block diagram showing the architecture of the proposed telehaptic communication framework. The design at the OP is similar to that of the TOP, and is not shown for brevity.

Figure 6.7 presents the telehaptic communication framework featuring DPM. For brevity, we describe the framework with respect to the TOP. The architecture at the OP remains similar. After 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 end-to-end (E2E) delay (d_{bwd}) (described in Chapter 5), which is a part of the header, is supplied to the *network feedback module* (described in Chapter 4) for learning the recent changes in the backward channel conditions. Based on the delay analysis, the network feedback module generates triggers (I_S, I_C) appropriately. On arrival of a trigger, the DPM updates k, which is communicated to the *packetizer* for composing the telehaptic packets. The TOP also calculates the forward channel E2E delay (d_{fwd}) after every packet reception, which is sent to the *packetizer* for inclusion in the outgoing packet header, transmitted to the OP on the backward channel. It is important to remark that by design, DPM is insensitive to the type of telehaptic application and the media devices being employed.

6.3 Experimental Design

In this section, we describe the setup used in our experiments for performance evaluation of DPM. The objective of the experiments is to investigate the ability of DPM to perform congestion control under heavy cross-traffic scenarios. We consider the following performance metrics: QoS adherence, signal-to-noise ratio (SNR) of the reconstructed haptic signal at the receiver, and the perceptual quality of the displayed haptic-audio-video signal. We first describe the setup used in our simulations, and then move to the the real-time telepottery experimental setup. The results of these evaluations follow in Section 6.4.

6.3.1 Simulation Setup

Our simulations are carried out using NS-3, a discrete event network simulator [74]. We consider a network with a single bottleneck dumbbell topology connecting the OP and the TOP, as shown in Figure 6.8. In order to simulate asymmetric networking conditions on the forward and the backward channels, we create unidirectional links between the OP and the TOP nodes. To simulate cross-traffic on the forward (respectively, backward) channel, we add source-destination pairs (c_i, d_i) (respectively, (d_j, c_j)) as indicated in Figure 6.8. Note that l_1 and l_2 act as the bottleneck links for the telehaptic traffic on the forward and backward channels, respectively. Thus, queueing delay experienced by the telehaptic application due to network

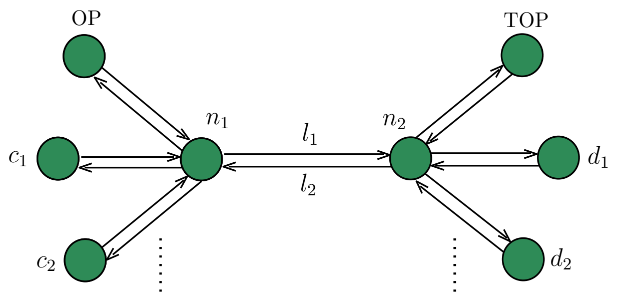


Figure 6.8: Single bottleneck dumbbell network topology for performance evaluation of the proposed DPM. c_1, c_2, d_1 and d_2 represent the cross-traffic nodes; l_1 and l_2 represent the bottleneck links on the forward and the backward channels, respectively. n_1 and n_2 represent the intermediate nodes on the forward and the backward channels, respectively.

cross-traffic is observable only at the intermediate nodes n_1 and n_2.

Since our QoS measurements are insensitive to the payload itself, we use simulated haptic/audio/video data for our simulations, unless otherwise stated. Of course, our perceptual experiments (described in Section 6.3.2) are performed with real haptic/audio/video data. The haptic payload rates on the forward and backward channels are set to 192 kbps and 96 kbps (as already mentioned in Chapters 3 and 5), respectively. The audio and the video source parameters (see Chapter 5) are set as follows: s_a = 160 bytes, f_a = 50 Hz, s_v = 2000 bytes, f_v = 25 Hz. This corresponds to audio and video payload rates of 64 kbps and 400 kbps, respectively, on the backward channel. Considering the network protocol layer header sizes (see Figure 6.3), the no-merge data rate on the forward and backward channels are calculated to be 688 kbps and 1096 kbps, respectively.

Finally, the propagation delays of each of the links l_1 and l_2 are set to 15 ms. The bottleneck link capacity is set to μ = 1.5 Mbps. This value 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. The access links connecting the nodes n_1 and n_2 to the traffic sources and receivers have very high bandwidth and zero propagation delay. Hence, the one-way propagation delay τ = 15 ms, which is typically the propagation delay exhibited by a transcontinental link of around 3200 kms. All nodes follow first-in-first-out (FIFO) and droptail queueing of packets.

6.3.2 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 the subjective evaluation of DPM, we use the real-time telepottery experiment described in Chapter 4.

Telepottery experiment: The experiment involves a human subject interacting with a remote, virtual pottery model on a real network through haptic, audio and visual feedback, as described in Chapter 4. The volume preserving pottery model [21] 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 Chapter 4 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 them the telepottery task 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 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 the following three perceptual parameters:

- *transparency:* the subjects felt as if they were present in the remote virtual environment and are directly interacting with the objects [58]
- *smoothness:* how smooth or jerky is the feedback [52]
- *overall experience.*

The grading of each of the three parameters was based on degradation category rating (DCR) [34,51,99] that assigns a subjective scale to a text descriptor, as shown in Table 6.1. 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.

Grade	Description
5	imperceptible
4	slight disturbance, but not annoying
3	slightly annoying
2	annoying
1	very annoying

Table 6.1: Degradation category rating for subjective evaluation of the real-time telepottery experiment.

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$\times$100 at 25 fps), 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 [82], with its clock rate configured to 1 kHz. The training phase of the telepottery experiment is performed with $\mu = 100$ Mbps and $\tau = 15$ ms. For the testing phase, the emulated network is configured to $\mu = 10.2$ Mbps and $\tau = 15$ ms, respectively, for both the forward and the backward channel. In the testing phase, we introduce constant bit rate (CBR) cross-traffic stream of intensity $R_{cbr} = 400$ kbps on the backward channel. In addition, we introduce 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.

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 twenty subjects (ten female and ten male, eighteen right-handed and two left-handed) participated in the perceptual task. The subjects belonged to the age group of twenty three to fifty two years, and none of them suffered from any known neurophysiological disorders. Out of the twenty participants, fourteen were novice haptic users and the rest were regular users of haptic devices. However, all subjects underwent appropriate training prior to the test experiments.

6.4 Performance Evaluation

In this section, we present a comprehensive experimental evaluation of DPM. Simulation results are presented in Section 6.4.1, and the results of our perceptual experiments are presented in Section 6.4.2. In Section 6.4.3, we compare the performance of DPM with the state of the art in telehaptic communication protocols.

6.4.1 Simulation Results

We begin by presenting 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 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 VBR stream on backward channel with intensity $R_{vbr} \in$ [320, 480] kbps with a mean of 400 kbps. 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. At $t = 500$ ms, we additionally introduce CBR cross-traffic stream on the backward channel. The intensity of the CBR traffic 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 $R_1 =$ 1096 kbps on the backward channel, whereas the minimum data rate $R_4 =$ 694 kbps. With $\mu =$ 1.5 Mbps and $R_{cbr} >$ 4 kbps, the telehaptic transmission at R_1 leads to congestion, and hence peak rate transmission is infeasible. On the other hand, $R_{cbr} >$ 406 kbps implies that the network is overloaded, since the transmission at R_4 also leads to congestion. Hence, for the current network setting, $R_{cbr} =$ 406 kbps is the upper bound on the CBR cross-traffic such that DPM's congestion control stabilizes the bottleneck link. 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. In others, we set $R_{cbr} =$ 260 kbps for a finer demonstration of the working of DPM. 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, i.e., over the interval $t \in$ [0.5, 500] seconds.

Temporal evolution of telehaptic delay: We begin by demonstrating the temporal evolution of the telehaptic delays under DPM. Figures 6.9 and 6.10 show the delay experienced by the haptic samples as a function of the sample generation time, corresponding to R_{cbr} = 260 kbps and 400 kbps, respectively.

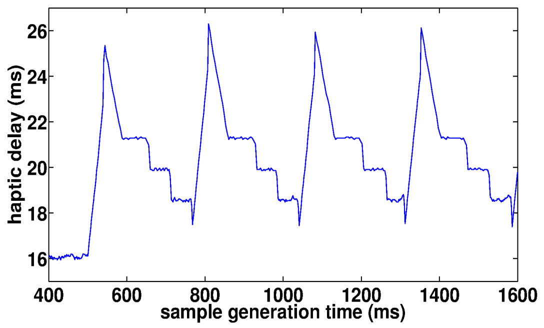


Figure 6.9: Temporal evolution of the haptic delay as a result of DPM in presence of $R_{cbr} =$ 260 kbps.

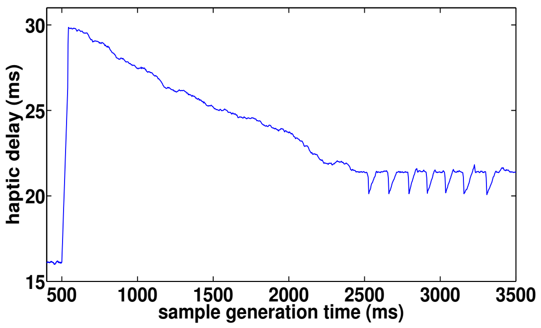


Figure 6.10: Temporal evolution of the haptic delay as a result of DPM in presence of $R_{cbr} =$ 400 kbps.

Let us first consider Figure 6.9. For R_{cbr} = 260 kbps, the capacity available to the telehaptic stream on the backward channel equals 840 kbps. It can be computed that $R_2 = 828$ kbps, and hence $R_2 <$ 840 kbps $< R_1$ (1096 kbps). Once the CBR source turns on at $t =$ 500 ms, the telehaptic stream, initially operating at $k = 1$, sees a rapid delay build-up. DPM responds to this delay build-up 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 instantaneous transmission rate once again exceeds the capacity of the bottleneck link. This in turn leads to a delay build-up, and the cycle repeats.

Figure 6.10 has a similar interpretation. For R_{cbr} = 400 kbps, the capacity available to the telehaptic stream on the bottleneck link equals 700 kbps. It can be computed that $R_3 = 739$ kbps, and hence R_4 (694 kbps) $<$ 700 kbps $< R_3$. In this case, the switch from $k = 4$ to $k = 3$ causes a delay build-up, 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 profile. This is typical of congestion control algorithms; see, for example, [42]. Note that even when the backward channel is highly congested (see Figure 6.10), DPM manages to maintain the haptic delays below the prescribed QoS limit of 30 ms.

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 6.11 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 $k < k_{max} = 4$). For this experiment, we set R_{cbr} = 400 kbps. Once the CBR stream starts transmission, the telehaptic stream, initially operating at $k = 1$, experiences a rapid delay build-up 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 hand, the multistep-increase approach cuts the transmission rate in stages, requiring three rate adaptations before setting $k = 4$. As a result, the network decongestion occurs much later, leading to a violation of the haptic delay QoS constraint. Thus, we conclude that DPM's SIMD approach is suitable for congestion control for delay-critical telehaptic applications.

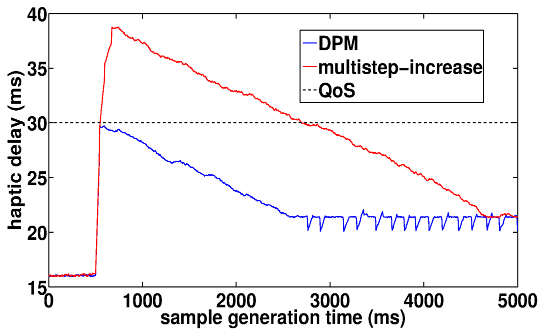


Figure 6.11: Comparison of haptic delay evolution of DPM and multistep-increase approaches, in presence of R_{cbr} = 400 kbps, demonstrating the effectiveness of DPM.

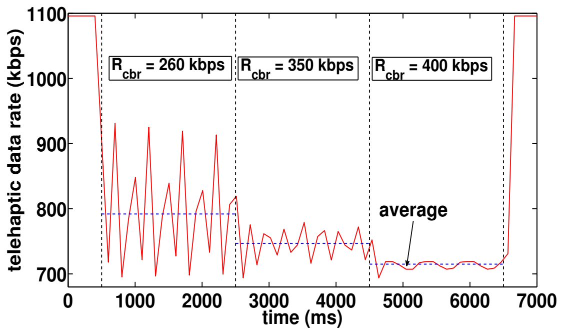


Figure 6.12: Telehaptic source rate evolution under time-varying cross-traffic conditions showing that DPM is extremely friendly to CBR cross-traffic sources.

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 (6.2).

$$R_{cbr} = \begin{cases} 0, & \text{for } 0 < t \leq 500 \text{ ms} \\ 260 \text{ kbps } (C_1), & \text{for } 500 \text{ ms} < t \leq 2500 \text{ ms} \\ 350 \text{ kbps } (C_1 \text{ and } C_2), & \text{for } 2500 \text{ ms} < t \leq 4500 \text{ ms} \\ 400 \text{ kbps } (C_1, C_2 \text{ and } C_3), & \text{for } 4500 \text{ ms} < t \leq 6500 \text{ ms} \\ 0, & \text{for } t > 6500 \text{ ms} \end{cases} \tag{6.2}$$

Figure 6.12 shows the temporal variation of DPM source rate. Until 500 ms, DPM achieves its peak rate since the network in 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 withdrawn at t = 6500 ms, DPM reverts to its peak rate. Thus, we see that DPM performs a robust congestion control under time-varying cross-traffic settings, thereby exhibiting a highly

CBR cross-traffic friendly behavior.

	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

Table 6.2: Comparison of the telehaptic delay and jitter observed for different media for R_{cbr} = 400 kbps, along with the corresponding QoS specifications.

Telehaptic delay and jitter measurements: Table 6.2 summarizes the maximum telehaptic delay and jitter observed 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 application to comply with the QoS limits. Note that the maximum haptic jitter of 3.628 ms corroborates well with the analytical expression derived in Section 6.2.2. All our measurements prove that indeed DPM maintains the telehaptic delay and jitter below the prescribed QoS specifications.

Haptic signal reconstruction: We now study the effects of cross-traffic sources, DPM and data extrapolation on the haptic signal reconstruction at the OP. We compare the reconstructed signal with that corresponding to an adaptive sampling 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. Twenty 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 $\delta = 0.1$ for adaptively sampling the force signal at the TOP. We use the standard zero-order hold strategy for haptic data extrapolation, described in Chapter 3. 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 6.13 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

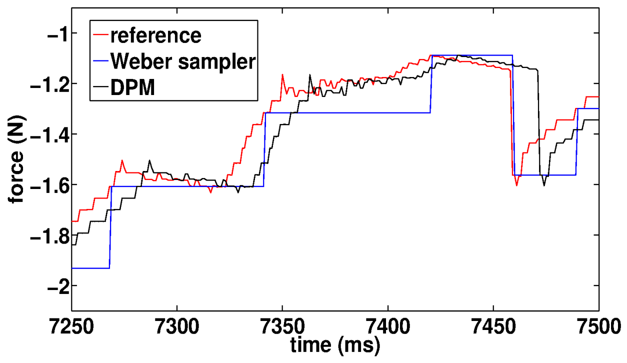


Figure 6.13: Graph showing the reconstructed force signals at OP with Weber sampling and DPM for $R_{cbr} = 400$ kbps.

sampled signal is a piecewise constant approximation of the reference signal. It is to be noted that under the Weber sampling strategy, *perceptually significant* samples are displayed earlier at the OP as compared to DPM. This is because of higher packetization and transmission delays that the packets encounter under DPM. Using SNR as a performance metric to measure the

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

Table 6.3: Comparison of SNR (in dB) in case of Weber sampler and DPM, with $R_{cbr} = 400$ kbps on backward channel.

reconstruction error at the OP (against the reference signal), Table 6.3 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 2.54 dB over the adaptive sampling strategy. In our experiments, we have found a comparable SNR improvement for other pilot telehaptic traces as well.

Throughput-Loss Measurements: Figures 6.14 and 6.15 compare the performances of DPM and no-merge scheme, in terms of their throughputs and packet losses, under various CBR 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 well before the network queues start to overflow 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.

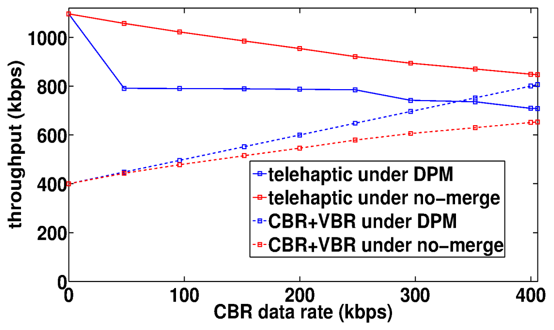


Figure 6.14: Telehaptic-CBR traffic interplay demonstrating the throughput improvement of DPM over no-merge.

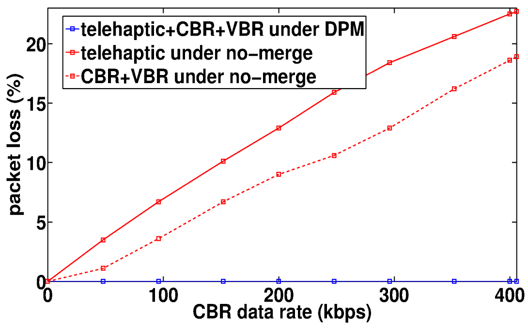


Figure 6.15: Telehaptic-CBR traffic interplay demonstrating the packet loss improvement of DPM over no-merge.

Figure 6.15 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 quickly adapting the telehaptic data rate to the cross-traffic intensity. We note that DPM is friendly to CBR and VBR cross-traffic. Indeed, the cross-traffic streams see full throughput (and hence zero loss) under DPM as opposed to the high packet losses under the no-merge scheme.

DPM with hold-up: Motivated by Figure 6.9, we propose a variant of DPM that seeks to reduce the jitter induced by frequent rate adaptations. Recall that in the experiment corresponding to Figure 6.9, 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 $k = 1$ immediately, 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 design 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 corresponding to the switch $\hat{k}+1 \to \hat{k}$. 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.

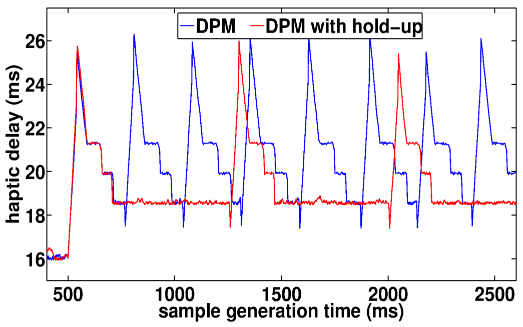


Figure 6.16: Haptic delay plots for DPM with and without hold-up techniques, in presence of R_{cbr} = 260 kbps.

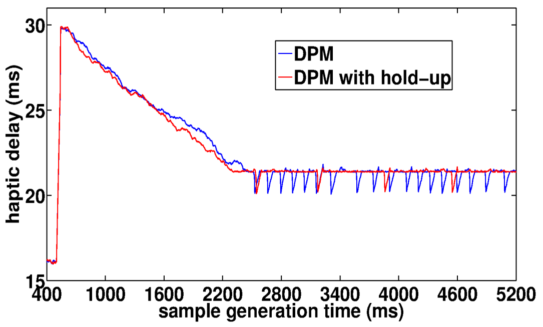


Figure 6.17: Haptic delay plots for DPM with and without hold-up techniques, in presence of R_{cbr} = 400 kbps.

Figures 6.16 and 6.17 show the haptic delay variation plots for DPM and DPM with hold-up in 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 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.

6.4.2 Telepottery Subjective Grading

We now move to the qualitative results of the real-time telepottery task. Figure 6.18 presents the mean opinion score (MOS) of the degradation category rating (DCR) recorded with twenty human subjects for each of the three perceptual parameters i.e., transparency, smoothness and overall experience. It can be seen 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

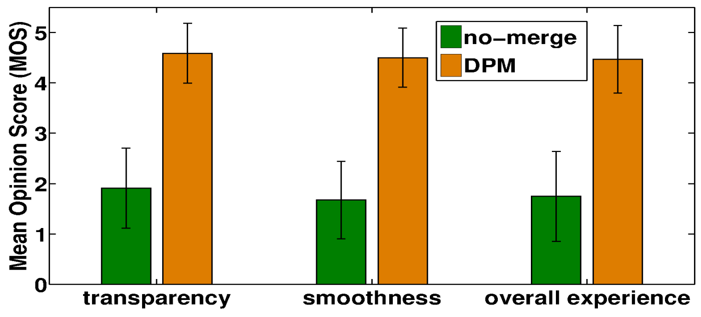


Figure 6.18: MOS of the subjective evaluation of the proposed technique on three specific perceptual parameters, averaged over twenty human subjects. The vertical bars denote the standard deviation of the subject grades.

the no-merge experience so disturbing that they hardly made any contact with the clay model throughout the duration of the experiment. On the other hand, 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 compared to the reference (training experience).

We also statistically evaluate the improvement in perception of DPM over the no-merge scheme. For this purpose, we resort to paired t-test performed over the subject grades recorded from twenty participants. The test results for the three perceptual parameters are as follows:

- transparency: $t(19) = 10.81, p < 0.001$
- smoothness: $t(19) = 13.97, p < 0.001$
- overall experience: $t(19) = 11.72, p < 0.001$,

where $t(19)$ indicates the ratio of inter-group and intra-group variances with a sample size of 20, and p indicates the probability of null hypothesis being true. These results indicate that the DPM outperforms no-merge with an extremely high confidence (1-p). 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 an enjoyable telepottery experience.

6.4.3 Comparison With Existing Telehaptic Communication Techniques

In this section, we compare the performance of DPM with RTP [53] and state of the art protocols for telehaptic communication.

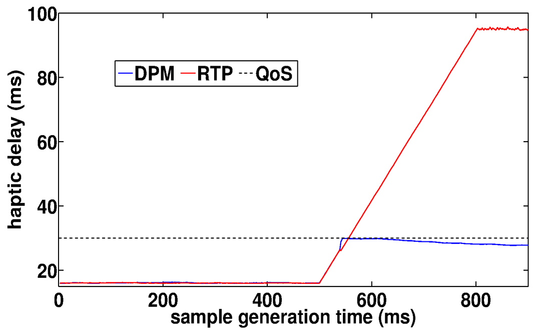


Figure 6.19: Demonstration of early congestion detection and responsiveness of DPM leading to QoS compliance as opposed to sluggish behavior of RTP causing QoS violations with $R_{cbr} = 400$ kbps.

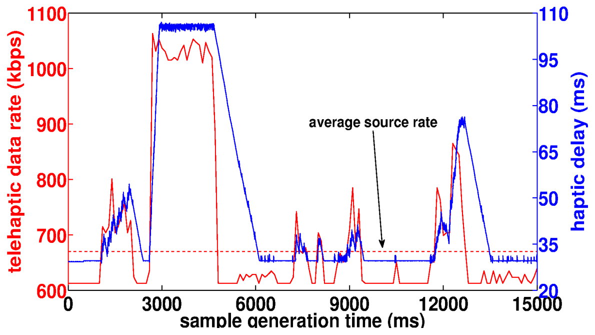


Figure 6.20: Rate-delay plot of visual-haptic multiplexing on the backward channel demonstrating severe delay QoS violations when the network is provisioned for the average data rate.

1) Real-time transport protocol (RTP): We begin by comparing the quality of congestion control of DPM with that of RTP [53], which is the dominant protocol for media streaming applications on the internet. We use the simulation setup from Section 6.4.1, with $R_{cbr} = 400$ kbps on the backward channel. Figure 6.19 shows the variation in delay encountered by the haptic samples versus their corresponding sample generation times. Note that once the CBR cross-traffic is introduced at 500 ms, DPM performs a prompt rate adaptation, maintaining haptic delays below the prescribed 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 6.19, severely 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. The flat delay zone beyond 800 ms indicates that the queue is full and overflowing.

2) Visual-haptic multiplexing: We now evaluate DPM against the visual-haptic multiplexing scheme [24], which employs the Weber sampler for force updates on the backward channel. We use the simulation setup from Section 6.4.1 with R_{cbr} = 400 kbps for this evaluation as well. Figure 6.20 shows the source rate evolution and the resulting delays under visual-haptic multiplexing obtained using one of the pilot 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 of the Weber sampler (670 kbps), the instantaneous rate of the Weber sampler fluctuates substantially, resulting in occasional QoS violations; for example, see the interval from 3000 to 6000 ms. Indeed, during times of interaction with a hard object, almost every force sample becomes perceptually significant, causing a Weber sampler's instantaneous transmission rate to far exceed its time-average value. 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 6.4.1 show that DPM meets the QoS constraints and results in zero packet loss.

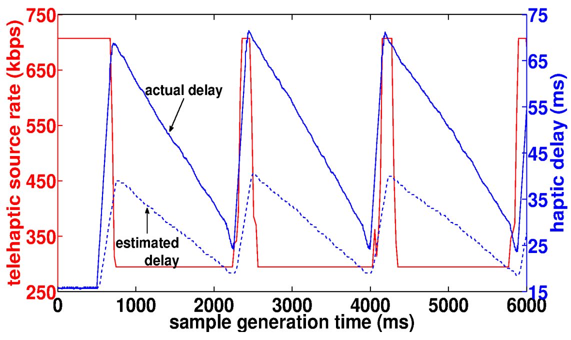


Figure 6.21: Telehaptic rate-delay plot of NAFCAH on forward channel.

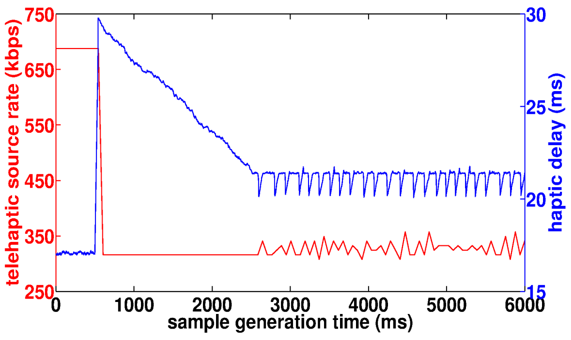


Figure 6.22: Telehaptic rate-delay plot of DPM on forward channel.

3) Network adaptive flow control algorithm for haptic data (NAFCAH): We now compare the performance of NAFCAH [57], a protocol that performs RTT-based rate adaptation on the *forward channel*, with DPM. We use the simulation setup from Section 6.4.1, except that the CBR cross-traffic intensity on the 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 6.21 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 already discussed in

Section 6.4.1, 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 6.21.

In contrast, as seen in Figure 6.22, DPM satisfies the QoS constraints well under the same network conditions, thanks to its aggressive *step-increase* mechanism for rate reduction, and the accurate end-to-end delay estimation mechanism by the network feedback module (see Chapter 4).

6.5 Delay Characterization

In Section 6.2.2, we characterized the jitter experienced by the haptic samples under DPM. Having understood the rate adaptation of DPM in detail (through design description and simulation results), we now present some theoretical analysis of the DPM behavior. The objective of this analysis is to characterize the delays encountered by the haptic-audio-video samples under DPM. Further, this serves to identify the class of network configurations where QoS-compliant telehaptic communication is feasible. In this section, we derive expressions for the maximum delay experienced by haptic samples under DPM over a single bottleneck network topology (see Figure 6.8) in presence of CBR cross-traffic. Note that since our protocol operates at the transport layer (TL), we characterize the maximum TL-TL latency, i.e., 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. We start by characterizing the haptic delay, and then move to the audio and the video delay analysis.

6.5.1 Haptic Delay

Recall that R_{cbr} denotes the CBR cross-traffic intensity on the channel under consideration and μ denotes the bottleneck link capacity, both in kbps. For simplicity, we assume that the reverse channel is uncongested, so that the packetization rate on the reverse channel equals 1 kHz. Let k_{opt} denote the minimal k that guarantees steady network state. Therefore, by definition

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

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 queue build-up 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 I_C, and T_h denote the sampling period of the haptic signal (1 ms in this case). In other words, 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. Therefore, we can write an expression for d_{inc} as follows.

$$d_{inc} = N(k_{opt} - 1)T_h + \frac{N(R_{cbr} + R_{k_{opt}-1} - \mu)}{\mu}(k_{opt} - 1)T_h + 2\tau + T_h. \tag{6.3}$$

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 (2τ) represents the round trip propagation delay, and the fourth term represents the maximum time gap between arrival of the N^{th} packet at the receiver and the piggybacking of its corresponding delay on the reverse channel.

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

$$q_{inc} = (R_{cbr} + R_{k_{opt}-1} - \mu)d_{inc}. \tag{6.4}$$

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 haptic delay:

$$d_{hap} = \tau + \frac{q_{inc}}{\mu} + (k_{opt} - 1)T_h. \tag{6.5}$$

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 Equations (6.3),(6.4) and (6.5), we obtain:

$$\begin{aligned} d_{hap} &= \tau + (k_{opt} - 1)T_h + [N(k_{opt} - 1)T_h + 2\tau + T_h]\left(\tfrac{R_{cbr}+R_{k_{opt}-1}-\mu}{\mu}\right) \\ &\quad + N(k_{opt} - 1)T_h\left(\tfrac{R_{cbr}+R_{k_{opt}-1}-\mu}{\mu}\right)^2. \end{aligned} \tag{6.6}$$

Note that Equation (6.6) enables us to characterize the set of link capacities, propagation delays, and cross-traffic intensities that satisfy the haptic delay QoS constraints. Since 30 ms is the upper bound on the haptic delay, d_{hap} can be set to 30 ms in Equation (6.6) to derive conditions on the aforementioned network parameters which guarantee that the maximum haptic delay does not exceed 30 ms.

As an example, let us consider a network with $\tau = 25$ ms and $\mu = 1.5$ Mbps. Let us take $R_{cbr} = 660$ kbps. As seen in Section 6.4, $k_{opt} = 2$ for this case. We can calculate from Equation (6.6) that $d_{hap} = 41.49$ ms. Now setting $d_{hap} < 30$ ms gives us an upper bound on R_{cbr} that satisfies the haptic delay constraint. Using Equation (6.6) we find that the sufficient condition for haptic delay QoS compliance is $R_{cbr} < 443$ kbps. Hence, this condition can be used by the network monitoring for limiting the CBR cross-traffic in order to ensure haptic delay compliance. In order to validate our analysis, we set $R_{cbr} = 440$ kbps, and measure the maximum haptic delay to be 29.514 ms. It can also be seen that for a given R_{cbr}, the link parameters (τ, μ) can be tuned to achieve haptic delay QoS compliance. We test our analysis for various values of network parameters, and in all the cases we note that the results corroborate well with our analytical estimates. The above analysis helps the network administrator to configure the network in order to guarantee satisfaction of haptic delay constraint.

In the following section, we will relate the maximum end-to-end haptic delay to the maximum end-to-end delay seen by the audio and video streams, and derive conditions for meeting the audio/video delay requirements.

6.5.2 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 6.8) with CBR cross-traffic. Similar to the haptic delay, we only consider the maximum TL-TL latency. Interestingly, these upper bounds involve the maximum haptic delay d_{hap} characterized in Section 6.5.1. Thus, we are able to relate haptic QoS compliance to QoS compliance for audio and video.

Recall from Chapter 5 that f_a and f_v represent the peak frame rates (in frames per second) 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 maximum size of the audio/video data (in bytes) in each telehaptic fragment.

As seen in the previous chapter, the instant an audio frame is generated, the previous audio frame would have 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 $k_{max} - 1$ ms. Finally, the maximum end-to-end delay experienced by the packet is equals d_{hap}. This yields the following upper bound on the TL-TL audio frame delay.

$$d_{aud} \leq d_{hap} + \frac{s_a}{s_m}(1\text{ms}) + (k_{max} - 1)(1\text{ms}). \tag{6.7}$$

Next, we move to the maximum delay experienced by a video frame (TL-TL). 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 frame delay.

$$d_{vid} \leq d_{hap} + \frac{1}{f_v} + (k_{max} - 1)(1\text{ms}) \tag{6.8}$$

From Equations (6.7) and (6.8), we can compute the maximum delay seen by audio/video frames assuming that the QoS constraint on the haptic delay is satisfied, i.e., $d_{hap} <$ 30 ms. Consider the settings assumed in our simulations: $f_a = 50$ fps , $f_v = 25$ fps , $s_a = 160$ bytes , $s_v = 2000$ bytes. This leads to $s_m = 58$ bytes. It then follows from Equations (6.7) and (6.8) that $d_{aud} \leq$ 35.75 ms, $d_{vid} \leq$ 73 ms. Note that these bounds are well below the audio/video QoS delay deadlines. Thus, under DPM, meeting the (strict) haptic delay constraint in general implies compliance with the audio/video delay constraint. Hence, no special attention needs to be given for meeting the audio/video delay constraints, as long as haptic delay QoS is satisfied.

6.6 Conclusions

In this chapter, we presented DPM, a lossless network-aware application layer protocol for 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. We compared DPM's performance with the state of the art in telehaptic protocols, and demonstrated that DPM outperforms them. Finally, we derived sufficient conditions for a given network setting that guarantees haptic delay adherence. Further, we extend the analysis to audio/video, and showed that haptic delay compliance implies audio/video compliance as well.

We tested DPM's proof of concept in presence of cross-traffic which was CBR in nature. A real world network, like the Internet, carries a variety of traffic flows which are not necessarily CBR in nature. We consider a heterogeneous cross-traffic setting in Chapter 7, where we analyze the interplay between network cross-traffic and a broad class of telehaptic protocols (not restricted to HoIP).

Chapter 7

Interplay Between Telehaptic Flows and Network Cross-Traffic

7.1 Introduction

In Chapters 4 and 6, we designed two rate control protocols, and carried out their performance evaluation in presence of steady or slowly varying cross-traffic conditions. A real world shared network, like the Internet, carries a wide variety of traffic flows which may be themselves rate-adaptive in nature. Transmission Control Protocol (TCP), for example, is the predominant rate-adaptive protocol that contributes to around 90% of the overall Internet traffic [85,112]. The rest of the traffic is primarily attributed to real-time teleconferencing applications, such as Skype [2] and Google Hangout [7], that use User Datagram Protocol (UDP) at the transport layer, and are not necessarily rate-adaptive in nature. Therefore, it is important to evaluate the performance of telehaptic protocols under heterogeneous cross-traffic conditions, and understand the implications of exogenous traffic flows on the feasibility of telehaptic QoS compliance under shared network conditions.

Even though several protocols have been designed for telehaptic communication, their performance evaluation has been carried out in highly controlled and simplistic network settings, which are far away from being realistic. Because of the large volume of TCP traffic in shared networks, the evaluation of any telehaptic protocol is largely incomplete without analyzing its interplay with TCP cross-traffic. We seek to fill this gap by providing a comprehensive assessment of the interplay between telehaptic protocols and heterogeneous cross-traffic, consisting of CBR as well as TCP flows. This leads to the formulation of a set of general conditions for

QoS compliance of telehaptic flows.

Specifically, we focus on the following two classes of telehaptic protocols.

CBR-based telehaptic protocols: This class of protocols generates a constant bit rate (CBR) data stream, i.e., they inject traffic into the network at a steady rate. Examples of such protocols include the designs proposed in [9, 29, 33].

Adaptive sampling based telehaptic protocols: This class of protocols employs adaptive sampling for haptic data rate adaptation [14, 47, 87]. Several papers propose telehaptic communication using adaptive sampling [24, 73, 87].

Interestingly, the two rate control protocols presented in this thesis (opportunistic adaptive sampling in Chapter 4 and DPM in Chapter 6) also belong to the above classes as discussed in Sections 7.4.4 and 7.5.3. Thus, the conclusions of this chapter apply to these protocols as well. Our contributions are the following.

- We perform an analytical characterization of the maximum and minimum queue occupancy at the ingress of the bottleneck link, when a CBR-based telehaptic protocol coexists with a TCP and CBR cross-traffic (see Section 7.3.1).

- Using the results from the queue occupancy characterization, we develop a set of conditions for QoS compliance for CBR-based telehaptic protocols (see Sections 7.3.2 and 7.3.3). While satisfying the delay constraint requires the network parameters to satisfy a certain condition, the jitter constraint is purely a function of cross-traffic source parameters. On the other hand, the packet loss depends heavily on the sizes of the telehaptic packets relative to TCP packets. We validate our claims through extensive simulations (see Section 7.4).

- We perform a simulation-driven investigation to demonstrate that the statistical compression provided by adaptive sampling is not useful from the standpoint of reducing the bandwidth requirement of telehaptic communication (see Section 7.5). Furthermore, we state the conditions for QoS compliance for adaptive sampling based telehaptic protocols as well.

The remainder of the chapter is organized as follows. In Section 7.2, we provide a brief overview of the working of TCP. In Section 7.3, we present our analytical model for capturing the dynamics of TCP-CBR interplay. We evaluate the performance of CBR-based telehaptic protocols in Section 7.4, and that of adaptive sampling based telehaptic protocols in Section 7.5. We state our conclusions in Section 7.6.

7.2 TCP Background

TCP forms the backbone of several internet applications that require reliable data transfer, such as web browsing, email, file transfer, and even video streaming applications like YouTube [3] and Netflix [1]. Several Internet measurement studies reveal that around 90% of the overall traffic belongs to TCP [85, 112]. TCP is a transport layer protocol that governs the rate at which the application injects traffic into the network based on the perceived network conditions. It achieves end-to-end reliability through the retransmission of lost packets, which are detected using packet acknowledgments (ACKs) that are sent to the source by the receiver. In this section, we give a brief overview of the working of TCP NewReno [32], which is the most widely deployed variant of TCP.

TCP rate adaptation is based on packet losses as detected by the source. In principle, the TCP source increases its data rate gradually until it detects a packet loss. As the instantaneous rate exceeds the network capacity, the network queues start building up. The overflowing of the network queues causes packet losses. The source responds to the loss by reducing the data rate in an aggressive manner. Once the congestion is controlled, the source resumes the rate increase, and the cycle continues. Thus, the inherent nature of TCP rate adaptation introduces large queueing delays and packet losses.

For every packet delivery, the receiver generates an acknowledgment that is transmitted back to the source, signaling successful packet reception. In case of a packet loss, the receiver transmits a duplicate ACK for every packet arrival subsequent to the loss. The source retransmits the lost packet on receiving three successive duplicate ACKs.

For the purpose of rate adaptation, the source maintains a variable named *congestion window* (denoted by W) that defines the number of TCP packets that are outstanding, i.e. transmitted but not yet acknowledged. Therefore, the source transmits one packet per ACK arrival in order to restore the amount of outstanding packets to W. If S_{tcp} denotes the size of a TCP packet in bytes, then WS_{tcp} denotes the amount of outstanding TCP bytes. W controls the rate at which traffic is injected into the network - a higher W corresponds to a higher transmission rate, and vice-versa.

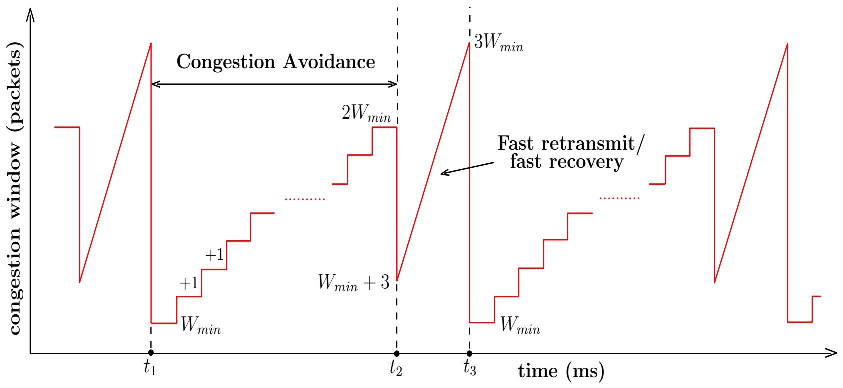


Figure 7.1: Depiction of TCP rate adaptation in terms of slot-wise variation of the congestion window.

The rate adaptation of TCP is carried out in multiple stages: slow start, congestion avoidance, and fast retransmit, fast recovery. The TCP source is in slow-start during the beginning of transmission. In steady state, the protocol operates between congestion avoidance and fast retransmit, fast recovery phases. In this section, we describe the steady state dynamics of TCP NewReno [32]. The steady state congestion window evolution is shown in Figure 7.1.

TCP transmission begins with the *slow-start*, during which W is increased by one for every ACK received. When W exceeds a predefined slow-start threshold, the source enters the *congestion avoidance* (t_1 to t_2 in Figure 7.1). In this phase, W is increased more prudently by 1 over W number of ACKs received. For example, let us assume that W is currently updated from 10 to 11. Since there are already 10 outstanding TCP packets, the source transmits another packet to satisfy the condition $W = 11$. For convenience, we name this packet as the *probing packet*, since it probes for the additional network bandwidth. Assuming no packet losses, the ACK corresponding to the probing packet reaches the source after 10 ACKs (corresponding to the packets that are ahead of the probing packet). Therefore, the ACK of the probing packet is the 11^{th} arrival, starting from the latest W-update. Upon reception of the ACK corresponding to the probing packet, W is further increased to 12 since the source has received 11 ACKs. In other words, W is updated after a round trip time (RTT) encountered by the probing packet. This leads to a *slot-wise* behavior of W [98], as shown in Figure 7.1. Essentially, a slot is the time duration between two consecutive W updates during the congestion avoidance phase. The length of a slot is equal to the RTT experienced by the probing packet transmitted at the start of that slot.

The increase in W (and consequently the rate) results in a build up of network queues,

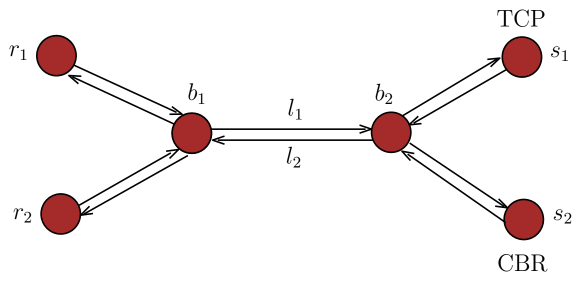


Figure 7.2: Single bottleneck dumbbell network topology design for the analysis of the queue dynamics in presence of a TCP source. Notations: l_1 and l_2 - intermediate links; s_1 and s_2 - traffic sources; r_1 and r_2 - traffic receivers; b_1 and b_2 - intermediate nodes.

eventually leading to packet losses. The receiver begins to transmit a duplicate ACK for every packet received subsequent to the loss. The source infers packet loss upon receiving three such duplicate ACKs, and retransmits the lost packet (at t_2 in Figure 7.1). This phase is known as *fast retransmit*. Let W_l denote the value of W when a packet loss is sensed. It should be noted that the source transmits no packets in response to the first and the second duplicate ACKs. Following the retransmission, W is reduced to $\frac{W_l}{2} + 3$ (3 to compensate for the three duplicate ACKs that indicate successful delivery of three packets subsequent to the lost packet). Thereafter, the source enters *fast recovery* phase (t_2 in Figure 7.1). Assuming a single packet loss, the number of outstanding packets at the start of the fast recovery phase is $W_l - 3$.

During fast recovery (t_2 to t_3 in Figure 7.1), the source executes the update $W + 1 \leftarrow W$ for every duplicate ACK arrival. The source transmits no packets until $W > W_l$. This means that it takes $\frac{W_l}{2} - 2$ ACK arrivals (from $W = \frac{W_l}{2} + 3$) for resuming the transmission. Thereafter, the transmission begins at the rate of 1 packet per duplicate ACK received, until the lost packet is acknowledged. Following this, W is updated to $\frac{W_l}{2}$, and the source re-enters congestion avoidance.

Let the minimum value of W in the steady state be denoted by W_{min}. Hence, $W_{min} = \frac{W_l}{2}$. Note that a total of W_{min} packets are pumped into the network, and $2W_{min}$ duplicate ACKs are received during the fast retransmit, fast recovery phase. Hence, $W = 3W_{min}$ is the maximum value of the congestion window. Note that during fast retransmit, fast recovery phase, W does not signify the number of outstanding TCP packets.

The work in [98] investigated the case of a TCP source in steady state on a single bottleneck dumbbell network topology, shown in Figure 7.2. The authors considered a network

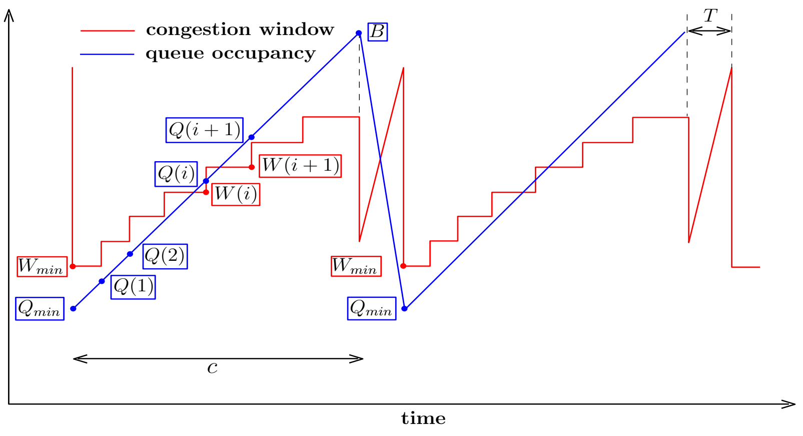


Figure 7.3: Demonstration of the periodic variation of congestion window and queue occupancy cycles in presence of a TCP source. It should be noted that the queue occupancy graph is shown to be linear only for the purpose of illustration. The analysis itself does not make any assumptions on the shape of the queue occupancy graph.

setting in which $B > 2\mu\tau$, where μ is the bottleneck link capacity (in kbps), 2τ is the round trip propagation delay of the source (in ms), and B is the size of the queue at the ingress of the bottleneck link (in bytes). This condition comes from the conventional wisdom, according to which the network is fully utilized under TCP traffic only when the queue size is larger than the bandwidth delay product (BDP) [103]. The authors in [98] showed that in such scenarios, the TCP congestion window profile exhibits a periodic behavior. Furthermore, they demonstrated that the bottleneck queue occupancy (denoted by Q in bytes) also evolves in synchronization with W, as shown in Figure 7.3. This means that the minimum queue occupancy ($Q = Q_{min}$) occurs when $W = W_{min}$. The duration between two consecutive congestion window or queue occupancy minima is called a *cycle*.

At the instant of packet loss, $Q = B$, since this condition causes the queue to drop packets. Since the quantity Q takes the maximum and the minimum values at the start and the end of the fast recovery phase, respectively, it is fairly simple to realize that the drain rate of the queue is the highest during this phase. At the start of a cycle, the instantaneous rate exceeds the channel capacity, and hence the queue starts to build up. The work in [98] showed analytically that the queue occupancy increases by S_{tcp} per slot, i.e. the source adds 1 packet per slot to the queue more than the previous slot. This is because W increases by 1 packet in every slot. Furthermore, the authors derived mathematical expressions for Q_{min} and W_{min} as follows.

$$Q_{min} = \frac{B - 2\mu\tau}{2} \tag{7.1}$$

$$W_{min} = \frac{B + 2\mu\tau}{2S_{tcp}}. \tag{7.2}$$

With this TCP background, we move to our analysis of the interplay between TCP and CBR flows existing concurrently on a shared network. In our analysis in Section 7.3, we take the model in [98] and refine it substantially to analyze the impact of heterogeneous traffic sources on a single bottleneck link.

7.3 TCP-CBR Interplay: Characterization

Various aspects of the TCP-CBR interplay have been investigated in the past. We discuss a few of the relevant works available in the literature. The work in [41] analyzes the impact of coexisting CBR streams on the throughput of TCP streams. This work also presents a novel mechanism to improve the TCP performance without affecting the UDP flow. The authors in [75] study the impact of TCP on real-time video streaming in terms of QoS for network stability and traffic fairness. The interaction between UDP, TCP and routing protocols has been studied in [84], with the focus primarily on the impact of TCP performance and network route stability. The study in [28] investigates the dependence of network utilization and fairness on the intermediate queue sizes. The works in [22, 114] have analyzed the ability of IEEE 802.11 protocol in supporting the QoS for voice and video. The works in [104–106] have studied the anomalous loss performance and TCP throughput in mixed real-time and TCP traffic with the variations in network queue sizes. The work in [88] has investigated the impact of UDP packet losses in presence of TCP sources. To summarize, all of the prior studies are centered around audio-video streaming applications. From the telehaptics point of view, such studies are insufficient for characterizing a holistic telehaptic QoS compliance.

In this section, we study the interplay between CBR and TCP flows through the development of an analytical model. The goal of this section is to develop a comprehensive understanding of the conditions that need to be satisfied for ensuring QoS-compliant telehaptic communication on a shared network for CBR-based telehaptic protocols.

We consider the single bottleneck network topology (shown earlier in Figure 7.2) with a single TCP source. For the purpose of analysis, multiple CBR cross-traffic sources can be thought of as a single CBR source with the rate R equal to the aggregate of the individual rates. Nodes s_1 and s_2 act as the TCP and CBR sources, respectively. Nodes r_1 and r_2 act as their respective receivers. Hence, l_2 becomes the bottleneck link, and the variation in the queue occupancy is observed at b_2.

7.3.1 Queue Occupancy

A. Assumptions: For the ease of our analysis, we make the following assumptions.

1. The TCP source has an infinite backlog of data.

2. The access links to l_1 and l_2 have very high bandwidth and negligible propagation delays. In effect, the traffic sources (respectively, receivers) directly feed into (respectively, read from) the intermediate nodes.

3. The queue size at the ingress of the bottleneck link (b_2) is greater than the BDP i.e. $B > 2\mu\tau$. This condition guarantees that the queue never empties, and hence the bottleneck link is never underutilized.

4. In every cycle, the TCP stream loses exactly one packet. Additionally, the packet losses occur solely due to queue overflow at b_2.

Under the above assumptions, when R is non-zero, it turns out that the instantaneous traffic injection rate at the start of congestion avoidance phase (t_1) is less than the channel capacity μ. As a result, the queue continues to drain for c_1 slots after t_1, as shown in Figure 7.4. Once the instantaneous rate exceeds μ, the queue begins to build until the packets start getting dropped. Hence, the queue continues to drain for c_1 slots (such that $0 < c_1 < c$) at the beginning of the congestion avoidance phase. Therefore, the minima of Q-plot (in Figure 7.4) lags by c_1 slots with respect to the minima of W-plot.

Let Q_{init} denote the queue occupancy at the start of the congestion avoidance phase (time t_1). Hence, $Q_{init} > Q_{min}$ for $R > 0$. Let $Q(i)$ denote Q at the start of the i^{th} slot, and let $W(i)$ denote W during i^{th} slot. Let T denote the duration of the fast retransmit, fast recovery phase.

B. Congestion avoidance: In this section, we analyze in detail the queue dynamics in the congestion avoidance phase (t_1 to t_2). Unlike in the single TCP source case (shown in Figure 7.3),

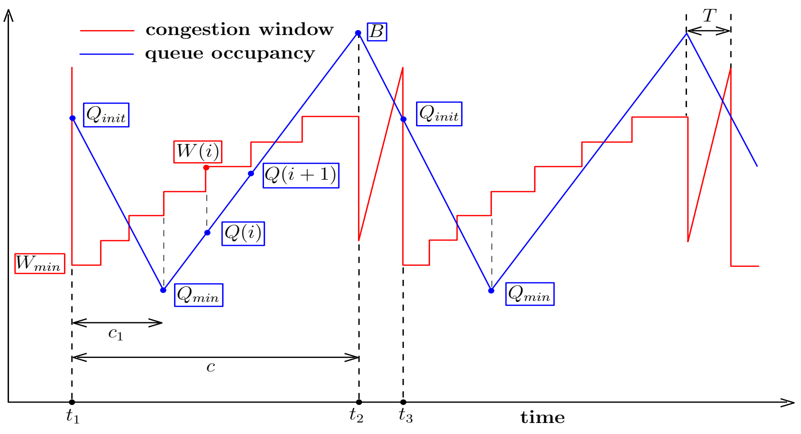


Figure 7.4: Demonstration of the periodic variation of congestion window and queue occupancy cycles in presence of coexisting TCP and CBR sources. It should be noted that the queue occupancy graph is assumed to be linear only for the purpose of illustration. The analysis itself is robust to the nature of the queue occupancy graph.

this phase can be divided into the following two regions based on the nature of the queue occupancy profile:

1. *increasing region*: the queue builds up from Q_{min} to B in $c - c_1$ slots,
2. *decreasing region*: the queue drains from Q_{init} to Q_{min} in c_1 slots.

We now analyze each of these regions in detail.

Increasing region: We begin by analyzing the relationship between queue occupancies at successive slot boundaries in the increasing region. At any point in time, $Q(i)$ is a combination of CBR and TCP bytes. Let $Q_C(i)$ and $Q_T(i)$ denote the CBR and TCP components of $Q(i)$, respectively. Let $D_C(i)$ and $D_T(i)$ denote the amount of CBR and TCP bytes, respectively, drained from the queue during the i^{th} slot. Let $RTT(i)$ denote the length of the i^{th} slot. Recall that $RTT(i)$ is the RTT encountered by the probing packet of the i^{th} slot. Therefore, we can write

$$RTT(i) = 2\tau + \frac{Q(i)}{\mu} \tag{7.3}$$

where 2τ is the round trip propagation delay, and the second term is the queueing delay faced by the probing packet of the i^{th} slot.

Recall that the TCP source injects $W(i)S_{tcp}$ bytes of data in the i^{th} slot. Since the TCP source adds 1 packet more to the queue in the current slot relative to the previous slot, we can write $Q_T(i+1) - Q_T(i) = S_{tcp}, \forall i \in [c_1+1, c-1]$. Relating the initial states of the queue, input and output during i^{th} and $i+1^{\text{th}}$ slots, we obtain the following equation for the TCP component of the queue.

$$[Q_T(i+1)+W(i+1)S_{tcp}-D_T(i+1)]-[Q_T(i)+W(i)S_{tcp}-D_T(i)] = S_{tcp}, \forall i \in [c_1+1, c-1] \tag{7.4}$$

We now move to the CBR component of the queue for which we derive an equation analogous to Equation (7.4). The amount of data injected by the CBR stream during the i^{th} slot is given by $RTT(i)R$. Let $\Delta Q_C(i)$ denote the amount of additional CBR data in the queue (in bytes) at the end of $i+1^{\text{th}}$ slot, relative to that at the end of i^{th} slot. Relating the initial queue states, input and output during i^{th} and $i+1^{\text{th}}$ slots, we obtain the following equation for the CBR component of the queue.

$$\begin{aligned}[Q_C(i+1) + RTT(i+1)R - D_C(i+1)] - [Q_C(i) + RTT(i)R - D_C(i)] \\ = \Delta Q_C(i), \forall i \in [c_1+1, c-1]\end{aligned} \tag{7.5}$$

We note that $Q_T(i) + Q_C(i) = Q(i)$, and the overall drain in the i^{th} slot $D_T(i) + D_C(i) = \mu RTT(i)$. Adding Equations (7.4) and (7.5), and using the above relationships we obtain

$$\begin{aligned}Q(i+1) - Q(i) + R[RTT(i+1) - RTT(i)] - \mu[RTT(i+1) - RTT(i)] \\ +[W(i+1) - W(i)]S_{tcp} = S_{tcp} + \Delta Q_C(i)\end{aligned} \tag{7.6}$$

Substituting the expression for $RTT(i)$ from Equation (7.3), and utilizing the relationships like $Q(i+1) - Q(i) = S_{tcp} + \Delta Q_C(i)$, and $W(i+1) - W(i) = 1$, Equation (7.6) yields

$$\Delta Q_C(i) = \frac{RS_{tcp}}{\mu - R}, \forall i \in [c_1+1, c-1] \tag{7.7}$$

Equation (7.7) suggests that in the increasing region, the CBR source accumulates data at a constant rate of $\frac{RS_{tcp}}{\mu - R}$ per slot in the queue.

We now look at the relationship between the end-to-end queue occupancies in the increasing region. We know that in this region there are $c - c_1 - 1$ jumps in the queue occupancy, with each jump of magnitude $\Delta Q_C(i) + S_{tcp}$. Thereby, relating Q_{min} and B we obtain

$$Q_{min} + (c - c_1 - 1)(\Delta Q_C(i) + S_{tcp}) = B, \forall i \in [c_1+1, c-1] \tag{7.8}$$

Substituting Equation (7.7) in (7.8), we obtain

$$Q_{min} + (c - c_1 - 1)\frac{\mu S_{tcp}}{\mu - R} = B, \forall i \in [c_1+1, c-1] \tag{7.9}$$

Decreasing region: We now move to the decreasing region i.e., $i \in [1, c_1]$, where we seek to capture the evolution of queue occupancies. Note that $Q(i)$ is a decreasing function of i in the decreasing region. Therefore, from Equation (7.3) we infer that RTT encountered by the probing packets in successive slots reduces progressively until c_1 slots. Using the basic input-output equation in each of the slots, we can write

$$Q(i+1) = Q(i) + RTT(i)R + W(i)S_{tcp} - \mu RTT(i), \forall 1 \leq i \leq c_1 \tag{7.10}$$

By definition, $Q(1) = Q_{init}$, $Q(c_1+1) = Q_{min}$ and $W(i) = W_{min} + i - 1$. Adding up the c_1 components of Equation (7.10), we obtain the relationship between the end-to-end queue occupancies in the decreasing regime as follows.

$$Q_{min} = Q_{init} + (R - \mu)\sum_{i=1}^{c_1} RTT(i) + \sum_{i=1}^{c_1}(W_{min} + i - 1)S_{tcp} \tag{7.11}$$

From Equations (7.3) and (7.11), we obtain

$$Q_{min} = Q_{init} \cdot \alpha^{c_1} + [(\alpha - 1)2\mu\tau + W_{min}S_{tcp}]\left[\frac{1-\alpha^{c_1}}{1-\alpha}\right] + S_{tcp}\sum_{j=0}^{c_1-2}(c_1 - 1 - j)\alpha^j \tag{7.12}$$

where $\alpha = R/\mu$.

C. Fast retransmit, fast recovery: We now move to modeling the queue dynamics during the fast retransmit, fast recovery phase.

End-to-end queue occupancy relationship: Let T denote the duration of the fast retransmit, fast recovery phase, as shown in Figure 7.4. Recall (from Section 7.2) that over the duration T, the TCP source transmits W_{min} packets and receives $2W_{min}$ duplicate ACKs. A total of $2W_{min}$ arrivals implies that a total of $2W_{min}$ TCP packets have made their way out of the bottleneck link over the duration T. Recall (from Section 7.2) that there are $2W_{min}$ outstanding packets while the packet loss is detected (at t_2 in Figure 7.4). This implies that the fast retransmit, fast recovery phase ends (at t_3) exactly when all of the TCP data present in the queue at t_2 has been acknowledged. We know, from Equation (7.7), that at t_2 the queue occupancy B is a mix of CBR and TCP data in the ratio $\frac{R}{\mu-R}$. Therefore, a total of $2W_{min}S_{tcp}$ bytes of TCP and $2W_{min}S_{tcp}\frac{R}{\mu-R}$ bytes of CBR have escaped the bottleneck link over the duration T. Therefore, we obtain the expression for T as

$$T = \frac{2W_{min}S_{tcp} + 2W_{min}\frac{R \cdot S_{tcp}}{\mu-R}}{\mu} = \frac{2W_{min}S_{tcp}}{\mu - R} \tag{7.13}$$

Relating the end-to-end queue occupancies during T, we obtain

$$Q_{init} = B + (R - \mu)T + W_{min}S_{tcp} \tag{7.14}$$

From Equations (7.13) and (7.14), we obtain

$$Q_{init} = B - W_{min}S_{tcp} \tag{7.15}$$

TCP component of queue at t_3: We now seek to compute the amount of TCP data stored in the queue at t_3. As discussed previously, the contents stored in queue at t_2 are flushed out of the network at t_3. This clearance of backlog implies that all of the data stored in the queue at t_3 must have been injected between t_2 and t_3. The data injected during this phase is either stored in the queue or is in transit in the channel. For the ease of analysis, let us assume that the ratio of the TCP data in the queue at t_3 and the TCP data injected in T is comparable to the corresponding ratio of the CBR stream. Let us denote this ratio by r. Hence, the ratio of CBR and TCP data in the queue at t_3 can be written as $\frac{RTr}{W_{min}S_{tcp}r}$. Using the expression for T from Equation (7.13), this ratio reduces to $\frac{2R}{\mu-R}$.

This finding is striking as the ratio of the CBR and the TCP components in the queue doubles during the interval T. Recall that this ratio is equal to $\frac{R}{\mu-R}$ at the beginning of the duration T. The justification for the increase in the ratio is fairly simple. After fast retransmission, the TCP source transmits nothing until the next W_{min} arrivals, whereas the CBR source continue to pump in data at the steady rate of R. Thus, the queue is predominantly occupied by the CBR contents at the end of fast retransmit, fast recovery phase.

Using this ratio, the queue component of TCP can be expressed as $Q_{init}(\frac{\mu-R}{\mu+R})$. It is reasonable to argue that the channel will be shared between the TCP and the CBR streams in the same proportion as the queue. Hence, the channel component of TCP can be approximated as $2\mu\tau(\frac{\mu-R}{\mu+R})$. Therefore we obtain an equation for the TCP component of Q_{init} as follows.

$$Q_{init}\left(\frac{\mu - R}{\mu + R}\right) = W_{min}S_{tcp} - 2\mu\tau\left(\frac{\mu - R}{\mu + R}\right) \tag{7.16}$$

D. Inter-cycle congestion window: The analysis so far was carried out to obtain inter-slot relationships in a cycle. We now move to the inter-cycle analysis. As discussed in Section 7.2, at the beginning of a cycle W is set to half its value at the end of congestion avoidance in the previous cycle. Recall that W is incremented $c-1$ times during congestion avoidance. Relating the congestion window values at the start of the congestion avoidance and the fast retransmit, fast recovery phases, we obtain

$$W_{min} = \frac{W_{min} + c - 1}{2}$$

which on simplification gives

$$W_{min} = c - 1 \tag{7.17}$$

Solving the simultaneous equations (7.15), (7.16) and (7.17), we obtain the closed form expressions for W_{min}, Q_{init} and c as follows.

$$W_{min} = \frac{(B + 2\mu\tau)(1 - \alpha)}{2S_{tcp}} \tag{7.18}$$

$$Q_{init} = \frac{(B + 2\mu\tau)(1 + \alpha)}{2} - 2\mu\tau \tag{7.19}$$

$$c = \frac{(B + 2\mu\tau)(1 - \alpha)}{2S_{tcp}} + 1 \tag{7.20}$$

Substituting Equations (7.18), (7.19) and (7.20) in (7.9) and (7.11), we obtain

$$Q_{min} + \left[\frac{(B + 2\mu\tau)(1 - \alpha)}{2S_{tcp}} - c_1\right]\left[\frac{S_{tcp}}{1 - \alpha}\right] = B \tag{7.21}$$

$$Q_{min} = \left[\frac{(B + 2\mu\tau)(1 + \alpha)}{2} - 2\mu\tau\right]\alpha^{c_1} + \left[\frac{(B - 2\mu\tau)(1 - \alpha^{c_1})}{2}\right] + S_{tcp}\sum_{j=0}^{c_1 - 2}(c_1 - 1 - j)\alpha^j \tag{7.22}$$

Obtaining closed form expressions for Q_{min} and c_1 from Equations (7.21) and (7.22) is hard. Hence, we resort to a numerical method of solving for the unknowns in finite steps, since $0 < c_1 < c$. Equation (7.22) can be converted to a more generic form as follows.

$$Q(i) = \left[\frac{(B + 2\mu\tau)(1 + \alpha)}{2} - 2\mu\tau\right]\alpha^{i-1} + \left[\frac{(B - 2\mu\tau)(1 - \alpha^{i-1})}{2}\right] + S_{tcp}\sum_{j=0}^{i-3}(i - 2 - j)\alpha^j, \forall i \in [1, c - 1].$$

Since we know that $Q(i)$ is a unimodal function over a cycle, Q_{min} and c_1 can be calculated as

$$Q_{min} = \min_i Q(i), \quad c_1 = \arg\min_i Q(i), \qquad \forall i \in [1, c - 1] \tag{7.23}$$

To summarize, through analysis we obtained the following expressions.

$$W_{min} = \frac{(B + 2\mu\tau)(1 - \alpha)}{2S_{tcp}}$$

$$Q_{init} = \frac{(B + 2\mu\tau)(1 + \alpha)}{2} - 2\mu\tau$$

$$c = \frac{(B + 2\mu\tau)(1 - \alpha)}{2S_{tcp}} + 1$$

We use these to compute Q_{min} and c_1 using a finite step numerical method through Equation (7.23). We now use this estimate for characterizing the packet delay, as explained next.

7.3.2 Packet Delay

Equation (7.23) gives the numerical estimate of the minimum queue occupancy for a given network setting. The minimum end-to-end delay d_{min} is experienced when the queue occupancy is at its minimum, and is given by

$$d_{min} = \frac{Q_{min}}{\mu} + \tau. \tag{7.24}$$

The maximum end-to-end delay d_{max} is experienced when the queue occupancy is close to its maximum i.e. B. Thus, the end-to-end is at most

$$d_{max} = \frac{B}{\mu} + \tau. \tag{7.25}$$

Therefore, in presence of a coexisting TCP source the end-to-end packet delays vary cyclically over the range $[d_{min}, d_{max}]$. When the CBR source is a telehaptic source, these bounds represent the bounds on the haptic delay. Further, these bounds allow us to check if QoS-compliant telehaptic communication is feasible in a given network configuration. It is important to note that the haptic sample delays depend on the type of multiplexing being employed. For simplicity, we consider the model for a peak rate transmission technique which adds negligible delay to the haptic samples. In this case, the minimum and the maximum delay encountered by the haptic samples is given by d_{min} and d_{max}, respectively. We validate the accuracy of our analysis through simulation results presented in Section 7.4.

Hence, the expressions for d_{min} (Equation (7.24)) and d_{max} (Equation (7.25)) give the bounds on haptic delay under the peak rate telehaptic transmission.

7.3.3 Packet Jitter

We now move to jitter characterization of the haptic samples in a heterogeneous cross-traffic setting. We focus on the peak jitter, denoted by ν_{max}, that the haptic samples encounter. It is fairly simple to visualize that the peak jitter is encountered by the haptic sample that sees a maximum jump in the buffer occupancy, denoted by ΔQ. In other words, the peak jitter occurs when the number of cross-traffic packets (including CBR and TCP) injected into the network between any two adjacent haptic packets is the maximum. Let R_{cbr} denote the data rate of the CBR cross-traffic source, and let S_{cbr} denote the corresponding packet size. Let T_h denote the inter-packet time of the telehaptic stream. Let m_{tcp} and m_{cbr} denote the maximum number of packets transmitted by the TCP and the CBR cross-traffic sources, respectively, in the interval T_h. Therefore, ΔQ can be expressed as

$$\Delta Q = m_{tcp}S_{tcp} + m_{cbr}S_{cbr} - \mu T_h \tag{7.26}$$

Here, the first and the second terms represent the TCP and the CBR components of the maximum jump, respectively, and the third term is the amount of buffer drain in the interval T_h. Therefore, ν_{max} can be expressed in terms of ΔQ as

$$\nu_{max} = \frac{\Delta Q}{\mu} \tag{7.27}$$

We now characterize each of the CBR and the TCP components in detail.

A. TCP Component: At this stage of analysis, it is appropriate to describe the *cumulative acknowledgement* principle of TCP NewReno, as the peak haptic jitter is largely governed by this process. All practical implementations of TCP NewReno adopt this principle [32]. According to this, the receiver generates an ACK every n^{th} data packet, where $n > 1$. Therefore, on arrival of an ACK, the source transmits a burst of $n + 1$ packets if the ACK corresponds to a probing packet. Otherwise, an $n-$packet burst is transmitted for an ACK corresponding to a non-probing packet. Recall that when an ACK corresponding to a probing packet arrives, the congestion window is updated, signifying the start of a slot. Therefore, the TCP component of the buffer occupancy jump is maximum at the start of a slot. Note that m_{tcp} could also include additional $n-$packet bursts apart from the $n + 1-$packet burst depending on the length of T_h.

As a running example, we demonstrate the working of cumulative acknowledgement with $n = 2$ in Figure 7.5. As seen from the figure, the receiver generates an ACK every alternate packet reception. It is important to note that the packets transmitted in a burst need not necessarily get ACKed simultaneously. For example, consider the burst containing the packets 1 and 2. Since the previous packet (call it 0) was not ACKed, the reception of packet 1 triggers an ACK. Consequently, packets 2 and 3 get ACKed concurrently. Note that the ACK corresponding to the probing packet (5 in this case) arrives in a shorter time after the previous ACK relative to that corresponding to non-probing packets ($\frac{2S_{tcp}}{\mu}$ for probing packet, versus $\frac{2S_{tcp}}{\mu - R}$ for non-probing packet). We elaborate on this in the next paragraph. This forms the key observation from the standpoint of jitter, as this scenario gives rise to peak haptic jitter. Note that all

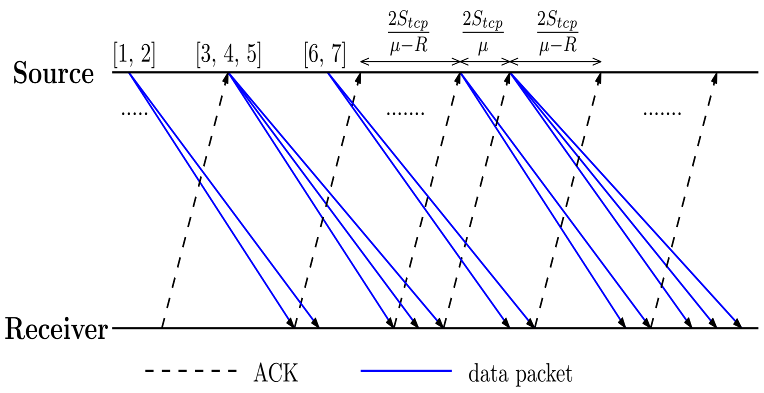


Figure 7.5: Demonstration of the working of TCP cumulative ACK with $n = 2$, along with the specification of inter-ACK gap.

probing packets need not necessarily give rise to peak jitter. However, it can be shown that the peak jitter case occurs once every n window updates.

As pointed out in Section 7.3.1, during congestion avoidance, the queue and hence the channel are shared by the CBR and TCP streams in the ratio $\frac{R}{\mu-R}$ (see Equation (7.7)). In other words, the TCP stream gets served at the rate of $\mu - R$. Let us now consider two adjacent ACKs, both corresponding to non-probing packets. Based on te TCP service rate, the time spacing between them can be given as $\frac{nS_{tcp}}{\mu-R}$ since there are n packet receptions between the two. Note that each of these ACKs trigger transmission of an $n-$packet burst.

The transmission of probing packet gives rise to a slightly different situation from that of a non-probing packet. Recall that the probing packet is always transmitted as a part of $n+1-$packet burst. When the first of these $n+1$ packets arrives at the receiver, two cases can occur depending on the number of packets waiting to be ACKed, denoted by p.

1. $p < n - 1$: In this case, the reception of the earliest $n - p$ packets trigger an ACK. The latter $p+1$ packets wait for the subsequent $n-p-1$ packets for triggering the next ACK. In this case, the inter-ACK time is $\frac{nS_{tcp}}{\mu-R}$.

2. $p = n - 1$: In this case, the reception of the first packet in the burst triggers an ACK. The latter n packets trigger another ACK. It is worth remarking that since the packet transmission is bursty in nature, it is reasonable to assume that the TCP packets belonging to a burst occupy contiguous locations in the queue. Hence, the TCP packets consume the full channel capacity (μ) until the transmission of the burst is complete. Therefore, the inter-ACK time in this case can be given as $\frac{nS_{tcp}}{\mu}$.

It can be shown that Case (2) is guaranteed to occur once every n^{th} window update during congestion avoidance. To summarize, the probing packets reduce the corresponding inter-ACK time, and thereby result in packet transmissions in quicker successions relative to that of non-probing packets. Therefore, the reception of ACKs corresponding to probing packets give rise to maximum number of TCP packets in between two successive telehaptic packets. We now focus on Case (2) for obtaining an expression for m_{tcp}.

We revisit Figure 7.5 for the current derivation. If $T_h < \frac{nS_{tcp}}{\mu}$, then $m_{tcp} = n+1$ as no more than one burst can be transmitted in the interval T_h. On the other hand, if $T_h > \frac{nS_{tcp}}{\mu}$ the number of $n-$packet bursts transmitted alongside the $n+1-$packet burst in the interval T_h depends on the length of T_h, and can be expressed as $1 + \left\lfloor \frac{T_h - \frac{nS_{tcp}}{\mu}}{\frac{nS_{tcp}}{\mu - R}} \right\rfloor$. Hence, m_{tcp} is given as

$$m_{tcp} = n + 1 + \left(1 + \left\lfloor \frac{T_h - \frac{nS_{tcp}}{\mu}}{\frac{nS_{tcp}}{\mu - R}} \right\rfloor\right) nI_{\left(T_h > \frac{nS_{tcp}}{\mu}\right)}, \tag{7.28}$$

where $I_z = 1$ if $z = 1$, and 0 otherwise.

B. CBR Cross-Traffic Component: It is fairly simple to realize that m_{cbr} is dependent on R_{cbr}, T_h and S_{cbr}, and can be given as as

$$m_{cbr} = \left\lceil \frac{R_{cbr} T_h}{S_{cbr}} \right\rceil + 1 \tag{7.29}$$

where the first term represents the maximum number of CBR cross-traffic packets in the duration T_h, and the second term (+1) accounts for the packet that could get transmitted close to the T_h boundary, and therefore influence the peak haptic jitter.

To summarize, we obtained the expressions for m_{tcp} and m_{cbr} as given by Equations (7.28) and (7.29), respectively. Using Equations (7.26), (7.27), (7.28) and (7.29), the expression for ν_{max} is given as

$$\nu_{max} = \frac{\left[n + 1 + \left(1 + \left\lfloor \frac{T_h - \frac{nS_{tcp}}{\mu}}{\frac{nS_{tcp}}{\mu - R}} \right\rfloor\right) nI_{\left(T_h > \frac{nS_{tcp}}{\mu}\right)}\right] S_{tcp} + \left[\left\lceil \frac{R_{cbr} T_h}{S_{cbr}} \right\rceil + 1\right] S_{cbr}}{\mu} - T_h \tag{7.30}$$

We validate the correctness of Equation (7.30) through simulations in Section 7.4.2.

Therefore, this characterization allows us to express the peak jitter faced by the haptic samples in a heterogeneous cross-traffic condition in terms of a set of source parameters ($n, T_h, S_{tcp}, R, R_{cbr},$ and S_{cbr}) and a network parameter (μ). Interestingly, the peak jitter has no dependence on other network parameters like B and τ, which govern the delay expressions. Therefore, we can say that the delay bounds are *network-driven* parameters, whereas the peak jitter is *source-driven*.

7.3.4 Packet Loss

As far as the packet loss is concerned, it is hard to characterize it using the fluid approximation of the traffic flows as packet losses are discrete-time events. Hence, we resort to a simulation-based characterization of the packet loss (see Section 7.4.3).

7.4 Performance Evaluation of CBR-Based Telehaptic Protocols

In this section, we consider the telehaptic protocols that exhibit CBR characteristics in their transmission rates, and investigate the telehaptic QoS compliance under heterogeneous cross-traffic conditions on a shared network. Since our analysis of Section 7.3 applies to any CBR flow in general, we leverage it for checking the feasibility of QoS for telehaptic protocols that manifest CBR characteristics in their transmission; see, for example, [9, 29, 33]. It is worth remarking that the precise haptic delay and jitter depend on the packetization and the multiplexing frameworks being used. For simplicity, we consider a telehaptic protocol that transmits at a peak rate (1000 packets per second), and investigate the impact of TCP on such telehaptic flows. In such a case, the expressions for d_{min} and d_{max} give the minimum and maximum haptic delays. Similarly, the worst case jitter expression of Section 7.3.3 gives the maximum haptic jitter. Note that the CBR traffic is now made up of a telehaptic component and a cross-traffic component.

For our experiments, we use NS3 - a discrete event network simulator [74]. Unless otherwise specified, we use the following network settings throughout this section. We set $\mu = 6$ Mbps, $\tau = 8$ ms and $B = 14$ kB. The chosen settings represent a medium speed internet link of length approximately equal to 1000 miles. We work with real world haptic traces generated by the Geomagic Phantom Omni device [4], which offers a single point of interaction between the human user and the haptic environment. Considering the standard haptic sampling rate of 1 kHz, and accounting for the overhead due to packet headers, we get a forward channel data rate $R_f = 688$ kbps, with packets of size 86 bytes transmitted every millisecond (see Chapter 3). On the backward channel, we simulate audio and video payload at the rate of 64 kbps and 400 kbps, respectively. We consider the media multiplexing mechanism proposed in (see Chapter 5), where each packet contains a single haptic sample and an audio/video fragment of fixed size. Including packet headers, this leads to a backward channel data rate of $R_b = 1.096$ Mbps, with packets of size 137 bytes transmitted every millisecond. For brevity, we report the simulation results for the case in which the cross-traffic sources are added to the backward channel only. In the notation of Section 7.3, note that the aggregate CBR rate on the backward channel $R = R_b + R_{cbr}$.

We introduce a TCP NewReno source on the backward channel, with the standard packet size $S_{tcp} = 578$ bytes. The CBR cross-traffic stream is composed of packets of uniform size $S_{cbr} = 150$ bytes. This is the typical packet size of a video-conferencing application such as Skype [2]. We use R_{cbr} as a control parameter by varying the inter-packet gap of CBR cross-traffic stream. For sustaining the TCP flow throughout the duration of the experiment, we need to ensure that $R < \mu$ so that the TCP flow has sufficient network bandwidth to perform rate adaptation. Our simulations are performed for a duration of 500 seconds. All measurements reported throughout this chapter are recorded after 5 seconds to allow the TCP source to enter the steady state.

7.4.1 Telehaptic Delay

We begin by investigating the telehaptic delays. Through simulations, we note that the TCP does not remain in steady state for $R > 5.5$ Mbps. Hence, we restrict our measurements to a maximum CBR data rate of $R = 5.5$ Mbps. In Table 7.1, we report the analytical bounds (derived in Section 7.3.2) as well as the measured minimum and maximum haptic delay by varying R_{cbr} to get R in the range $[R_b,\ 5.5$ Mbps$]$. We note that while the analytical lower bound d_{min} has a modest accuracy until $R_{cbr} = 4$ Mbps, the upper bound d_{max} is highly accurate. In Figure 7.6, we plot the temporal variation of the haptic delay with time for $R = 3$ Mbps (i.e., $R_{cbr} = 1.904$ Mbps), along with the analytical bounds. As expected, the haptic delay evolves periodically in time.

Based on our observations, we make the following remarks.

- The upper bound d_{max}, which is insensitive to R, remains highly accurate. However, the

R (Mbps)	d_{min} (ms)		d_{max} (ms)	
	A	S	A	S
1.096	9.91	8.89	26.66	26.47
2	10.74	9.21	26.66	26.40
3	12.27	11.71	26.66	26.62
4	14.81	12.95	26.66	26.45
5	19.87	16.95	26.66	26.55
5.5	22.68	19.77	26.66	26.43

Table 7.1: Comparison of d_{min} and d_{max} by analysis (A) and simulation (S) for a wide range of R.

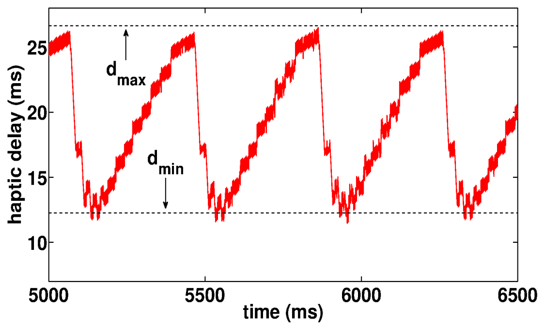


Figure 7.6: Plot of haptic delay showing the theoretical bounds for $R = 3$ Mbps.

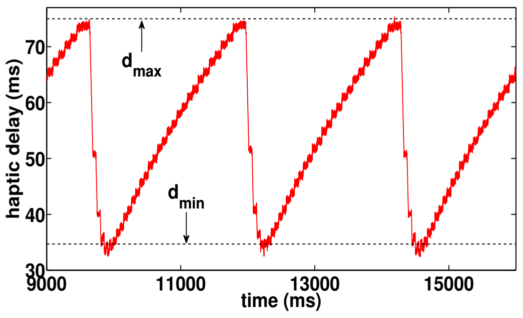


Figure 7.7: Plot of haptic delay demonstrating the severe QoS violation.

lower bound d_{min} becomes inaccurate for larger values of R. This is because our characterization of d_{min} assumes a single TCP packet loss in each cycle. However, we observe in our traces that for large values of R, TCP suffers multiple losses per cycle, leading to a very different congestion window evolution from the one considered for analysis. Simulating a wide range of network settings, we observe that a sufficient condition for a single TCP packet loss per cycle (and consequently for the accuracy of d_{min}) is $R \leq 0.65\mu$.

- Since the analytical upper bound d_{max} is highly accurate, it can be used to check for compliance of the haptic delay constraint for a given network setting. In the network setting under consideration, $d_{max} = 26.66$ ms which is less than the QoS limit of 30 ms. Indeed, our measurements confirm that the haptic delay constraint is satisfied in this case.

To see another realistic example, consider the setting $\mu = 6$ Mbps, $\tau = 15$ ms, and $B = 45$

kB. In this case, the analytical $d_{max} = 75$ ms which suggests that the haptic delay is way higher than the QoS limit, and hence the QoS constraint cannot be met. The simulations demonstrate that this is indeed the case as shown in Figure 7.7.

This motivates us to derive sufficiency conditions for telehaptic delay QoS compliance.

Sufficiency condition for delay QoS-compliance: In our evaluation, we observed that maximum haptic delay depends heavily on the network configuration. In some cases (for example, Figure 7.7), the haptic delay can get severely violated because of the large queue sizes. In this section, we will derive a sufficient condition that guarantee QoS compliance for haptic/audio/video delay in a single bottleneck network scenario.

- **Haptic delay:** Since the maximum haptic delay depends on the network parameters viz. μ, τ and B, they can be appropriately tuned to ensure that $d_{max} < 30$ ms, i.e.,

$$\frac{B}{\mu} + \tau < 30. \tag{7.31}$$

  As an example, for a given link (fixed μ and τ) the network administrator can configure the queue size B such that the above condition is satisfied. Consider the network setting corresponding to Figure 7.7. Using Equation (7.31), we see that configuring the queue size to $B = 11.25$ kB results in a maximum delay that conforms to 30 ms deadline (see Figure 7.8). Similarly one can determine conditions on the link dimensions that guarantee strict haptic delay QoS adherence. It is to be noted that $d_{min} = \tau$ for this setting. This is because in this case $B < 2\mu\tau$, resulting in network underutilization. This guarantees that the queue empties occasionally. However, network studies [11, 110] have shown that the coexistence of multiple TCP sources on a real-world network guarantees full network utilization even under the condition $B < 2\mu\tau$.

- **Audio/Video delay:** As in case of the haptic delays, the precise delays encountered by the audio/video frames depend on the type of multiplexing scheme being employed. To get a sense of the maximum audio/video delay let us consider the QoS-aware multiplexing scheme that we designed in Chapter 5. However, an analogous analysis can also be performed for other multiplexing mechanisms as well. For the benefit of the reader, we summarize the working principle of the QoS-aware multiplexing mechanism here. An audio/video fragment of fixed size s_m (in bytes) is transmitted along with each haptic sample, with audio payload having strict priority over video. Let s_a and s_v (both in

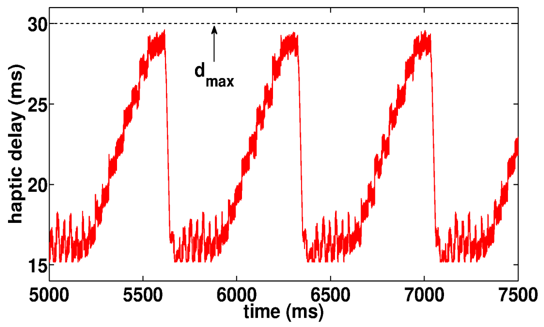


Figure 7.8: Plot of haptic delay showing the theoretical bounds for $R = 3$ Mbps. It can be observed that appropriately configuring the queue size results in a haptic delay QoS-compliant communication.

bytes) denote the size of each audio and video frame, respectively. Let f_a and f_v (in Hz) denote the frame rates of the audio and the video signals, respectively. For simplicity, we consider the case where f_a is an integral multiple of f_v. Assuming that the haptic delay deadline of 30 ms is met, it follows from the analysis carried out in Chapter 6 that maximum delay experienced by the audio and the video frames are as follows.

$$d_{aud} = 30ms + \frac{s_a}{s_m}(1ms), \qquad d_{vid} = 30ms + \frac{1}{f_v}, \tag{7.32}$$

For our default network setting, it can be shown that $s_a = 160$ bytes, $s_m = 58$ bytes, and $f_v = 25$ Hz. Hence, $d_{aud} = 32.75$ ms and $d_{vid} = 70$ ms. We see that even the worst case audio and video delays are comfortably within their respective QoS limits. Thus, we conclude that the compliance of the haptic delay constraint (which follows from Equation (7.31)) implies compliance of the delay constraint for audio and video as well.

To summarize, Equation (7.31) provides the sufficient condition for the design of networks which provide delay QoS-compliant telehaptic communication.

7.4.2 Haptic Jitter

We now turn to peak haptic jitter measurements. Figure 7.9 shows the peak haptic jitter curves, both by analysis and simulation, as a function of R. It can be seen that the analytical estimates of peak haptic jitter corroborate well with the simulation values, thereby validating our model. Moreover, the peak jitter remains comfortably within the QoS limit of 10 ms for the current setting. It is to be noted that the indicator variable in Equation (7.30) $I_{(T_h > \frac{nS_{tcp}}{\mu})}$ takes the value 0 for the chosen setting. In order to test the robustness of our model, we choose another setting that results in $I_{(T_h > \frac{nS_{tcp}}{\mu})} = 1$. Specifically, we set $R_{cbr} = 6.9$ Mbps, so that $R = 8$Mbps. We use μ as the control parameter and vary it in the range [10, 25] Mbps. In Table 7.2, we report the peak haptic jitter by analysis (J_A) and simulation (J_S). Throughout the range of μ, the analytical and simulation values corroborate well, further substantiating the validity of our model.

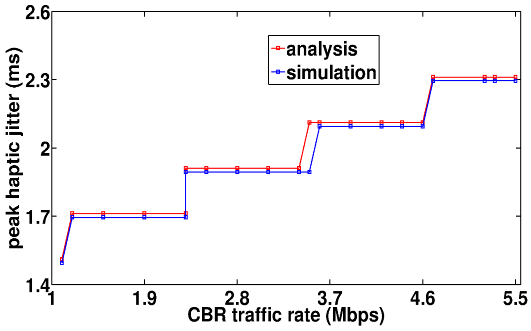


Figure 7.9: Graph showing the comparison between analytical approximation and measured haptic jitter on the backward channel. The accurate match between the two values validate the correctness of our model.

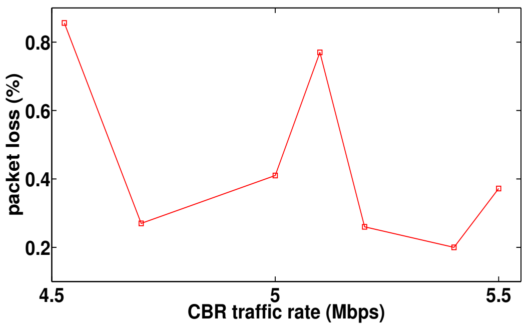


Figure 7.10: Packet loss on backward channel for a higher dimensional haptic device demonstrating that large sized packets are more vulnerable to packet drops. The measured packet losses are close to the QoS limit.

As another example, consider the network setting corresponding to Figure 7.9. For $n = 5$ and $S_{tcp} = 1.5$ kB, the peak haptic jitter turns out to be 12 ms, which clearly violates the QoS requirement of 10 ms. Hence, it is crucial to ensure that the source parameters are configured in a way that guarantees $\nu_{max} < 10$ ms.

7.4.3 Packet Loss

We now turn our focus on the packet losses suffered by the telehaptic stream. Since the fluid approximation cannot capture packet losses, we make a simulation-driven observation on the impact of heterogeneous traffic on the telehaptic packet losses. Interestingly, we notice that telehaptic packet losses are *zero* in spite of the regular queue overflows induced by the TCP flow. The reason for this is that the telehaptic stream uses smaller packets compared to TCP (137 bytes per packet on the backward channel for the telehaptic stream, versus 578 bytes per

μ (Mbps)	J_A (ms)	J_S (ms)
10	2.15	2.14
13	1.42	1.42
16	0.97	0.96
19	0.66	0.65
22	0.43	0.43
25	0.63	0.63

Table 7.2: Analytical and simulation results for the peak haptic jitter over a wide range of μ demonstrating the significant accuracy of our analytical model.

packet for the TCP stream). As a result, even when the queue drops a TCP packet, the adjacent telehaptic packets can still (potentially) be accommodated.

To confirm our conjecture that smaller telehaptic packet sizes are responsible for the absence of telehaptic packet losses, we simulate a scenario with higher resolution haptic, audio, and video devices, so that the telehaptic packet size becomes comparable to the TCP packet size. Specifically, consider a haptic device like Cybergrasp [8] or Festo exohand [6]. Assuming two interaction points for each of the ten fingers of the hand results in a twenty-fold increase in the haptic payload rate. Additionally, we simulate a high-definition audio and video payload with rates of 128 kbps and 2 Mbps, respectively. This results in R_b = 4.528 Mbps, and a packet size of 566 bytes on the backward channel for every millisecond. Figure 7.10 presents the packet loss (in %) encountered by this telehaptic stream, where we vary R_{cbr} to get R in the range [R_b, 5.5 Mbps]. Note that with the larger telehaptic packets, losses do occur. While the measured telehaptic losses pose no threat to haptic media which has a QoS limit of 10%, the audio and video, which have a more stringent limit of 1%, are susceptible to severe QoS violations.

To summarize, if telehaptic packets are small relative to TCP packets, the telehaptic stream sees little or no packet loss. However, if the telehaptic packets become comparable in size to the TCP packets (due to higher fidelity media devices), packet losses become noticeable. This could potentially result in catastrophic effects on the telehaptic activity.

In conclusion, we see that for QoS-compliant communication under CBR-based telehaptic protocols, the following conditions need to be satisfied.

- For buffer stability, we naturally require that the aggregate CBR data rate is less than the link capacity, i.e., $R < \mu$.

- In order to satisfy the haptic delay constraint, we need $d_{max} < 30$ ms, i.e.,

$$\tau + \frac{B}{\mu} < 30ms. \tag{7.33}$$

- In order to satisfy haptic jitter constraint, we need $\nu_{max} < 10$ ms.

- To avoid loss in the presence of concurrent TCP traffic, the packet sizes used by the telehaptic protocol should be less than the TCP packet sizes.

7.4.4 DPM Under Heterogeneous Cross-Traffic

We now consider the lossless rate adaptive protocol DPM that we designed in Chapter 6, and comment on its interplay with heterogeneous cross-traffic. Recall that DPM bunches $k \geq 1$ haptic samples in a packet depending on the nature of piggybacked end-to-end (E2E) delays. The packetization parameter k is used to control the data rate of DPM, and it relies heavily on the observed trend of the E2E delay. k is increased when the delays exhibit an increasing pattern, and is reduced when the delays exhibit a steady trend.

We saw in Section 7.2 that under TCP, the queueing delay experiences continuous variations, without any scope for steady behavior. The congestion avoidance phase (which induces delay build-up) pushes the telehaptic source to back off to the minimal data rate corresponding to $k = k_{max}$. Since the telehaptic source never sees a steady delay profile hereafter, it never gets an opportunity to increase the rate. In other words, the presence of TCP flows reduces the DPM-based telehaptic stream to a constant bitrate (CBR) stream, with the minimum data rate corresponding to k_{max}. We validate our above argument with simulations, and the result is shown in Figure 7.11. It can be observed that when W is increased during the slow-start phase (0 to 800 ms), k is increased to 4, and then it stays there throughout. Therefore, the analysis of interplay between the CBR-based telehaptic protocols and the TCP applies to DPM-based telehaptic streams as well. However, it should be noted that DPM introduces additional packetization delay due to its merging technique, as mentioned earlier. Hence, the expressions for d_{min} and d_{max} change slightly to accommodate the packetization delay. Similarly, our analysis can be used for haptic jitter approximation due to DPM.

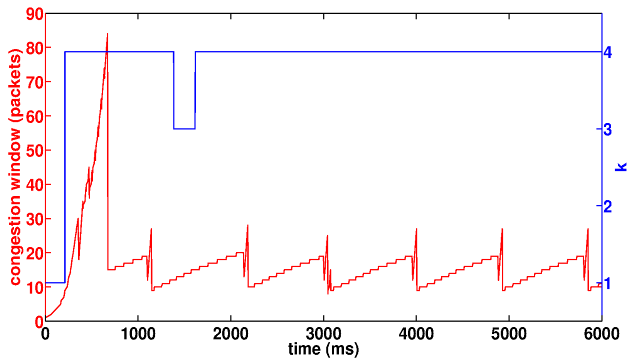


Figure 7.11: Demonstration of DPM's back off phenomenon during the TCP steady state, thereby behaving like a CBR source.

7.5 Performance Evaluation of Adaptive Sampling Based Telehaptic Protocols

In this section, we study the interplay between adaptive sampling based telehaptic protocols and heterogeneous cross-traffic flows. Recall that an adaptive sampling scheme based protocol transmits only perceptually significant haptic samples on the forward and/or backward channels. Like before, the goal of this section is to understand the conditions for QoS-compliant telehaptic communication.

If the protocol employs Weber sampling [47] on the backward channel, it must also specify how the perceptually significant haptic samples are multiplexed with audio/video data. For a working example, we consider the visual-haptic multiplexing protocol [24], which assumes haptic and video data on the backward channel. It multiplexes these two media streams as follows: Every perceptually significant haptic sample is packed with video data to generate a packet of worth 1 ms. On the other hand, when a series of haptic samples are perceptually insignificant, the protocol packs a large chunk of a video frame, not exceeding data of worth 15 ms, into a single packet.

In order to evaluate the protocol with realistic data, we record ten pilot signals collected from Phantom Omni device during the real-time telepottery activity. For brevity, we report the results only for one of these traces, but note that our findings apply to the remaining traces as well. The video payload rate is set to 400 kbps, as before. We use the network settings described previously in Section 7.4. In Figure 7.12, we plot the instantaneous telehaptic transmission rate on the backward channel due to this protocol. It can be seen that the instantaneous rate exhibits large fluctuations within the range [613, 1079] kbps, while the long term average rate of 712 kbps is substantially lower compared to the peak instantaneous rate (1079 kbps). We now turn to the interplay between this telehaptic flow and network cross-traffic. We begin by considering the impact of CBR cross-traffic alone, and then move to the heterogeneous cross-traffic case.

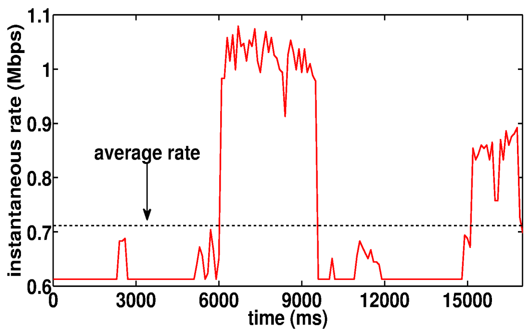


Figure 7.12: Instantaneous telehaptic rate exhibiting rapid fluctuations under visual-haptic multiplexing.

7.5.1 CBR Cross-Traffic

In this section, we consider the interplay between the visual-haptic multiplexing protocol and CBR cross-traffic on the backward channel. Our goal is to demonstrate that from the standpoint of QoS compliance on a shared network, the statistical compression provided by adaptive sampling is not meaningful. In other words, the network has to be able to support the peak transmission rate of the telehaptic flow for QoS compliance.

To see this, we consider an example where the network is provisioned for the average telehaptic rate. This means that the amount of network bandwidth available for the telehaptic stream throughout the application duration is atleast its average rate. To simulate this scenario, we set $R_{cbr} = 5.28$ Mbps so that the bandwidth available to the telehaptic stream is 720 kbps (note that this is higher than the average rate of 712 kbps). We remove the TCP source for this experiment. In Figure 7.12, consider the interval between 6000 ms and 10000 ms, when the instantaneous rate exceeds the available bandwidth. In this interval, our simulation traces reveal a significant haptic and video payload losses of around 6.2%. Even though the haptic loss is below the QoS limits (10%), the video loss is alarmingly high, causing severe violations of the QoS requirement (1%). As another example, setting μ = 3 Mbps and R_{cbr} = 2.28 Mbps results in larger haptic and video payload losses of around 9.6%.

7.5.2 Heterogeneous Cross-Traffic

For the case of heterogeneous cross-traffic, we reinstate the TCP source on the backward channel. Since $R_b \in [613, 1079]$ kbps, we vary R_{cbr} in the range [0, 4.4] Mbps so that $R \in [R_b$, 5.5 Mbps]. Figure 7.13 shows the haptic delay due to the visual-haptic multiplexing scheme. It can be seen that the buffering involved in the multiplexing scheme of this protocol gives rise to a large haptic delay, far higher than the deadline, causing severe QoS violations. An equation analogous to Equation (7.31) can be derived using our analysis for meeting the delay constraint under visual-haptic multiplexing scheme.

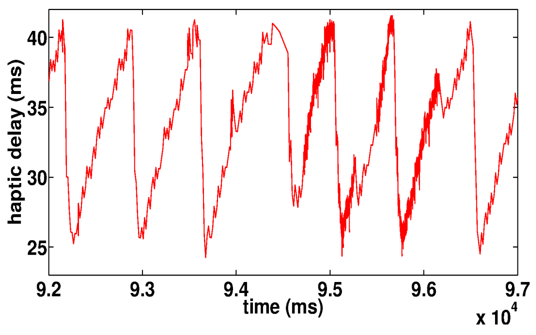


Figure 7.13: Demonstration of the haptic delay QoS violations by visual-haptic multiplexing scheme in presence of heterogeneous cross-traffic sources.

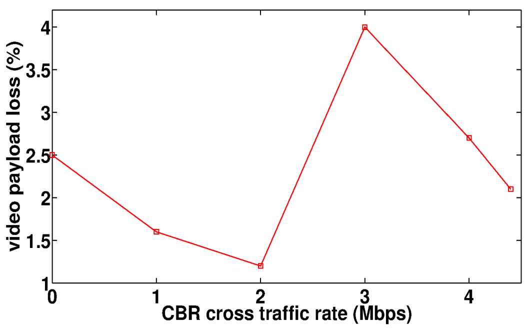


Figure 7.14: Demonstration of QoS violations of video payload loss in presence of heterogeneous cross-traffic under visual-haptic multiplexing.

Figure 7.14 shows the video payload loss (in %) recorded for various values of R. It can be seen that the loss QoS for video faces severe violations irrespective of the R value. This is because the visual-haptic multiplexing protocol transmits large packets with video payload in the absence of perceptually significant haptic samples. Recall, from our discussion in Section 7.4.3, that large packets are more likely to get dropped in the presence of TCP cross- traffic. Interestingly, haptic media suffers zero losses in this case. This is because the protocol transmits the perceptually significant samples in smaller packets of size 137 bytes. Once again, this illustrates that the packet sizing performed by a telehaptic protocol plays a crucial role in the loss experienced in the presence of TCP cross-traffic.

To summarize, the conditions for QoS compliance for adaptive sampling based telehaptic protocols are:

- The network needs to be provisioned for the peak telehaptic rate as telehaptic applications are highly sensitive to QoS violations.
- To meet the haptic delay constraint, we require that Equation (7.31) is satisfied.
- To avoid packet loss in presence of a TCP flow, large packet sizes should be avoided.

7.5.3 Opportunistic Adaptive Sampling Under Heterogeneous Cross-Traffic

We now comment on the behavior of the opportunistic adaptive sampling, that we designed in Chapter 4, under heterogeneous cross-traffic in shared network conditions. Since the rate control is based on the E2E delays, it is fairly simple to understand that with TCP streams, the opportunistic scheme backs off aggressively to the minimal telehaptic data rate that corresponds to the data rate of a static adaptive sampler. Therefore, the conclusions of Sections 7.5.1 and 7.5.2 apply to the opportunistic adaptive sampling scheme as well.

7.6 Conclusions

In this chapter, we presented a comprehensive assessment of the interplay between CBR and TCP flows in a shared network. Through the design of a mathematical model, we characterized bounds on the packet delays and jitter. We apply the analysis for evaluating the performance of CBR-based telehaptic protocols with heterogeneous cross-traffic. Using the expression for maximum haptic delay, we derived sufficiency condition for meeting the haptic delay QoS deadline in terms of network parameters. We showed that meeting delay QoS needs for haptic media is a sufficient condition for guaranteeing the audio and the video delay QoS compliance. We validate our analysis through extensive simulations. Further, we derived an important, simulation-driven condition for QoS compliance according to which a near-zero packet loss can be ensured if the telehaptic packets are small relative to the TCP packet size. We then evaluated the performance of adaptive sampling based protocols in presence of heterogeneous cross-traffic. We showed that the network should be provisioned for the peak telehaptic rate to prevent QoS violations.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

Realistic teleoperation on a shared network is not possible unless the challenges associated with telehaptic communication are fully addressed. In this regard, we studied the feasibility of telehaptic QoS compliance when the telehaptic flow shares the network with other cross-traffic flows. Through the design of a novel transport layer protocol, named *Haptics over Internet Protocol* (HoIP), we attempted to resolve some of the issues that a telehaptic flow typically encounters in a shared network.

We began by designing a low-delay packetization framework for telehaptic data on the forward channel in Chapter 3. We demonstrated that the protocol renders support for a wide variety of lossless as well as lossy haptic signal transmission schemes. We described the in-header delay notification mechanism for providing a low overhead, fine-grained, swift network feedback mechanism, thus enabling an accurate network monitoring and rate control. The time-out mechanism minimizes the possibility of signal drift at the receiver due to packet losses. The decoupling of the protocol from the haptic interface provides a plug and play type communication scheme that is independent and robust to nature of telehaptic application and the media devices being employed. This feature allows both the haptic interface and our protocol to be developed and upgraded asynchronously.

In Chapter 4, we proposed an opportunistic rate control mechanism on the forward channel with the objective of enhancing the haptic signal reconstruction at the teleoperator. This scheme uses the adaptive sampling threshold as the control parameter for varying the telehaptic data rate. We demonstrated that the proposed algorithm enhances the telehaptic throughput by a significant margin through both timely detection of the network state, as well as appropriate tuning of the data rate, without overwhelming the underlying shared network. The results presented show haptic QoS adherence even under heavy cross-traffic scenarios. The SNR measurements reveal that the opportunistic adaptive sampler outperforms the static adaptive sampler in terms of the quality of haptic signal reconstruction at the teleoperator. We also demonstrated that our opportunistic scheme is extremely friendly to exogenous cross-traffic sources in terms of network resource sharing.

Since the backward channel carries heterogeneous data types, we designed a multiplexing framework, in Chapter 5, for efficiently scheduling the transmission of the media frames in order to ensure a holistic telehaptic QoS compliance. While the haptic samples receive the highest multiplexing priority, the priority of audio and video frames is decided based on two selection rules. First, a first come, first serve (FCFS) based multiplexing rule that assigns priority depending on the time instants of frame generation. Second, a QoS-aware multiplexing rule that assigns a higher priority to the media with a stricter QoS constraint. Through simulations, we demonstrate that indeed the QoS insensitivity of the FCFS scheme can result in an audio jitter that is very close to the QoS limit. On the other hand, the QoS-aware rule eliminated the audio jitter completely due to its preemptive multiplexing technique. We adopted the packetization framework presented in Chapter 3 with minor modifications for accommodating the audio and the video payload.

We discovered two crucial drawbacks of an adaptive sampling based telehaptic communication. First, it results in a severe loss of fidelity so far as the signal reconstruction on the teleoperator end is concerned. This might be potentially catastrophic in a highly sensitive teleoperation, like telesurgery. Second, since the instantaneous data rate of the adaptive sampler depends on the nature of human-remote interaction, the network should be provisioned for the peak rate in order to guarantee telehaptic QoS-compliance. This provides no real economies in terms of network bandwidth requirement. We overcame these drawbacks through the design of a lossless, network-aware telehaptic congestion control technique called dynamic packetization module (DPM). This technique uses the packetization rate as the control parameter to perform efficient rate control in a lossless manner. 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. We compared DPM's performance with the state of the art in telehaptic protocols, and showed that DPM outperforms them. Finally, we derived sufficient conditions for a given network setting that guarantees haptic delay QoS adherence. Further, we extend the analysis to audio and video modalities, and showed that haptic QoS delay compliance implies audio and video QoS delay compliance as well.

Until this point, the performance evaluation of the two rate control techniques was performed under steady or slowly varying cross-traffic conditions. Additionally, we observed that none of the telehaptic protocol designs in the literature have considered performance assessment with coexisting TCP flows, TCP being the dominant protocol on the Internet. In Chapter 7 we carried out a comprehensive performance assessment of the interplay between two broad classes of telehaptic protocols and heterogeneous cross-traffic, consisting of TCP and CBR flows. Through the design of a mathematical model, we characterized bounds on the haptic delays, and approximated the haptic jitter. Using the expression for maximum haptic delay, we derived sufficiency condition for meeting the haptic delay QoS deadline in terms of network parameters. We showed that meeting delay QoS needs for haptic media is a sufficient condition for guaranteeing the audio and the video delay QoS compliance. We validate our analysis through extensive simulations. Further, we derived an important, simulation-driven condition for QoS compliance according to which a near-zero packet loss can be ensured if the telehaptic packets are small relative to the TCP packet size. We then evaluated the performance of adaptive sampling based protocols in presence of heterogeneous cross-traffic. We showed that the network should be provisioned for the peak telehaptic rate to prevent QoS violations.

8.2 Future Work

All of the performance evaluations and analyses presented in this thesis are carried out on a single bottleneck link. A realistic network may have a topology that is far more complicated than the considered one. In such cases, the telehaptic flows may be subjected to different network dynamics that pose additional challenges so far as the telehaptic QoS compliance is concerned. A more rigorous analysis to identify and circumvent these issues could be a potential direction going forward. Also, the analysis of the impact of existence of multiple TCP sources on the shared network on the telehaptic delay, jitter and packet losses could be an interesting avenue for future research. Furthermore, it is not obvious as to how multiple telehaptic sources running

DPM (or opportunistic adaptive sampler), encountering different networking conditions share the resources amongst themselves. Specifically, analyzing the intra-protocol fairness of rate adaptive telehaptic protocols is another potential research direction. In Chapters 4 and 6, we have considered SNR as the performance metric to gauge the improvement in performance of the proposed rate control techniques over the existing protocols. However, the implications of this improvement on the quality of the telehaptic activity has not been explored in this thesis. This could be another interesting direction for future research in telehaptic communication.

Bibliography

[1] Netflix: https://www.netflix.com/, 1997.

[2] Skype: https://www.skype.com, 2003.

[3] Youtube: https://www.youtube.com/, 2005.

[4] Geomagic phantom omni device reference: http://www.geomagic.com/en/products/phantom-omni/overview, july 2012.

[5] Hapi reference: www.h3dapi.org, july 2012.

[6] Festo exohand: http://www.festo.com/, 2013.

[7] Google hangout: https://hangouts.google.com, 2013.

[8] Cybergrasp user's guide v1.2, immersion corporation, San Jose, CA, USA, 2003.

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

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

[11] Guido Appenzeller, Isaac Keslassy, and Nick McKeown. *Sizing router buffers*, volume 34. ACM, 2004.

[12] Cagatay Basdogan, C-H Ho, and Mandayam A Srinivasan. Virtual environments for medical training: graphical and haptic simulation of laparoscopic common bile duct exploration. *IEEE/Asme Transactions On Mechatronics*, 6(3):269–285, 2001.

[13] Cagatay Basdogan, Chih-Hao Ho, Mandayam A Srinivasan, and Mel Slater. An experimental study on the role of touch in shared virtual environments. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(4):443–460, 2000.

[14] Amit Bhardwaj, Subhasis Chaudhuri, and Onkar Dabeer. Design and analysis of predictive sampling of haptic signals. *ACM Transactions on Applied Perception (TAP)*, 11(4):16, 2015.

[15] Leonardo Bonanni, Cati Vaucelle, Jeff Lieberman, and Orit Zuckerman. Taptap: a haptic wearable for asynchronous distributed touch therapy. In *CHI'06 extended abstracts on Human factors in computing systems*, pages 580–585. ACM, 2006.

[16] Azzedine Boukerche, Haifa Maamar, and Abu Hossain. An efficient hybrid multicast transport protocol for collaborative virtual environment with networked haptic. *Multimedia Systems*, 13(4):283–296, 2008.

[17] Lawrence S. Brakmo and Larry L Peterson. Tcp vegas: End to end congestion avoidance on a global internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–1480, 1995.

[18] Fernanda Brandi and Eckehard Steinbach. Low-complexity error-resilient data reduction approach for networked haptic sessions. In *Haptic Audio Visual Environments and Games (HAVE), 2011 IEEE International Workshop on*, pages 135–140. IEEE, 2011.

[19] Fernanda Brandi and Eckehard Steinbach. Perceptual coding of recorded telemanipulation sessions. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 985–988. ACM, 2011.

[20] Zhiwei Cen, Matt W Mutka, Danyu Zhu, and Ning Xi. Supermedia transport for teleoperations over overlay networks. In *International Conference on Research in Networking*, pages 1409–1412. Springer, 2005.

[21] Subhajit Chaudhury and Subhasis Chaudhuri. Volume preserving haptic pottery. In *Proceedings of Haptics Symposium (HAPTICS)*, 2014.

[22] Xiang Chen, Hongqiang Zhai, Xuejun Tian, and Yuguang Fang. Supporting qos in ieee 802.11 e wireless lans. *IEEE Transactions on Wireless Communications*, 5(8):2217–2227, 2006.

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

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

[25] Stella Clarke, Gerhard Schillhuber, Michael F Zaeh, and Heinz Ulbrich. Telepresence across delayed networks: a combined prediction and compression approach. In *Haptic Audio Visual Environments and their Applications, 2006. HAVE 2006. IEEE International Workshop on*, pages 171–175. IEEE, 2006.

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

[27] S Dodeller and ND Georganas. Transport layer protocols for telehaptics update messages. In *Proceedings of the 22nd Biennial Symposium on Communications*, 2004.

[28] Rushabh Doshi and Pei Cao. Streaming traffic fairness over low bandwidth wan links. In *Internet Applications. WIAPP 2003. Proceedings. The Third IEEE Workshop on*, pages 30–34. IEEE, 2003.

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

[30] Abdulmotaleb El Saddik. The potential of haptics technologies. *IEEE Instrumentation & Measurement Magazine*, 10(1):10–17, 2007.

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

[32] Sally Floyd, Andrei Gurtov, and Tom Henderson. The newreno modification to tcp's fast recovery algorithm. 2004.

[33] Masaki Fujimoto and Yutaka Ishibashi. Packetization interval of haptic media in networked virtual environments. In *Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games*, 2005.

[34] Takeshi Fujimoto, Yutaka Ishibashi, and Shinji Sugawara. Influences of inter-stream synchronization error on collaborative work in haptic and visual environments. In *Proceedings of Symposium on Haptic interfaces for virtual environment and teleoperator systems*, 2008.

[35] Vineet Gokhale, Subhasis Chaudhuri, and Onkar Dabeer. Hoip: A point-to-point haptic data communication protocol and its evaluation. In *Proceedings of Twenty First National Conference on Communications (NCC)*, 2015.

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

[37] Vineet Gokhale, Jayakrishnan Nair, and Subhasis Chaudhuri. Teleoperation over shared networks: When does it work? ACM Transactions on Multimedia Computing, Communications, and Applications (submitted).

[38] Vineet Gokhale, Jayakrishnan Nair, and Subhasis Chaudhuri. Opportunistic adaptive haptic sampling on forward channel in telehaptic communication. In *Haptics Symposium (HAPTICS)*. IEEE, 2016.

[39] Vineet Gokhale, Jayakrishnan Nair, and Subhasis Chaudhuri. Congestion control for network-aware telehaptic communication. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2017.

[40] Bur Goode. Voice over internet protocol (voip). *Proceedings of the IEEE*, 90(9):1495–1517, 2002.

[41] Vikram Gupta, Srikanth V Krishnamurthy, and Michalis Faloutsos. Improving the performance of tcp in the presence of interacting udp flows in ad hoc networks. In *International Conference on Research in Networking*, pages 64–75. Springer, 2004.

[42] Sangtae Ha, Injong Rhee, and Lisong Xu. Cubic: a new tcp-friendly high-speed tcp variant. *ACM SIGOPS Operating Systems Review*, 42(5):64–74, 2008.

[43] Kenji Hikichi, Hironao Morino, Isamu Arimoto, Kaoru Sezaki, and Yasuhiko Yasuda. The evaluation of delay jitter for haptics collaboration over the internet. In *Proceedings of Global Telecommunications Conference, 2002. GLOBECOM*, volume 2, 2002.

[44] Kenji Hikichi, Hironao Morino, I Fukuda, S Matsumoto, Yasuhiko Yasuda, I Arimoto, M Iijima, and K Sezaki. Architecture of haptics communication system for adaptation to network environments. In *Proc. IEEE ICME*, pages 563–566, 2001.

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

[46] Peter Hinterseer, E Steinbach, and Subhasis Chaudhuri. Perception-based compression of haptic data streams using kalman filters. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 5, pages V–V. IEEE, 2006.

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

[48] Peter Hinterseer and Eckehard Steinbach. A psychophysically motivated compression approach for 3d haptic data. In *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2006 14th Symposium on*, pages 35–41. IEEE, 2006.

[49] Sandra Hirche, Peter Hinterseer, Eckehard Steinbach, and Martin Buss. Transparent data reduction in networked telepresence and teleaction systems. part i: Communication without time delay. *Presence: Teleoperators and Virtual Environments*, 16(5):523–531, 2007.

[50] Peter F Hokayem and Mark W Spong. Bilateral teleoperation: An historical survey. *Automatica*, 42(12):2035–2057, 2006.

[51] Sosuke Hoshino, Yutaka Ishibashi, Norishige Fukushima, and Shinji Sugawara. Qoe assessment in olfactory and haptic media transmission: Influence of inter-stream syn-

chronization error. In *Proceedings of International Workshop Technical Committee on Communications Quality and Reliability (CQR)*, 2011.

[52] Eiichi Isomura, Shuji Tasaka, and Toshiro Nunome. A multidimensional qoe monitoring system for audiovisual and haptic interactive ip communications. In *Proceedings of Consumer Communications and Networking Conference (CCNC)*, 2013.

[53] Van Jacobson, Ron Frederick, Steve Casner, and H Schulzrinne. Rtp: A transport protocol for real-time applications. 2003.

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

[55] Jung Kim, Hyun Kim, Boon K Tay, Manivannan Muniyandi, Mandayam A Srinivasan, Joel Jordan, Jesper Mortensen, Manuel Oliveira, and Mel Slater. Transatlantic touch: A study of haptic collaboration over long distance. *Presence: teleoperators and virtual environments*, 13(3):328–337, 2004.

[56] Keehoon Kim, James Edward Colgate, Julio J Santos-Munné, Alexander Makhlin, and Michael A Peshkin. On the design of miniature haptic devices for upper extremity prosthetics. *IEEE/ASME Transactions on Mechatronics*, 15(1):27–39, 2010.

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

[58] Dale A Lawrence. Stability and transparency in bilateral teleoperation. *IEEE transactions on robotics and automation*, 9(5):624–637, 1993.

[59] Jae-Young Lee and Shahram Payandeh. Performance evaluation of haptic data compression methods in teleoperation systems. In *World Haptics Conference (WHC), 2011 IEEE*, pages 137–142. IEEE, 2011.

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

[61] Seokhee Lee, Sungtae Moon, and JongWon Kim. A network-adaptive transport scheme for haptic-based collaborative virtual environments. In *Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games*, page 13. ACM, 2006.

[62] Robert W Lindeman, Robert Page, Yasuyuki Yanagida, and John L Sibert. Towards full-body haptic feedback: the design and deployment of a spatialized vibrotactile feedback system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 146–149. ACM, 2004.

[63] Jacques Marescaux, Joel Leroy, Francesco Rubino, Michelle Smith, Michel Vix, Michele Simone, and Didier Mutter. Transcontinental robot-assisted remote telesurgery: feasibility and potential applications. *Annals of surgery*, 235(4):487–492, 2002.

[64] Alan Marshall, Kian Meng Yap, and Wai Yu. Providing qos for networked peers in distributed haptic virtual environments. *Advances in Multimedia*, 2008, 2008.

[65] Soju Matsumotoy, Ichiro Fukuday, Hironao Morinoy, Kenji Hikichiy, Kaoru Sezakiz, and Yasuhiko Yasuda. The influences of network issues on haptic collaboration in shared virtual environments. 2000.

[66] Martin Mauve, Volker Hilt, Christoph Kuhmunch, and Wolfgang Effelsberg. Rtp/i-toward a common application level protocol for distributed interactive media. *IEEE Transactions on Multimedia*, 3(1):152–161, 2001.

[67] Margaret L McLaughlin, Gaurav Sukhatme, Cyrus Shahabi, Joao Hespanha, Antonio Ortega, and Gerard Medioni. The haptic museum. In *Proceedings of the EVA 2000 conference on electronic imaging and the visual arts*, 2000.

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

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

[70] Douglas C Montgomery. *Introduction to statistical quality control*. John Wiley & Sons, 2007.

[71] Dan Morris, Neel Joshi, and Kenneth Salisbury. Haptic battle pong: High-degree-of-freedom haptics in a multiplayer gaming environment. In *Experimental gameplay workshop, GDC*. Citeseer, 2004.

[72] Takamichi Nakamoto and Takao Yamanaka. Odor reproduction with movie and its application to teleolfaction. *Intelligent Systems for Machine Olfaction: Tools and Methodologies. Hines EL, & Leeson MS (Eds), IGI Globa*, pages 126–1153, 2011.

[73] Qassim Nasir and Enas Khalil. Perception based adaptive haptic communication protocol (pahcp). In *Computer Systems and Industrial Informatics (ICCSII), 2012 International Conference on*, pages 1–6. IEEE, 2012.

[74] NS3. The network simulator: http://www.nsnam.org/, 2011.

[75] Panagiotis Papadimitriou and Vassilis Tsaoussidis. On transport layer mechanisms for real-time qos. *J. Mobile Multimedia*, 1(4):342–363, 2006.

[76] Joann Peck and Terry L Childers. To have and to hold: The influence of haptic information on product judgments. *Journal of Marketing*, 67(2):35–48, 2003.

[77] Angelika Peer, Sandra Hirche, Carolina Weber, Inga Krause, Martin Buss, Sylvain Miossec, Paul Evrard, Olivier Stasse, Ee Sian Neo, Abderrahmane Kheddar, et al. Intercontinental cooperative telemanipulation between germany and japan. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2715–2716. IEEE, 2008.

[78] Li Ping, Lu Wenjuan, and Sun Zengqi. Transport layer protocol reconfiguration for network-based robot control system. In *Networking, Sensing and Control, 2005. Proceedings. 2005 IEEE*, pages 1049–1053. IEEE, 2005.

[79] Jon Postel. User datagram protocol. Technical report, 1980.

[80] Jon Postel. Transmission control protocol. 1981.

[81] Microsoft Press et al. *Microsoft Win 32 Application Programming Interface: The Programmer's Reference*, volume 1. Microsoft Pr, 1992.

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

[83] Gabriel Robles-De-La-Torre. The importance of the sense of touch in virtual and real environments. *IEEE Multimedia*, 13(3):24–30, 2006.

[84] Christian Rohner, Erik Nordström, Per Gunningberg, and Christian Tschudin. Interactions between tcp, udp and routing protocols in wireless multi-hop ad hoc networks. In *Proc. IEEE ICPS Workshop on Multi-hop Ad hoc Networks: from theory to reality (REALMAN05)*, 2005.

[85] Seungwan Ryu, Christopher Rump, and Chunming Qiao. Advances in internet congestion control. *IEEE Communications Surveys & Tutorials*, 5(1):28–39, 2003.

[86] K Sajith, Preeti Gopal, and Subhasis Chaudhuri. Hand tremor analysis using deformable object manipulation in a haptic environment. In *2013 IEEE Point-of-Care Healthcare Technologies (PHT)*, pages 13–17. IEEE, 2013.

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

[88] Hidenari Sawashima, Yoshiaki Hori, Hideki Sunahara, and Yuji Oie. Characteristics of udp packet loss: Effect of tcp traffic. In *Proceedings of INET97: The Seventh Annual Conference of the Internet Society*, 1997.

[89] Clemens Schuwerk, Nakul Chaudhari, and Eckehard Steinbach. An area-of-interest based communication architecture for shared haptic virtual environments. In *Haptic Audio Visual Environments and Games (HAVE), 2013 IEEE International Symposium on*, pages 57–62. IEEE, 2013.

[90] Clemens Schuwerk, Wolfgang Freund, and Eckehard Steinbach. Low-delay compression of polygon mesh deformation data for remote haptic interaction with simulated deformable objects. In *Haptics Symposium (HAPTICS), 2016 IEEE*, pages 229–234. IEEE, 2016.

[91] Clemens Schuwerk, Xiao Xu, Rahul Chaudhari, and Eckehard Steinbach. Compensating the effect of communication delay in client-server–based shared haptic virtual environments. *ACM Transactions on Applied Perception (TAP)*, 13(1):5, 2015.

[92] Shervin Shirmohammadi and Nicolas D Georganas. An end-to-end communication architecture for collaborative virtual environments. *Computer Networks*, 35(2):351–367, 2001.

[93] KG Sreeni, K Priyadarshini, AK Praseedha, and Subhasis Chaudhuri. Haptic rendering of cultural heritage objects at different scales. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, pages 505–516. Springer, 2012.

[94] Mandayam A Srinivasan and Cagatay Basdogan. Haptics in virtual environments: Taxonomy, research status, and challenges. *Computers & Graphics*, 21(4):393–404, 1997.

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

[96] Eckehard Steinbach, Sandra Hirche, Julius Kammerl, Iason Vittorias, and Rahul Chaudhari. Haptic data compression and communication. *IEEE Signal Processing Magazine*, 28(1):87–96, 2011.

[97] Eckehard Steinbach, Sandra Hirche, Julius Kammerl, Iason Vittorias, and Rahul Chaudhari. Haptic data compression and communication. *IEEE Signal Processing Magazine*, 28(1):87–96, 2011.

[98] Jinsheng Sun, Moshe Zukerman, King-Tim Ko, Guanrong Chen, and Sammy Chan. Effect of large buffers on tcp queueing behavior. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 751–761. IEEE, 2004.

[99] Nobuhiro Suzuki and Seiichiro Katsura. Evaluation of qos in haptic communication based on bilateral control. In *Proceedings of International Conference on Mechatronics (ICM)*, 2013.

[100] Tim Szigeti and Christina Hattingh. Quality of service design overview. *Cisco, San Jose, CA, Dec*, 2004.

[101] Uras Tos and Tolga Ayav. Adaptive rtp rate control method. In *Proceedings of Computer Software and Applications Conference Workshops (COMPSACW)*, 2011.

[102] Yutaka Uchimura, Kouhei Ohnishi, and Takahiro Yakoh. Bilateral robot system on the real time network structure. In *Advanced Motion Control, 2002. 7th International Workshop on*, pages 63–68. IEEE, 2002.

[103] Curtis Villamizar and Cheng Song. High performance tcp in ansnet. *ACM SIGCOMM Computer Communication Review*, 24(5):45–60, 1994.

[104] Arun Vishwanath and Vijay Sivaraman. Routers with very small buffers: Anomalous loss performance for mixed real-time and tcp traffic. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*, pages 80–89. IEEE, 2008.

[105] Arun Vishwanath, Vijay Sivaraman, and George N Rouskas. Considerations for sizing buffers in optical packet switched networks. In *INFOCOM 2009, IEEE*, pages 1323–1331. IEEE, 2009.

[106] Arun Vishwanath, Vijay Sivaraman, and George N Rouskas. Anomalous loss performance for mixed real-time and tcp traffic in routers with very small buffers. *IEEE/ACM Transactions on Networking*, 19(4):933–946, 2011.

[107] Iason Vittorias, Julius Kammerl, Sandra Hirche, and Eckehard Steinbach. Perceptual coding of haptic data in time-delayed teleoperation. In *EuroHaptics conference, 2009 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2009. Third Joint*, pages 208–213. IEEE, 2009.

[108] David X Wei, Cheng Jin, Steven H Low, and Sanjay Hegde. Fast tcp: motivation, architecture, algorithms, performance. *IEEE/ACM Transactions on Networking (ToN)*, 14(6):1246–1259, 2006.

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

[110] Damon Wischik and Nick McKeown. Part i: Buffer sizes for core routers. *ACM SIGCOMM Computer Communication Review*, 35(3):75–78, 2005.

[111] Xiao Xu, Burak Cizmeci, Clemens Schuwerk, and Eckehard Steinbach. Haptic data reduction for time-delayed teleoperation using the time domain passivity approach. In *IEEE World Haptics Conference (WHC)*, pages 512–518, 2015.

[112] Shun Yao, Fei Xue, Biswanath Mukherjee, SJ Ben Yoo, and Sudhir Dixit. Electrical ingress buffering and traffic aggregation for optical packet switching and their effect on tcp-level performance in optical mesh networks. *IEEE Communications Magazine*, 40(9):66–72, 2002.

[113] Mehrdad Hosseini Zadeh, David Wang, and Eric Kubica. Perception-based lossy haptic compression considerations for velocity-based interactions. *Multimedia Systems*, 13(4):275–282, 2008.

[114] Hongqiang Zhai, Xiang Chen, and Yuguang Fang. How well can the ieee 802.11 wireless lan support quality of service? *IEEE Transactions on Wireless Communications*, 4(6):3084–3094, 2005.

Publications from this research

Published Work

1. V. Gokhale, O. Dabeer, S. Chaudhuri, "HoIP: Haptics over internet protocol", in *Proceedings of International Symposium on Haptics Audio Visual Environments and Games*, Istanbul, 2013.

2. V. Gokhale, S. Chaudhuri and O. Dabeer, "HoIP: A point-to-point haptic data communication protocol and its evaluation", in *Proceedings of National Conference on Communications*, Mumbai, 2015.

3. V. Gokhale, J. Nair and S. Chaudhuri, "Opportunistic adaptive haptic sampling on forward channel in telehaptic communication", in *Proceedings of Haptics Symposium*, Philadelphia, 2016.

4. V. Gokhale, J. Nair and S. Chaudhuri, "Congestion control for network-aware telehaptic communication", *ACM Transactions on Multimedia Computing, Communications, and Applications*, 13, 2, Article 17, 2017.

Under Review

1. V. Gokhale, J. Nair and S. Chaudhuri, "Interplay between telehaptic and cross-traffic streams" submitted to *International Symposium on Haptics Audio Visual Environments and Games*, 2017.

2. V. Gokhale, J. Nair and S. Chaudhuri, "Teleoperation over shared networks: When does it work?" submitted to *ACM Transactions on Multimedia Computing, Communications, and Applications*.

Acknowledgments

Before signing off, I take this opportunity to thank a few people who have directly or indirectly played important roles in helping me navigate smoothly during this wonderful journey. First of all, a huge thanks to my doctoral supervisor Prof. Subhasis Chaudhuri whose relentless guidance and support has been instrumental in making this journey a pleasurable one. Your vision, innate ability to see through complicated barriers and conceive new research ideas are phenomenal. Above all, the empathy towards students and the humility that you display make you an extraordinary human being.

I am also grateful to Prof. Jayakrishnan Nair for giving interesting perspectives to my research. I am fortunate that our casual technical discussion turned into a more serious academic relationship. Thank you for agreeing to be my co-supervisor. Your coming on-board has substantially refined my thesis objectives. Your effortless articulation of complex ideas, clarity of thoughts, and the constant strive for perfection are remarkable.

I am also deeply indebted to Dr. Onkar Dabeer for his research guidance during the initial days of my research. Your constant motivation boosted my confidence immensely. Your commitment to the work during your stay in India was unparalleled. I would like to extend my thanks to Profs. Animesh Kumar and Saravanan Vijayakumaran for serving on my RPC and providing insightful suggestions that helped in shaping the thesis. I would also like to show my gratitude to the thesis reviewers for providing timely and invaluable feedback on my work.

I am also grateful to the staff of EE office and Academic section of IIT Bombay for being extremely student-friendly in terms of diligently handling all the academic-related formalities. Special thanks to Bharti Centre for Communication for generously disbursing travel funds for attending international conferences.

None of this would have been possible without the constant encouragement and backing of a decade old, close companionship of Varun, Sachin, Vinodh and Vishal. I have also been fortunate enough to have found a lovely company of friends without whom my IITB stay would have been rather uneventful. I would specially like to thank Prasanna, Amit, Abhisekh, Santosh, Nanditha, Amitalok, Prachi, Vinita, Agnes and Jeevan. A big shout-out to my friends from Vision and Image Processing Lab - Renu, Sunil, Sreeni, Sajith, Ketan, Avik, Subhajit, Amit, Neha, Omkar, Dwaipayan, Saurabh, Sayan, Fasil and Feroz. You guys make the lab all the more amicable.

It would be impossible to imagine this journey without the unconditional persuasion, trust and love of my parents - Leela and Heramba Gokhale. You both have always stood by me in this roller-coaster ride, and given me the courage to take tough decisions in life. It would only be appropriate to say that I am space constrained here to express my gratitude towards you. I am also thankful to my brother Vikram and sister-in-law Rashmi for showing immense belief in me and tolerating all my idiosyncrasies. I consider myself extremely fortunate to have you both in my life. I also take this opportunity to welcome my little nephew Nishchal to this world. You are a lovely bundle of joy. A great share of gratitude also goes to my in-laws - Malini and Shreedhar Naik, for showing pronounced belief in me and being extremely supportive in all aspects of life. Special thanks to Divya and Shyamsundar for showering tremendous love on me.

It would only be incomplete to wind this up without showing gratitude to my wife Sowmya. I thank you deeply for stopping by me and realizing that we both would be only happier in spending the rest of our lives together. I am grateful to you for affording to compromise on a few of your dreams, and gearing up for a rather boring life with me. Our discussions and debates on worldly matters add a different flavor to my perspectives. I particularly attribute the ramping up of my final year's PhD work to you. Thanks for backing me up during all my ups and downs. It has truly been an enjoyable journey with you so far. I look forward to a blissful and an eventful life ahead.

Finally, I bow down to almighty God for giving me the strength to carry out this research work. Nothing of this would have been possible without your blessings.

Date: ______________ ______________________

Vineet Gokhale

(Roll No. 114070002)