Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik
Lehrstuhl für Medientechnik

Client/Server-Based Shared Haptic Virtual Environments: Perceptual Data Reduction, Communication, and Delay Compensation

Dipl.-Ing. Univ. Clemens Schuwerk

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr.-Ing. habil. Dirk Wollherr
Prüfer der Dissertation:	1.	Prof. Dr.-Ing. Eckehard Steinbach
	2.	Prof. Dr. Subhasis Chaudhuri

Die Dissertation wurde am 20.04.2016 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 03.09.2016 angenommen.

Abstract

Virtual Reality technology promises the sensation of being present in computer-simulated virtual environments (VEs). This experience, however, is not complete without the ability to physically interact and manipulate objects in the virtual space. Collaborative virtual environments enhanced with haptic force feedback, so-called Shared Haptic Virtual Environments (SHVEs), enable the remote collaboration between distant users and facilitate several promising applications in the areas of education, training, teaching, or entertainment.

SHVEs can be realized using a client/server architecture. In this architecture, the clients maintain a copy of the virtual environment to locally render the scene visually and to calculate force feedback, which is displayed to the user through a haptic device. A centralized physics simulation, running on the server, calculates the object states in the VE, that is, the objects' position, rotation and deformation, based on the users' interactions.

This architecture ensures a consistent simulation state and makes the SHVE robust against instabilities in the coupling between the user, the haptic device and the VE. On the other hand, it requires the transmission of the haptic device state from the clients to the server and the simulated object states vice versa, which demands high transmission capacities. Furthermore, the round-trip-time (RTT) in the communication between the clients and the server leads to changed force magnitudes rendered at the clients and a loss of transparency, that is, the RTT deteriorates the user experience.

This thesis addresses these challenges and presents novel data reduction approaches to reduce the transmission capacities required during the interaction with remotely simulated rigid movable and deformable objects. These schemes explicitly consider the human visual and haptic perceptual limitations to avoid the transmission of unnecessary data. The evaluation with various subjective user studies shows that the proposed schemes find a good balance between data reduction and fidelity.

Furthermore, the transparency of the client/server architecture in the presence of communication delays is systematically analyzed for cooperative multi-user interactions. The analyses reveal guidelines on the tolerable communication delay, below which the effects of the RTT are not perceivable by the users. Adaptive force rendering algorithms to compensate for the increased force magnitudes, if this delay is exceeded, are presented. The experimental results, including subjective tests, verify the theoretical analyses and the applicability of the proposed compensation schemes.

Kurzfassung

Seit mehr als einem Jahrzehnt werden Technologien erforscht, um dem Benutzer ein vollkommenes Eintauchen in computergenerierte virtuelle Umgebungen zu ermöglichen. Sogenannte VR-Brillen vermitteln zum Beispiel räumliche Tiefe in einer virtuellen Szenerie, um den Realismus zu erhöhen und diesem Ziel einen Schritt näher zu kommen. Dieses rein visuelle Erlebnis ist jedoch ohne die Möglichkeit, virtuelle Objekte zu fühlen und haptisch mit ihnen interagieren zu können, nicht komplett. Virtuelle Umgebungen mit einer solchen haptischen Rückkopplung, in denen zusätzlich mehrere Benutzer über geographische Distanzen hinweg gemeinsam interagieren können, ermöglichen zahlreiche vielversprechende Anwendungen in den Bereichen Bildung, Training oder Unterhaltung.

Solche Anwendungen können mit Hilfe einer Client/Server Architektur realisiert werden. Dabei verfügt jeder Client über eine lokale Kopie der virtuellen Umgebung, um die Szene graphisch darzustellen und Kraftsignale zu berechnen, welche über ein haptisches Ein-Ausgabegerät dem Benutzer dargestellt werden. Der zentrale Server berechnet das physikalische Verhalten, also die Bewegung, Rotation oder Verformung von virtuellen Objekten anhand der Interaktionen der einzelnen Benutzer.

Diese Architektur gewährleistet Simulationskonsistenz und gleichzeitig Robustheit gegenüber Instabilitäten in dem gekoppelten System aus Benutzer, haptischem Ein-Ausgabegerät und virtueller Umgebung. Die Benutzereingaben und das Ergebnis der physikbasierten Simulation müssen jedoch zwischen den Clients und dem Server übertragen werden, damit der Server das physikalische Verhalten der Objekte berechnen kann und die lokalen Kopien der virtuellen Umgebung aktuell gehalten werden. Dies stellt hohe Anforderungen an die zugrundeliegende Kommunikationsarchitektur. Zusätzlich führen Kommunikationsverzögerungen dazu, dass den Benutzern veränderte Kraftsignale dargestellt werden und die Transparenz des Systems verloren geht.

Diese Dissertation beschäftigt sich mit diesen Herausforderungen. Es werden Verfahren zur Datenreduktion vorgestellt, um das Ergebnis der physikbasierten Simulation vom Server zu den Clients zu übertragen. Diese Verfahren sind wahrnehmungsbasiert, das heißt, sie verwenden ein mathematisches Wahrnehmungsmodell um die Grenzen der menschlichen visuellen und haptischen Wahrnehmung zu berücksichtigen. Dadurch wird die Übertragung von nicht wahrnehmbarer Information vermieden und eine möglichst effiziente Kommunikation gewährleistet. Die Evaluierung mit subjektiven Tests zeigt, dass die vorgestellten Verfahren einen sehr guten Kompromiss zwischen Datenreduktion und visuell-haptischer Darstellungsqualität erreichen.

Außerdem wird die Transparenz der verwendeten Architektur systematisch analysiert. Die Analysen ergeben theoretische Grenzen für die tolerierbare Kommunikationsverzögerungen. Sind die Verzögerungen unterhalb dieser Grenzen, bleiben die Auswirkungen auf die berechneten Kräfte an den Clients unterhalb der menschlichen Wahrnehmungsschwellen. Für den Fall dass diese Verzögerungsgrenzen überschritten werden, präsentiert diese Arbeit außerdem Algorithmen um die erhöhten Kräfte an den Clients zu kompensieren. Die Ergebnisse der Evaluierung verifizieren die Anwendbarkeit der vorgestellten Verfahren.

Acknowledgements

The work presented in this dissertation was carried out as a member of the academic staff at the Chair of Media Technology (LMT) at the Technical University of Munich. My research was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant agreement no. 258941.

First of all, I would like to express my gratitude to my supervisor Prof. Eckehard Steinbach for inviting me into his team and giving me the opportunity of being part of the "Pro Haptics" project, which covered an exciting area that had been completely novel for me at this time. I would like to thank Prof. Steinbach for his ongoing support and academic advice. Despite his busy schedule, he has always been accessible for discussing ideas, providing feedback and giving me fresh perspectives.

I would like to thank Prof. Subhasis Chaudhuri for the stimulating discussions about haptic technology in general and my work in particular during his visits in Germany and at various conferences in the past years, and for agreeing to become the second examiner.

My sincere appreciation also goes to all my current and former colleagues of the haptics group at LMT, especially to Dr. Rahul Chaudhari, Xiao Xu, Burak Cizmeci, Fernanda Brandi, Dr. Giulia Paggetti, Dr. Nicolas Alt and Dr. Julius Kammerl. At the beginning of my PhD, Dr. Chaudhari and Dr. Kammerl provided important impulses for my research. The excellent collaboration with Dr. Chaudhari continued over the years not only in technical topics, but also during the Computational Haptics Lab. I would like to thank Xiao Xu for the various fruitful discussions. I would like to thank Dr. Giulia Paggetti for providing me a deeper insight into psychophysical techniques and research. I would also like to thank all my students, especially Wolfgang Freund, who contributed to this dissertation. Thanks go also to Matti Strese with whom I've collaborated on the very interesting topic of tool-based surface classification and retrieval.

My appreciation also goes to the professional administrative support provided by Dr. Martin Maier, Ingrid Jamrath, Simon Krapf and Brigitte Vrochte. Thanks go to all my current and former colleagues who made the time at the chair so special and enjoyable.

My final thanks go to my parents, my brother, and my sister who always supported me in every possible way, and to my wonderful girlfriend Nicole who gave me support and motivation particularly during the months of writing this thesis.

Clemens Schuwerk, Munich, December 28, 2016

Contents

Notation

Abbreviations

Abbreviation	Description	Definition
AABB	Axis-Aligned Bounding Box	page 20
ANOVA	Analysis of Variance	page 55
AOI	Area-of-Interest	page 63
CNS	Central Nervous System	page 7
CS	Client/Server	page 2
DOF	Degree-of-Freedom	page 12
DR	Dead Reckoning	page 32
GPU	Graphics Processing Unit	page 22
HDR	Haptic Dead Reckoning	page 40
HVE	Haptic Virtual Environment	page 2
JND	Just Noticeable Difference	page 9
LTI	Linear Time Invariant	page 5
MMT	Model-Mediated Teleoperation	page 27
MOS	Mean Opinion Score	page 71
MSE	Mean Squared Error	page 67
ODE	Ordinary Differential Equation	page 17
P2P	Peer-to-Peer	page 2
PD	Perceptual Deadband	page 30
PDE	Partial Differential Equation	page 17
PC	Passivity Controller	page 13
PO	Passivity Observer	page 13
RTT	Round-Trip-Time	page 3
SDT	Signal Detection Theory	page 98
SHVE	Shared Haptic Virtual Environment	page 2
SLERP	Spherical Linear Interpolation	page 51
UDP	User Datagram Protocol	page 30
VE	Virtual Environment	page 1
VR	Virtual Reality	page 1
VDR	Visual Dead Reckoning	page 39
WF	Weber-fraction	page 9
ZOH	Zero-Order-Hold	page 9

Scalars and vectors

x	scalar
$\|x\|$	absolute value of scalar x
$x(\cdot)$	scalar function
$\boldsymbol{x}$	vector
$\|\boldsymbol{x}\|$	Euclidean norm of vector $\boldsymbol{x}$
$\boldsymbol{x}(\cdot)$	vector function
$\boldsymbol{x} \cdot \boldsymbol{y}$	dot product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$\boldsymbol{X}$	matrix
$\dot{x}, \ddot{x}$	equivalent to $\frac{d}{dt}x$, $\frac{d^2}{dt^2}x$

Subscripts and superscripts

$x_{c,i}$	signal x at the client associated with user i
$x_{s,i}$	signal x at the server associated with user i
$x_{d,i}$	signal x associated with the haptic device of user i
x_o	signal x associated with an object
x_p	signal x associated with the virtual proxy
$\bar{x}$	mean of x
$\hat{x}$	estimated/predicted value of x

Symbols

$b_{(\cdot)}$	damping coefficient
c	Weber-fraction
E	energy
$f_{(\cdot)}$, $\boldsymbol{f}_{(\cdot)}$	force
$k_{(\cdot)}$	stiffness coefficient
$m_{(\cdot)}$	mass
p	deadband parameter (traffic control parameter)
$r_{(\cdot)}$, $\boldsymbol{r}_{(\cdot)}$	rotation
$\dot{r}_{(\cdot)}$, $\dot{\boldsymbol{r}}_{(\cdot)}$	rotational velocity
R_i	round-trip-time of client i
$\boldsymbol{S}$	object state
S	stimulus
s	independent variable in the Laplace domain
t	time, independent variable in the time-domain
$t_{(\cdot)}$	discrete time step
$T_{f,i}$	time delay on the forward channel for client i (from the client to the server)
$T_{b,i}$	time delay on the backward channel for client i (from the server to the client)
T_c	computation time
T_s	sampling time
$\boldsymbol{u}_{(\cdot)}$	vertex displacement
$x_{(\cdot)}$, $\boldsymbol{x}_{(\cdot)}$	position
$\dot{x}_{(\cdot)}$, $\dot{\boldsymbol{x}}_{(\cdot)}$	velocity
$\ddot{x}_{(\cdot)}$, $\ddot{\boldsymbol{x}}_{(\cdot)}$	acceleration
Z	mechanical impedance

Chapter 1

Introduction

Virtual Reality (VR), although being subject of research for more than a decade, has seen a major boost and has attracted public attention in recent years due to the emergence of consumer-targeted head-mounted displays like the Oculus Rift [21]. VR technology promises the sensation of being present in remote spaces or artificially designed virtual environments (VEs), that is, the feeling of actually being there.

This experience, however, is not complete without the ability to *physically* touch, interact, manipulate and feel virtual objects. The reason behind this is the fact that humans heavily rely on haptic percepts during the daily interaction with their environment. Haptic technology aims at filling this gap by conveying haptic feedback to the human user. This includes contact forces and torques (kinesthetic feedback) as well as surface properties such as roughness or friction (cutaneous feedback). The goal is to provide a coherent multi-modal, audio-visual-haptic experience that allows the user to fully immerse into the VE.

Research in the area of haptics, that is, science, technology, software, and applications related to the sense of touch, has repeatedly shown that enhancing traditional human-machine interaction with the haptic modality brings several benefits. Firstly, it increases the realism of technical systems [22] and improves the sense of immersion into virtual environments [23]. Secondly, it also improves task performance [24]. For these reasons, many applications in the areas of telepresence, telesurgery, education, teaching, training, or entertainment heavily benefit from the integration of the haptic modality [25].

Kinesthetic haptic feedback is typically conveyed through hand-held human-machine interfaces, called kinesthetic haptic devices. They sense the motion of a tool held by the user and simultaneously display force/torque feedback signals. This bidirectional exchange of information is the most distinguishing feature of haptic devices, when compared to other human-machine interfaces such as a computer mouse or keyboard [26].

This unique feature also constitutes the main challenge in the design of virtual applications enhanced with force feedback. Kinesthetic haptic devices couple the human user with the VE and exchange energy bidirectionally [26]. The haptic device closes a control loop between the user and the simulated environment. The stability of this loop requires the sampling of the tool's position/velocity and the rendering of forces/torques displayed through the device at rates of typically 1 kHz or higher. This constitutes a computational burden for software and algorithms to artificially recreate real-world interactions in virtual applications.

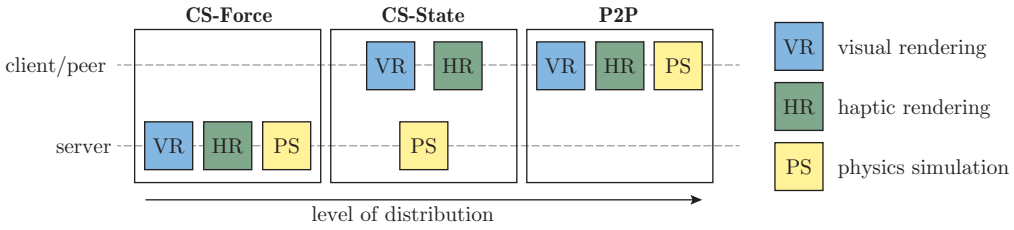


Figure 1.1: Different architectures, from fully centralized to fully distributed, have been conceived for the implementation of SHVEs.

In the last decades, advances in communication technology have also stimulated research on collaborative virtual environments, that is, virtual applications where several users can collaborate with each other [27], [28]. It has been shown that haptic feedback incorporated into such applications increases the sense of togetherness between users collaborating on a common task [24]. Applications for so-called *Shared Haptic Virtual Environments* (SHVEs) [29] include, among others, collaborative surgical training [30]–[32], assembly simulation [33], collaborative virtual prototyping [34], [35] or multi-user gaming [25], [36].

The users of a SHVE are possibly distributed around the globe and connected over a communication network. This enables the remote collaboration between distant users, but also imposes several challenges. The application has to be provided to all users and kept consistent among them.

The rendering pipeline of a haptic virtual environment (HVE) typically consists of three major parts: visual rendering, haptic rendering and a physics simulation, which calculates the dynamics in the virtual world. In a SHVE with geographically distributed users, parts of the application can be calculated either on a centralized server or shifted to the user-side. Three different architectures based on the client/server (CS) or the peer-to-peer (P2P) communication paradigms have been conceived for the implementation of SHVEs. They differ in the level of distribution as illustrated in Figure 1.1.

In the fully centralized implementation, denoted as the *CS-Force architecture*, a centralized server fully handles the simulation of the SHVE (see Figure 1.1). The clients transmit their device state to the server and receive the haptic force/torque feedback signals together with a video stream representing the view on the virtual environment. This ensures a consistent application state, but introduces other problems.

Each client/server connection resembles a traditional bilateral teleoperation system [37], where the power variables velocity and force are transmitted over the communication network, which closes a global control loop. The data transmission between geographically distributed users is typically afflicted with time delay, delay variation, and packet loss. This is particularly true for packet-switched communication using todays Internet's best-effort service. Even a small delay of several milliseconds in the communication between the server and the client leads to instabilities, as experienced in [38] and analyzed in [39].

Furthermore, the transmission of position/velocity and force/torque signals in addition to the video stream imposes strong demands on the communication network [40]–[42]. To keep the delay as small as possible, haptic sensor readings are typically packetized and trans-

mitted once available. The aforementioned high sampling and rendering rates lead to 1000 or more haptic data packets per second transmitted bi-directionally between the clients and the server.

In the other extreme, the fully distributed *P2P architecture* (see Figure 1.1), each participating user locally renders the virtual environment and also deploys a local physics simulation. The users exchange information about their interactions to drive the individual simulations. This leads to consistency issues in the presence of communication delays. The local physics simulations use outdated information about the other users and become inconsistent [29]. Maintaining consistency is particularly challenging for complex simulations and during the collaborative interaction with shared objects [35]. The advantages of the P2P approach, however, are the responsiveness of the HVE and the avoidance of delay in the control loop between the haptic device and the simulation, as the object dynamics and the force feedback are calculated locally at each user.

When considering the level of distribution, the so-called *CS-State architecture* lies in between the fully centralized and the fully distributed architecture. It maintains a copy of the VE at the clients to locally render the scene visually and haptically (compare Figure 1.1) [29]. The object dynamics are calculated at the server by means of a centralized physics simulation and transmitted to the clients to update the local copies of the VE. This approach combines the advantages of the two aforementioned architectures. The centralized physics simulation ensures a consistent object state and the local force rendering at the clients avoids delay in the control loop between the haptic device and the VE. The CS-State architecture, however, requires the transmission of object states to the clients, which demands high transmission capacities. Furthermore, the round-trip-time (RTT) in the communication between the clients and the server reduces the responsiveness of the VE and leads to a loss of transparency [43], [44], that is, the RTT deteriorates the user experience.

Providing high-quality audio-visual-haptic feedback, enabling responsive and realistic interactions with virtual objects as well as providing consistency between all users in a SVHE with geographically distributed users becomes challenging due to the aforementioned communication impairments. CS-based, P2P-based or even hybrid solutions may be preferred depending on the type of application, the computational complexity and the network conditions [35].

Major contributions and thesis organization

This thesis focuses on SHVEs built on top of the CS-State architecture and the inherent challenges of

1. the required communication resources and
2. the loss of transparency due to the RTT in the communication between the clients and the server.

This is motivated by the two aforementioned advantages of the CS-State architecture, namely consistency and robustness against communication delay. If these challenges are addressed

successfully, the advantages of the CS-State architecture outweigh the increase of responsiveness in the P2P architecture. This is particularly true for complex physics-based simulations, which are used, for example, to calculate organ deformations in virtual surgery simulations, but are difficult to synchronize and to keep consistent in the P2P architecture [35]. Additionally, computationally demanding simulations per client are avoided. A cloud-based physics simulation server can be used instead, entailing the general advantages of cloud-based services like scalability, flexibility or availability [45].

Addressing these challenges is also important to facilitate the implementation of SHVEs on top of packet-switched networks. The communication load increases with the number of users connected to the server and the transmission of object state updates quickly becomes a bottleneck in the system. The high packet rates may lead to congestion and dropped packets in the involved network nodes.

The main contributions of this dissertation can be summarized as follows.

1. **Perception-based data reduction in CS-State-based SHVEs (Chapter 3)**
   In contrast to the state-of-the-art for the communication of object states in distributed virtual applications, the presented work takes a human-centric approach and explicitly considers the limitations of human perception to avoid the transmission of unnecessary data.

   - **Data reduction for the interaction with movable rigid objects**
     Section 3.2 presents a data reduction scheme that reduces the number of object state update packets transmitted from the centralized server to the clients. It is based on the local prediction of positions and rotations and considers individual error thresholds for the visual and the haptic modality to trigger new state updates. The haptic error thresholds are evaluated rigorously with subjective tests to find the best-possible trade-off between perceptual robustness and packet rate reduction.
   - **Data reduction for the interaction with deformable objects**
     Lower packet rates, but higher payload characterize the transmission of the state of deformable objects in the CS-State architecture when compared to the interaction with movable rigid objects. The low-delay lossy compression scheme presented in Section 3.3 uses vertex displacement prediction, error thresholds, quantization and entropy coding to compress 3D polygon mesh deformation data. It also considers the haptic and visual modalities individually to achieve the best-possible trade-off between bit rate and rendering fidelity.

2. **Compensating the effect of communication delay (Chapter 4)**
   Communication delay leads to a loss of transparency in the CS-State architecture. The reason is the time needed until the local copies of the VE at the clients are updated. This thesis systematically analyzes the transparency of the CS-State architecture in the presence of communication delay.

- **Interaction with movable rigid objects**
  Section 4.2 presents a linear time-invariant (LTI) model of the CS-State architecture, which represents two-user interactions with a shared movable rigid object. This model is used to analyze the transparency of the SHVE. The analysis reveals guidelines on the tolerable communication delay. If this delay is exceeded, the increased force magnitude becomes haptically perceivable. An adaptive force rendering scheme is proposed, which dynamically changes the parameters of the local force rendering at the clients to hide the effect of delay from the users. The experimental results, including a subjective user study, verify the applicability of the proposed scheme to compensate the effect of constant and time-varying communication delay in the tested delay range of up to 150 ms.
- **Interaction with deformable objects**
  Analogously, Section 4.3 presents a LTI model of the CS-State architecture, which represents two-user interactions with deformable objects. It is used to derive the tolerable communication delay and a scheme to compensate for the loss of transparency during the interaction with deformable objects. The conducted objective and subjective evaluations show that the proposed method successfully compensates for the effect of communication delay in the tested delay range of up to 100 ms.

The outline of this thesis is illustrated in Figure 1.2. Chapter 2 introduces important scientific background and related work. It discusses work in the areas of human haptic perception (Section 2.1), haptic interface design and control (Section 2.2), rendering of virtual environments with haptic feedback (Section 2.3), and Shared Haptic Virtual Environments with a special focus on communication (Section 2.4).

Chapter 3 deals with the perception-based data reduction for the interaction with movable rigid and deformable objects in the CS-State architecture. Section 3.1 first characterizes the traffic in the CS-State architecture. Section 3.2 and Section 3.3 introduce and evaluate the two novel data reduction schemes.

Chapter 4 analyzes the effect of communication delay on the transparency of the CS-State

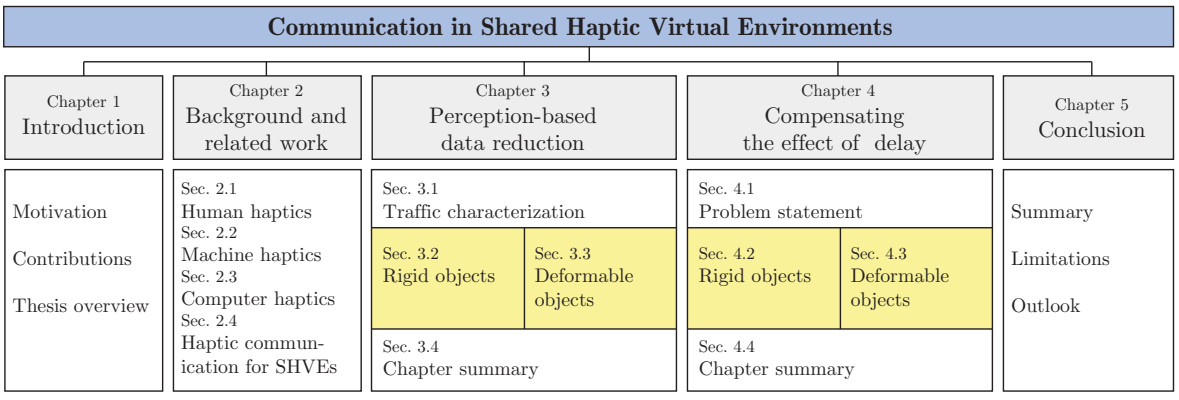


Figure 1.2: Organization of this thesis. Sections with original contributions are highlighted in yellow.

architecture. First, Section 4.1 details on the effect of delay on the force magnitudes at the clients. Section 4.2 and Section 4.3 analyze the haptic transparency, derive tolerable delay bounds and present schemes to compensate the effect of communication delay with movable rigid and deformable objects, respectively.

Chapter 5 concludes with a summary of the results, limitations of the presented work and a discussion about directions for future work.

Parts of the work presented in this thesis have been published in international peer-reviewed scientific journals [1]–[7] and conferences [8]–[20].

Chapter 2

Background and related work

This chapter presents relevant background and related work in the area of haptics, which can be broadly categorized, following the taxonomy introduced by Srinivasan *et al.* [22], into the areas of *human haptics* (Section 2.1), *machine haptics* (Section 2.2), *computer haptics* (Section 2.3) and *haptic communication* with a special focus on Shared Haptic Virtual Environments (Section 2.4).

Human haptics refers to research about the human haptic sensory and manipulation system. Machine haptics concentrates on the design, construction, and control of devices to artificially recreate or augment the sense of touch by displaying various stimuli to the human body. Computer haptics refers to the rendering of contact forces and vibrations in computer-simulated applications. Haptic communication refers to research about the communication of audio-visual-haptic information, which is required to facilitate distributed multimedia systems enhanced with haptic feedback.

This thesis focuses on human-centric communication in Shared Haptic Virtual Environments, that is, it falls within the latter category of haptic communication. To understand the challenges in the implementation of SHVEs, a holistic view of all four interdisciplinary areas is required.

2.1 Human haptics: the human haptic sensory system

The haptic sensory system provides us with a number of cues related to different object properties like, for example, weight, softness, roughness, stickiness, shape, or size during the interaction with objects in our daily environment. Mechanoreceptors in the skin, joints, limbs and muscles relay sensory information to the central nervous system (CNS) [46]. These neural signals are combined with the information from other modalities, including vision and audition, into an overall percept [47].

Additionally, the human motor control system uses these sensory signals to ensure appropriate hand movements, for example, to enable stable grasps of objects or to perform complex interactions [46]. This emphasizes the importance of haptic feedback in applications that mimic real world interactions, like, for example, virtual surgery simulation [48].

Two complimentary subsystems, namely the cutaneous sense and the kinesthetic sense, contribute to haptic percepts. Cutaneous inputs from mechanoreceptors in the skin, also

referred to as tactile inputs, facilitate the perception of tactile surface properties, e.g., roughness, by sensing skin stretch, vibration and deformation [49]. Kinesthetic inputs from mechanoreceptors in muscles, tendons and joints contribute to the perception of limb position/movement and the associated forces. This facilitates the perception of mechanical object properties such as, for example, weight and softness [50]. Most of our everyday interactions include both cutaneous and kinesthetic cues [51].

The distinction between the cutaneous and the kinesthetic modalities is important, because technology to artificially recreate the sense of touch requires hardware and software, which are adapted specifically to the target modality [52].

This thesis considers virtual environments, which are enhanced with kinesthetic force feedback mediated through specialized haptic devices (see Section 2.2). The deployed force rendering algorithms render linear virtual springs with a defined stiffness (see Section 2.3.4). For these reasons, the perception of force and stiffness, including the perceptual limitations, are briefly introduced in the following.

2.1.1 Perception of force

Kinesthetic perception, also referred to as proprioception [53], comprises the perception of limb position and movement along with the applied forces. The primary and secondary spindle receptors, which are mechanoreceptors lying in parallel to the muscle fibers, respond to changes in muscle length. The primary spindle receptors are much more sensitive to the velocity and acceleration of muscle contractions than the secondary spindle receptors. For this reason, they are assumed to signal the velocity and direction of muscle stretch or limb movements to the CNS, whereas the secondary spindle receptors signal the information about their static length or position [50].

Golgi tendon organs, which lie in series with the muscle fibers, are the third mechanoreceptors found in muscles. They sense intramuscular force and provide the CNS with information about the forces produced by muscles. Additionally, the brain integrates information about the body pose to perceive forces [50].

2.1.2 Perception of stiffness

The kinesthetic sensory information is also used to estimate other mechanical properties such as stiffness, viscosity and inertia [50]. The haptic perception of stiffness (often denoted by its inverse, compliance) involves the integration of information from several different cues [54].

A distinction has to be made between the interaction with objects having deformable surfaces, for example, a rubber pad, and objects having rigid surfaces, for example, a metal plate mounted on a spring. Srinivasan *et al.* [55] found that cutaneous information derived from the contact area and pressure on the fingerpad is sufficient to discriminate between the compliance of different rubber specimens with deformable surfaces.

Kinesthetic information, however, is required to discriminate between the stiffness of objects with rigid surfaces. Researchers agree that humans perceive the compliance of such objects based on sensory signals conveying force and movement information [50], [55]–[57].

The kinesthetic and visual sensory modalities provide movement and displacement data at any time during the interaction with the object, but not all of this information is actually processed by the perception system [57].

There is no consensus about the most important piece of information integrated into a stiffness percept. Perceived stiffness seems to be related to the maximum force and/or the mechanical work performed during the interaction [57]–[60]. Pressman *et al.* [60] tested different perception models and found that the model which divides the maximum force by the amount of penetration best explained the stiffness the subjects perceived during the blind interaction with objects. Di Luca *et al.* [57] emphasized on the relevance of the mechanical work cue, but concluded from their experiments that neither mechanical work nor maximum force alone fully explain the perceived stiffness in visual-haptic applications. Instead, they proposed that the perceptual system weighs visual and kinesthetic information during loading and unloading differently to obtain a stiffness estimate.

2.1.3 Limitations of the human kinesthetic perception system

Human haptic perception has limitations in terms of detecting and discriminating certain stimuli. The *absolute threshold* denotes the smallest amount of stimulus energy necessary to produce a detectable sensation [61].

The *difference threshold* is defined as the stimulus change required to produce a Just Noticable Difference (JND) compared to an initial stimulus. The German physiologist Ernst Heinrich Weber was one of the first to systematically investigate difference thresholds of the human perception system. He found that the difference threshold is a linear function of the stimulus intensity itself [62]. This relationship is known as *Weber's Law*:

$$\Delta I = cI \qquad \text{or:} \qquad \frac{\Delta I}{I} = c = \text{const.} \tag{2.1}$$

Weber's Law states that the just perceivable difference between two stimuli (ΔI) is proportional to the initial stimulus (I) itself. The constant fraction c is called the Weber-fraction (WF). It was confirmed experimentally that Weber's Law holds for a variety of different haptic stimuli, and over a wide range of stimulus intensities. The WF, however, increases for very low stimuli intensities [56], [61], [63], [64].

The knowledge of perceptual limitations is important for the design of hardware, software and algorithms to sense, reconstruct, display, and communicate haptic signals. This thesis takes a human-centric approach, that is, the data reduction schemes and algorithms introduced in Chapter 3 and Chapter 4 explicitly incorporate the knowledge of perceptual limitations to relax technical constraints in the SHVE. For this reason, the limitations of human force and stiffness perception are briefly discussed in the following.

2.1.3.1 Limitations of human force perception

The force difference threshold was found to be quite consistent for different muscle groups. Various studies indicate an average WF of around 7 % to 10 % over a force amplitude range of 0.5 N to 200 N [50] with a tendency to increase for force amplitudes below 5 N [63]–[65].

Reference	Variable	Weber-fraction	Experimental condition
Weber [62]	weight	10 %	arm/forearm, lifting 32 oz (0.9 kg)
Ross *et al.* [63]	weight	8 % to 12 %	arm/forearm, lifting 50g, 200g and 400g
Jones [66]	amplitude	7 %	arm/forearm
Pang *et al.* [67]	amplitude	7 %	two-finger pinching (2.5 N, 5 N, 10 N)
Tan *et al.* [59]	amplitude	6 %	two-finger pinching
Allin *et al.* [68]	amplitude	10 %	single finger (forces < 4N)
Höver *et al.* [64]	amplitude	10 %, 14 %, 17 %, 22 %	arm/forearm, (2.5 N, 1.0 N, 0.5 N, 0.3 N)
Dorjgotov *et al.* [65]	amplitude	13 %	arm/forearm (1.5 N, 3.5 N)

Table 2.1: Overview of relevant studies that investigate the human force amplitude difference threshold. The range of the Weber-fraction is between 6 % and 13 % with a tendency towards the upper end for forces between 1 N and 5 N [63]–[65]. The WF strongly increases for very small force amplitudes below 1 N [64].

Höver *et al.* [64], for example, studied the difference threshold for small force amplitudes and found average Weber-fractions of 10 %, 14 %, 17 % and 22 % for the force amplitudes of 2.5 N, 1.0 N, 0.5 N and 0.3 N. The maximum displayable force magnitude of the haptic device used throughout this thesis is 3.5 N and falls within in this range. Table 2.1 summarizes results of fundamental research about force difference thresholds, together with results of studies that closely resemble the hardware setup used in this thesis.

Interestingly, several studies (e.g., [65], [69], [70]) found strong evidence for the anisotropy of human force amplitude perception, that is, the WF depends on the direction of the reference force. The force magnitude estimation experiments of van Beek *et al.* [70] revealed that subjects perceived a force exerted perpendicularly to the arm in the horizontal plane as larger compared to the same force being exerted along the arm. The anisotropy was best-described with an elliptical pattern with its minor axis pointing from the shoulder to the hand.

Tan *et al.* [71] studied the force-direction difference threshold for forces applied on the index finger and found an average threshold of 33°. They also found that this threshold does not depend on the direction of the reference force, that is, the perception of force-direction seems to be isotropic, although human haptic perception is generally assumed to be anisotropic [71]. Similar results have been reported in [72], where the influence of visual feedback has been considered additionally. The isotropic difference threshold reduced from 25.6° to 18.4° if congruent visual feedback was provided to the subjects. In comparison, the constant force-direction threshold of around 1.85° reported in [69] is much smaller. A direct comparison between those results, however, is not possible because the WFs were obtained for different parts of the body and under different experimental setups and conditions. The authors of [70] studied the perception of force directions for forces applied on the arm. Their results did not confirm that the previously discussed isotropy of the index finger also holds for forces applied on the hand/forearm. Instead, they found distorted perception patterns, which varied between subjects. The force-direction threshold, however, was found to be independent of the force magnitude.

Reference	Weber-fraction	Experimental condition
Jones *et al.* [56]	23 % (670 N/m to 6260 N/m)	rigid surface, pressing with hand/arm
Tan *et al.* [59]	22 %	rigid surface, two-finger pinching
Bergmann Tiest *et al.* [54]	15 %	rigid and deformable, pressing with single finger
Varadharajan *et al.* [73]	14.2 %	rigid surface, pressing with haptic device
Koçak *et al.* [74]	13 %	rigid surface, pressing with haptic device
Paggetti *et al.* [75]	15 % (135 N/m to 390 N/m) 28 % (50 N/m)	rigid surface, pressing and pinching with haptic device

Table 2.2: Overview of relevant studies that investigate the human capability to discriminate stiffness. The range of the Weber-fraction lies between 13 % and 28 %. The human perception of stiffness is less accurate as compared to the perception of force (compare Table 2.1). The sensitivity decreases for small stiffness values [56], [75].

2.1.3.2 Limitations of human stiffness perception

The difference threshold for stiffness is considerably higher than for force [50]. Studies indicate a WF of 13 % to 28 % as summarized in Table 2.2.

Jones *et al.* [56] found an average threshold of 23 % in a stiffness matching experiment. Subjects had to press on a electromagnetic motor with computer-controlled reference stiffness with one arm, while adjusting the stiffness of a second motor with the second arm to match it to the reference stiffness. The experiments revealed a WF between 19 % and 28 % for stiffness values between 670 N/m to 6260 N/m. Similar to the perception of forces, the found WF increased for smaller stiffness values.

The studies in [73]–[75] used haptic devices to interact with simulated compliant objects with rigid surfaces. The WF was found to be around 13 % to 15 %, but significantly increased for stiffness values below 100 N/m.

Note that the differences in the experimental methodologies, hardware and stimuli makes it hard to directly compare these results. The studies summarized in Table 2.2 suggest a minimum stiffness difference threshold of about 14 % if the human uses a haptic device to interact with a simulated compliant object with a rigid surface. This kind of interaction and hardware setup closely resembles the setup used in this thesis.

2.2 Machine haptics: haptic device and interface design

A haptic interface consists of hardware and software to generate computer-controlled mechanical signals that stimulate the human kinesthetic and/or cutaneous perception system [52]. Haptic devices, that is, the hardware to create these stimuli, are categorized according to the modality they are targeting at. Vibrotactile devices, surveyed in [76], recreate cutaneous sensations, for example, by creating high-frequency low-amplitude vibrations with voice-coil actuators [77]. Kinesthetic devices, on the other hand, are designed to display low-frequency force signals [52].

Figure 2.1: A kinesthetic haptic device (Geomagic Touch [79]) to interact with a virtual environment.

In the context of this thesis, only kinesthetic force feedback devices are of interest. The term "haptic device" denotes such kinesthetic devices in the remainder. Haptic devices (Section 2.2.1) and the stability of a haptic interface (Section 2.2.2) are briefly introduced in the following.

2.2.1 Haptic devices for kinesthetic force feedback

Impedance-based kinesthetic haptic devices sense the motion (position, orientation and the corresponding velocities) of a tool held by the user and simultaneously display kinesthetic force/torque feedback signals. This bidirectional exchange of information is the most distinguishing feature of haptic devices, when compared to other human-machine interfaces such as a computer mouse [26], [78].

The number of degrees-of-freedom (DOF) in the sensing and actuation capabilities characterizes haptic devices. Other device characteristics are the physical inertia, friction and damping of the device, the size and shape of the workspace, and the dynamic range of displayable stiffness and forces, which is limited by the mechanical design of the device. The device used throughout this thesis is shown in Figure 2.1. It senses the 3D position and orientation of a stylus, that is, it has 6-DOF input, and displays 3D forces, that is, it has 3-DOF output. Hence, it supports only single-point interactions due to the lacking torque display capabilities.

2.2.2 Stability of a haptic interface

Haptic devices are robotic systems, which couple the human user with a virtual environment and exchange energy bidirectionally [26]. In other words, the haptic device closes a control loop between the user and the simulated VE. Computer-based simulations are generally time-discrete. The inherent sampling, computational delay, together with sensor quantization effects in the haptic device and its imperfect electromechanical components afflict the stability of the haptic interface [80]–[85]. Simulating contact with a rigid object, for example, a rigid wall, is especially challenging, because it implies high-gain feedback loops, that is, large stiffness values in the force rendering (see Section 2.3.4.1).

Guaranteeing stability requires careful design of the coupling between the hardware and the software. A frequently used tool for the analysis of stability problems in control in gen-

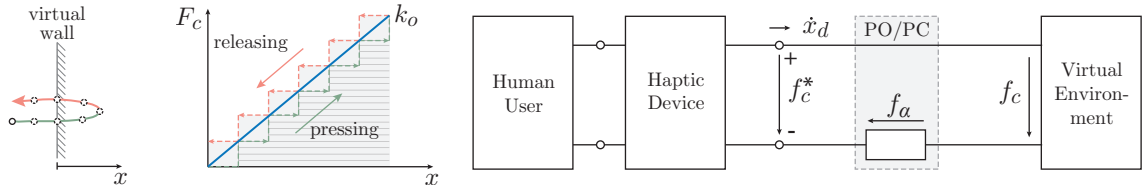


Figure 2.2: (a) Penetrating a virtual wall. The user presses (green) and releases (red). (b) Illustration of the energy gain due to sampling. The solid line denotes the ideal time-continuous force. The shaded area below the position-force-plot denotes the energy injected into the system while pressing on the spring. The gray-colored area denotes the energy extracted from the system while releasing the spring. The system is active, as more energy is extracted from the system than injected. (c) The passivity observer (PO) measures the energy flow at the one-port network connected to the haptic device (the virtual environment). If necessary, the passivity controller (PC) adapts the damping force f_α to dissipate the required amount of energy and to ensure a passive coupling between the haptic device and the virtual environment. ((b) reproduced from [88] © 2015 IEEE, (c) reproduced from [89] © 2002 IEEE)

eral, and specifically of haptic interfaces, is passivity [81], [82]. A system is passive if the supplied energy is greater or equal to the stored energy at all times, that is, it consumes more energy than it creates. Passivity is a sufficient condition for stability, applies to linear and nonlinear systems, and has the beneficial property that interconnected passive systems are also passive [86]. Passivity is described mathematically by

$$\int_0^t f(\tau)\dot{x}(\tau)d\tau + E(0) \geq 0 \quad \forall t \geq 0 \tag{2.2}$$

where the force f and the velocity $\dot{x}$ are the conjugate power variables, which describe the power flow into the system, and $E(0)$ is the initial energy stored in the system at time $t = 0$ [84], [87]. The signs of f and $\dot{x}$ are defined so that their product is positive if energy flows into the system.

The sampled-data implementation of the haptic virtual environment leads to an energy gain and non-passive behavior, called energy leak in [83], as illustrated in Figure 2.2b. Colgate *et al.* have derived a passivity condition for the interaction with a simulated rigid virtual wall [81], [82]. A 1-DOF haptic device, modeled as a mass-damper system with mass m_d and damping b_d, displays force feedback, calculated by a linear spring with stiffness k_o (see Section 2.3.4) at a rate of $1/T_s$. The condition for passivity of this sampled-data system is [81], [82]:

$$b_d > \frac{k_o T_s}{2} \tag{2.3}$$

This result demonstrates a fundamental relation between the design of a haptic device and the design of a haptic virtual environment. It can be considered as an upper stiffness bound, below which the aforementioned energy gain is dissipated by the mechanical damping of the haptic device. Low friction and damping (i.e., small values of b_d) are desired properties for the mechanical design of a high-quality kinesthetic haptic device [52], [78]. As a consequence, high sampling rates (i.e., small values of T_s) are required. The displayable stiffness k_o increases proportionally with the sampling rate $1/T_s$. For these reasons, a sampling rate

(and simulation rate) of around 1 kHz is widely considered as a prerequisite for a stable haptic interface and haptic simulation [90]–[93].

It is also known that virtual damping, besides the mechanical damping of the haptic device, contributes to system stability [94]. Control schemes such as the ones proposed in [81], [82], [89], [95], [96] add an energy-dissipating damping element to the simulation to achieve passivity if the aforementioned criterion cannot be fulfilled. Colgate *et al.* have derived the parameters of a fixed spring-damper system, called virtual coupling, which ensures passivity in the force feedback calculation [81], [82]. For low-frequency inputs (less than half of the Nyquist-frequency $\omega_n = \pi/T_s$), condition (2.3) is extended for a spring-damper virtual wall (k_o, b_o) to

$$b_d + b_o > \frac{k_o T_s}{2} \tag{2.4}$$

where $b_o > 0$ denotes the additional damping element in the virtual environment. The fixed damping parameter b_o has to be selected to guarantee passivity under all operating conditions [89] and, hence, may be chosen too conservatively.

The authors of [89], [95], [96] have introduced the time-domain passivity control scheme. It monitors the energy flow in and out of the system and uses an adaptive damping element to dissipate only the necessary amount of energy to maintain passivity. Figure 2.2c illustrates this concept. The stability of the overall system is guaranteed, because the human user is passive for the frequencies of interest [89], [96], [97] and the cascade of passive subsystems is also passive [86].

The drawback of all passivity-based control schemes is their conservatism. Even if the haptic interface is active, the human operator might not be able to destabilize it [80], [82], [97], because the impedance of the human arm also contributes to system stability by dissipating energy [94]. Adams *et al.* [97] derived methods to design the virtual coupling based on two-port network theory and Llewellyn's stability criteria. This approach is less conservative, but requires a model of the haptic device. Another less conservative approach, which does not require any system parameters and allows a finite amount of energy to be extracted from the system, has been presented recently in [88], [98].

Equation (2.3) and Equation (2.4) consider stability from a control engineering point-of-view. If there are, for example, decaying force transients during the initial contact with an object, the system is still stable, but the experience is not realistic [99]. Choi *et al.* [100] introduce the definition of perceived instabilities to denote any unrealistic sensations that can not be attributed to the physical properties of the touched object. Hence, even a sampling rate of 1 kHz may not be enough and significantly higher rates may be required to achieve perceived stability for very stiff contacts [99].

2.3 Computer haptics: a haptic virtual application

Computer haptics refers to software and algorithms to synthesize computer-generated kinesthetic and cutaneous feedback for the interaction with simulated virtual objects [101]. It can be seen as the equivalent of computer graphics, which deals with the visual rendering of

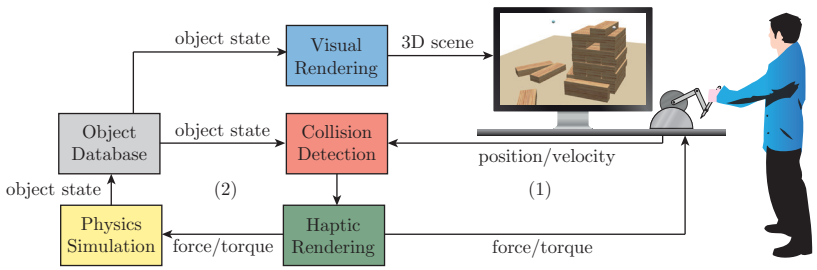


Figure 2.3: Overview of the typical components required to render a haptic virtual environment. Loop (1) closes a control loop between the haptic device and the virtual environment. Loop (2) denotes the physics simulation loop to calculate the virtual objects' displacements or deformations resulting from the user's interaction.

computer-generated virtual scenes. This section introduces the main components required to enhance a VE with kinesthetic haptic feedback.

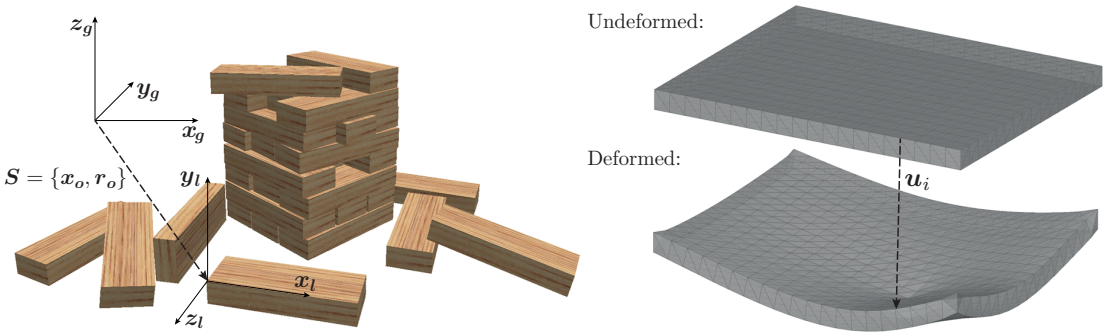
2.3.1 Overview

A haptic virtual environment typically consists of three rendering threads, which run at different temporal update rates (Figure 2.3). The visual thread renders the virtual scene visually at around 30 Hz to 60 Hz. The haptic rendering thread calculates force feedback to enable the touch and feel of virtual objects. The so-called virtual probe [90] represents the haptic device in the VE. Collision detection checks if the virtual probe collides with an object (see Section 2.3.3). If a collision has been found, the force rendering algorithm calculates a corresponding feedback force (see Section 2.3.4). The haptic device, collision detection and force rendering close the loop (1) in Figure 2.3, that is, the control loop between the haptic device and the virtual environment. The stability of this loop typically requires a temporal update rate of 1 kHz or higher (see Section 2.2.2). As a result, collision detection and force rendering need to be performed at this high temporal update rate.

Following Newton's Third Law, the negative of the force displayed to the user is forwarded to the physics simulation, which calculates the virtual objects' displacement and/or deformation resulting from the user's interaction. The temporal update rate of the physics-based simulation thread (loop (2) in Figure 2.3) is favorably equivalent to the update rate of the force rendering loop. This is, however, not always achievable, especially for the computationally expensive physics-based simulation of object deformations [102]–[104].

2.3.2 Physics simulation

The physics-based simulation of the virtual object dynamics is a research area, which originates from the field of computer graphics. It has to be differentiated between the simulation of rigid object dynamics (i.e., the movement of objects with a non-deformable surface) and the simulation of deformable objects (i.e., the deformation of surfaces or volumetric objects) as illustrated in Figure 2.4 The main challenge in the physics-based simulation of virtual object dynamics in HVEs lies in the high temporal update rate required to achieve a stable coupling between the virtual environment and the haptic device, as discussed in Section 2.2.



(a) Rigid objects (b) Deformable object

Figure 2.4: Examples for the application of physics-based simulations. (a) The position $\boldsymbol{x_o}$ and the rotation $\boldsymbol{r_o}$ describe the rigid object state with respect to a global coordinate system $\{x_g, y_g, z_g\}$. (b) A FEM-based simulation calculates the deformation $\boldsymbol{u}_i$ for all vertices of the deformable object.

2.3.2.1 Rigid body physics simulation

The state $\boldsymbol{S} = \{\boldsymbol{x_o}, \boldsymbol{r_o}, \dot{\boldsymbol{x}}_{\boldsymbol{o}}, \dot{\boldsymbol{r}}_{\boldsymbol{o}}\}$ of a rigid object in a three-dimensional space is described by its position $\boldsymbol{x_o}$, rotation $\boldsymbol{r_o}$, velocity $\dot{\boldsymbol{x}}_{\boldsymbol{o}}$ and rotational velocity $\dot{\boldsymbol{r}}_{\boldsymbol{o}}$. A rigid body physics simulation calculates this state by numerically integrating the equations of motions under the consideration of the object's mass, its moment of inertia and forces/torques acting on the object [105]. These forces/torques stem from contacts with other objects, for example, two objects resting on each other [106] or frictional contacts between them [107], and from interactions of the user with the object. A detailed introduction to the simulation of rigid body dynamics is given in [105], [106], [108].

There is a trade-off between speed, that is, the achievable temporal update rate of the simulation, and physical accuracy [107], [109]. Fortunately, even complex rigid body contact dynamics can be simulated sufficiently accurate at the rate required in HVEs [107]. Rigid body physics simulations, however, are prone to simulation instabilities due to the inaccuracy of numerical integration schemes, especially when dealing with multi-object contacts [107]. To ensure stability, artificial damping is added to the simulation [107], [109], [110].

2.3.2.2 Deformable object physics simulation

Applications in the areas of computer animation or interactive virtual mechanical simulation have driven research on a wide variety of different simulation approaches to emulate the deformation behavior of real-world objects in computer applications. There is, however, a conflict between the physical realism and the achievable rendering rate [102].

Virtual surgery simulation is a prominent example, which preferably requires the physically correct simulation of organ deformations. At the same time, the simulation rates need to be high enough to provide an interactive experience to the user [102]. If the simulation is enhanced with haptic feedback, the coupling between the human user, the haptic device and the VE, which necessitates update rates of typically 1 kHz as discussed in Section 2.2.2, further complicates the implementation. Physical realism, low computational complexity and simulation stability are contradicting implementation requirements in this context.

Existing object deformation simulation schemes are broadly categorized into continuum mechanical and heuristic approaches [102]. Among the former, the Finite Element Method (FEM) and, among the latter, mass-spring models are the most prominent. The result of such simulations is a deformed polygon mesh, which is described by the displacement $\boldsymbol{u_i} \in \mathbb{R}^3$ of each vertex i from its undeformed position: $\boldsymbol{S} = \{\boldsymbol{u_1}, \ldots, \boldsymbol{u_N}\}$.

Finite Element Method In continuum mechanics, partial differential equations (PDEs) describe how physical objects and materials deform subject to external forces acting on them. These equations are impossible to solve analytically [111], but can be discretized into a system of ordinary differential equations (ODEs), which are solved numerically [112].

The Finite Element Method discretizes the volume of an object into a finite number of sub-volumes called elements. The simulated 3D object is then represented by a volumetric mesh, consisting of, for example, tetrahedral elements. The displacement of the vertices of this mesh (also called nodes) describe the deformation of the object. The displacement of an arbitrary point within the 3D object is calculated by interpolating the displacements of the nodes of the corresponding element [112]. The discretization results in a system of second-order ODEs [111]:

$$\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{D}\dot{\boldsymbol{u}} + \boldsymbol{k}(\boldsymbol{u}) = \boldsymbol{f} \tag{2.5}$$

The unknown variable of this system is the vector $\boldsymbol{u} \in \mathbb{R}^{3N}$, which describes the 3D displacement of the N vertices of the volumetric mesh. The matrix $\boldsymbol{M} \in \mathbb{R}^{3N \times 3N}$ is the mass matrix, which is constant over time and depends on the object's mass density distribution in the undeformed state. The matrix $\boldsymbol{D} \in \mathbb{R}^{3N \times 3N}$ is the damping matrix and, hence, $\boldsymbol{D}\dot{\boldsymbol{u}}$ describes internal damping forces. The vector $\boldsymbol{k}(\boldsymbol{u}) \in \mathbb{R}^{3N}$ describes internal deformation forces and the vector $\boldsymbol{f} \in \mathbb{R}^{3N}$ denotes external forces acting on the vertices [111], [112].

FEM is considered as physically relatively accurate, but suffers from computational complexity [102], [112]. A commonly used approach to reduce the complexity is to assume a mass distribution concentrated solely at the vertices of the elements (called mass lumping), which leads to a diagonal matrix $\boldsymbol{M}$ [112].

In the linear FEM approach, the internal deformation forces are linearized to further reduce the complexity [111], [113]. The internal deformation force vector $\boldsymbol{k}(\boldsymbol{u})$ is then described by $\boldsymbol{K}\boldsymbol{u}$, where the matrix $\boldsymbol{K} \in \mathbb{R}^{3N \times 3N}$ is constant and can be precomputed during the initialization of the simulation. This leads to a purely linear system of ODEs to be solved at every time step. But still, the FEM-simulation remains computationally expensive due to the high dimensionality of the problem [114]. Obviously, the computational effort increases with N, that is, with the fineness of the discretization.

The FEM-simulation adopted in the experiments presented in Chapter 3 and Chapter 4 uses the co-rotational linear elasticity model [113]. A purely linear FEM simulation is valid only close to the equilibrium configuration. If the object undergoes large rotational deformations, the linearization causes artifacts like unrealistic growth of the volume of the object. The stiffness warping approach employed in the co-rotational linear elasticity model avoids these artifacts, but requires a partial recomputation of $\boldsymbol{K}$ during the simulation [113]. The simulation rate achieved for the relatively simple object shown in Figure 2.4b, which consists

of 714 vertices, 1424 triangles and 1920 tetrahedral elements, was only about 60 Hz[1]. Hence, there is still a big gap to the desired temporal update rate of 1 kHz (see Section 2.2.2).

Modal analysis The modal analysis technique has been proposed originally in [114] for graphical animations. It has been also used to reduce the computational complexity of the FEM simulation in order to match the required haptic update rate [115]–[117].

Modal analysis is based on the aforementioned linear FEM approach. It diagonalizes the matrices $\boldsymbol{M}$, $\boldsymbol{D}$ and $\boldsymbol{K}$ with the whitening transformation (a sort of eigenvalue decomposition), which transforms Equation (2.5) into a set of $3N$ independent (decoupled) ODEs [114]. The explicit solution of each ODE can be obtained as a function of time, which greatly simplifies the real-time simulation. Each ODE describes a single vibration mode of the object, which is associated with a certain resonance frequency. These vibration modes can be thought of as the characteristic vibrations of solid objects, for example, of a string or a membrane. The linear superposition all modes accurately describes the object deformation [112], [114].

The computational complexity is then reduced by considering only the vibration nodes with the lowest natural frequencies during the real-time simulation. This procedure is called modal reduction [115]. High-frequency, small-amplitude modes are discarded as these modes normally have only little effect on the deformed object shape. If more vibration modes are considered, the accuracy increases, but, of course, the computational complexity also increases [112], [114]. The works presented in [115]–[117] have shown that the required haptic simulation rates can be achieved using modal analysis at the cost of reduced realism.

Similar to the aforementioned original linear FEM approach, linear modal analysis struggles with large object deformations [114] due to the underlying linear internal deformation force model. Extensions have been proposed to avoid such artifacts [117], [118].

Mass-spring models Mass-spring models further simplify the deformable object model to gain computation speed. A network of mass points interconnected with linear springs (or spring-damper elements) represents the surface of a deformable object. Typically, the mass points are placed at the vertices and springs are placed along the edges of the surface polygon mesh. This representation results in a system of linear ODEs, which is solved by numerical integration [102].

The main advantages of the mass-spring model compared to FEM are the simple implementation and the achievable simulation rate. The mass-spring model typically represents only the object surface, which already greatly reduces the dimensionality of the system of ODEs as compared to FEM, but comes at the cost of reduced realism. In fact, mass-spring-based deformable objects become unrealistic for large deformations, do not maintain their volume during deformation and tend to oscillate due to their inherent structure [102].

Reproducing real-world deformable objects, for example, for the use in a virtual surgery simulation, is difficult in practice, because the parameter identification (topology, masses and spring stiffnesses) is challenging. Machine learning algorithms have been applied to

[1] The FEM-simulation ran on standard workstation with a quad-core i7 CPU and 8 GB RAM.

learn the parameters from a reference, for example, a physically very precise offline FEM simulation [119], [120].

Pre-computation of deformations Extensive pre-computations of deformations in the simulation of non-linear soft tissue have been used in [121], [122] to speed up the simulation. The pre-computation allows the usage of complex non-linear Finite Element models, which are physically very precise, but impossible to solve at interactive rates.

The approach presented in [122] calculates the force required to displace a vertex from its equilibrium position to a finite set of possible target positions and stores the resulting static mesh deformation. This is repeated for each surface vertex, which leads to a large database of force samples associated with certain vertex displacements and object deformation configurations. During the real-time rendering, the haptic device is coupled to the closest surface vertex once a collision with the object occurs. Then, the device position inside the object is sampled and the corresponding vertex target position, together with the object deformation and the force is retrieved from the database. The object is deformed accordingly and the force is displayed through the haptic device. Interpolation is used to smoothly switch between the retrieved states. This approach achieves the required high temporal haptic update rate as no complex FEM simulation is required during the interaction.

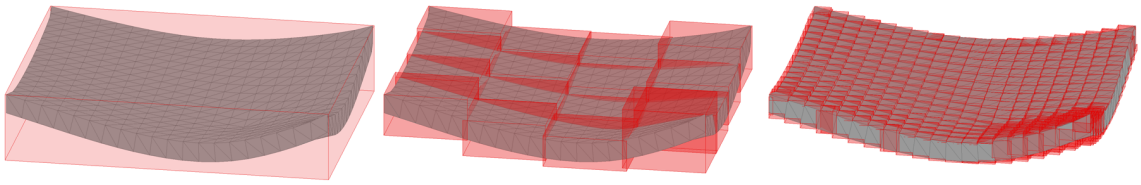
The disadvantage is that it considers only static object deformations and neglects the dynamic properties during the interaction. Also, pre-computing deformations quickly becomes unfeasible for multi-user interactions, as the number of possible user input configurations grows exponentially.

2.3.3 Collision detection

Collision detection is highly relevant for many applications in the area of computer graphics and has been researched for many years. Comprehensive surveys about the detection of collisions between 3D objects can be found in [123], [124] and, with a special focus on deformable objects, in [125], [126].

Collision detection in a HVE is necessary to determine if the haptic device, represented by the virtual probe, collides with any other virtual object. The work presented within this thesis uses a single-point haptic device, that is, the device is represented with a single virtual sphere. If no collision is found, the device is in free space and no force feedback is displayed to the user. If a collision is found, the collision information (e.g., the closest triangle and the contact point on this triangle) is forwarded to the force rendering algorithm (see Section 2.3.4). The high update rates of the haptic rendering complicate collision detection as compared to computer graphics applications, which generally require update rates of only 30 Hz to 60 Hz.

Prominent collision detection algorithms exploit hierarchical bounding volumes, which are proven to be among the most efficient data structures for collision detection [126]. They recursively subdivide the virtual space, typically until each primitive of the polygon mesh is encapsulated by a single bounding volume. Spheres, boxes or discrete-oriented polytopes have been used to implement hierarchical bounding volumes [125], [126].



(a) Root node (b) Intermediate nodes at level 4 (c) Leaf nodes

Figure 2.5: Exemplary nodes of an AABB tree constructed for a deformable polygon mesh.

Figure 2.5 shows the *Axis-Aligned Bounding Boxes* (AABBs) [127] of a deformed polygon mesh. The AABBs are stored in a tree (a so-called AABB tree in this case) to allow for efficient collision detection [127]. The boxes are called axis-aligned, because they are aligned to the local coordinate system of the virtual object.

The tree is traversed from the rough object representation, the root node, down to the leaf nodes (called the broad phase). The bounding boxes allow for a fast check if a query point lies inside their volume. If any check during this traversal is negative, the collision search is canceled. Only if a leaf node is reached, the computationally more complex procedure, which checks if the query point really lies within the object volume, has to be invoked (called the narrow phase).

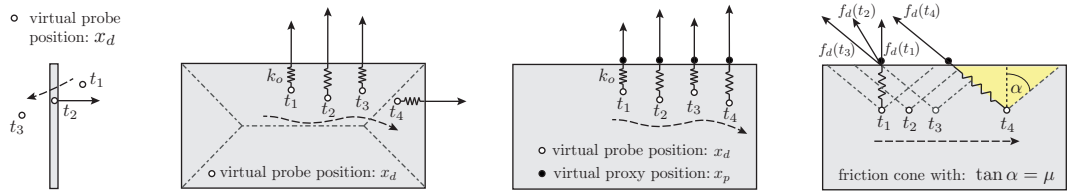
The AABBs and the AABB tree are typically precomputed to allow for a fast collision detection. Deformable objects, however, require the continuous update (refitting) of the bounding volumes while the object shape is deformed [125]–[127]. Such a refitting process is about ten-times faster than a complete re-initialization [127], but the update for an object containing many primitives is still challenging in the context of HVEs due to the required high rendering rates.

AABB trees are generally assumed to yield a worse performance for rigid objects compared to oriented bounding box trees (OBB trees) [127], [128], but the update procedure for deformable objects is considerably faster [127]. For this reason, the collision detection algorithm implemented in this thesis relies on AABB trees, which are updated at runtime as proposed in [127].

2.3.4 Haptic rendering

Once a contact between the haptic device and a virtual object has been found, the haptic rendering algorithm calculates the haptic response, which is displayed to the user through the haptic device. An introduction to haptic rendering can be found in [22], [92], [93].

Haptic rendering algorithms are categorized according to their target modality, that is, there are kinesthetic (e.g., [90], [129], [130]) and vibrotactile (e.g., [131], [132]) algorithms. The HVEs investigated in this thesis employ point-based kinesthetic force feedback. Hence, special focus is put on algorithms to calculate point-based force feedback for the interaction with rigid and deformable objects in the following. Furthermore, the focus is on objects that are represented by polygon meshes. Other algorithms consider, for example, surfaces that are defined by implicit functions [130] or point clouds obtained from depth cameras [133], [134].



(a) Thin object (b) Direct rendering (c) Proxy algorithm (d) Friction cone

Figure 2.6: Haptic rendering of interactions with rigid objects. (a) If the history of the device trajectory is not considered, pop through artifacts occur during the interaction with thin objects (time step $t_2 \rightarrow t_3$). (b) For sharp edges, force direction jumps occur ($t_3 \rightarrow t_4$) if the force rendering algorithm always considers the closest surface triangle. (c) The virtual proxy helps to avoid these artifacts. (d) The friction cone extends the original proxy algorithm to model frictional surfaces.

2.3.4.1 Haptic rendering of interaction with rigid objects

The term "rigid object" implies the rendering of a spring with an infinitely large stiffness. This is not possible in practice for two reasons. First, the stiffness a haptic device can display mechanically is limited (see Section 2.2.1). Second, the stiffness a haptic interface can display stably is limited and depends on the sampling rate and the mechanical and virtual damping in the system (see Section 2.2.2).

Hence, the penetration of the virtual probe into the virtual object is unavoidable. Haptic rendering algorithms that are based on this penetration are called penalty methods [90]. A straightforward approach, which uses the penetration distance between the virtual probe and the closest point on the object surface to calculate force feedback, struggles with thin objects and edges of polygon meshes. For thin objects, the virtual probe might pop through the object as illustrated in Figure 2.6a. For objects with sharp corners as shown in Figure 2.6b, the closest surface triangle might change instantaneously, which leads to unwanted jumps in the force amplitude and direction [92].

Proxy algorithm The constraint-based force rendering algorithms in [90], [129] address these limitations by adding the so-called virtual proxy to the virtual environment. In free space, the virtual probe and the virtual proxy are co-located. During the contact with an object, the virtual proxy is always constrained to stay on the polygon surface (Figure 2.6c). In each iteration, the algorithm tries to move the proxy from its last position towards the probe and checks if this path is blocked by an object. If any interfering primitive was found, the proxy is moved until it touches the surface of this primitive [90]. Hence, the proxy algorithm considers the device trajectory history to avoid the aforementioned artifacts.

The virtual proxy serves two purposes. First, it serves as a visual reference to hide the virtual probe penetration from the user. Second, it allows the implementation of the spring-based (or spring-damper-based) virtual coupling (see Section 2.2.2) between the probe and the proxy. The force feedback $\boldsymbol{f_d} \in \mathbb{R}^3$ is calculated with the position of the haptic device (virtual probe) $\boldsymbol{x_d} \in \mathbb{R}^3$ and the virtual proxy $\boldsymbol{x_p} \in \mathbb{R}^3$ for a spring-based virtual coupling as [129]

$$\boldsymbol{f_d} = k_o(\boldsymbol{x_p} - \boldsymbol{x_d}), \quad \text{with } k_o > 0 \tag{2.6}$$

If a virtual damping element is implemented, for example, to ensure stability (see Section 2.2.2), the spring-damper virtual coupling is calculated as

$$\boldsymbol{f_d} = k_o(\boldsymbol{x_p} - \boldsymbol{x_d}) + b_o(\dot{\boldsymbol{x}}_{\boldsymbol{p}} - \dot{\boldsymbol{x}}_{\boldsymbol{d}}), \text{ with } k_o > 0,\ b_o \geq 0 \tag{2.7}$$

The proxy algorithm commonly runs at a rate of 1 kHz or higher, even for complex geometric objects [90], because the proxy update procedure as well as collision detection for the point-based interaction with rigid objects (see Section 2.3.3) can be performed at the required high update rates. A generalization of the proxy algorithm to 6-DOF interactions between rigid objects with complex geometry is presented in [135].

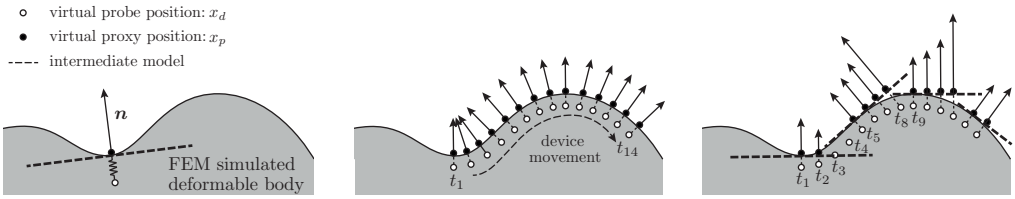
Friction rendering The proxy algorithm illustrated in Figure 2.6c leads to forces, which are aligned with the surface normal direction, that is, it renders frictionless surfaces. The friction cone algorithm extends the proxy algorithm to efficiently render coulomb friction effects on the surface of a polygon mesh [136]. The Coulomb friction coefficient μ and the applied normal force (the penetration depth) define a 3D friction cone, which is illustrated in Figure 2.6d in 2D. The intersection of the cone with the surface defines a friction circle. The virtual proxy is not updated if it is currently within this friction circle and the device is moved (static friction). Once the virtual proxy leaves the friction circle, it is always updated to stay on the circumference of the friction circle (dynamic friction).

2.3.4.2 Haptic rendering of interaction with deformable objects

The computational demand of the deformable object simulation (see Section 2.3.2) and the required collision tree update (see Section 2.3.3) complicate the rendering of force feedback for the interaction with deformable objects, because the stability of the coupling between the haptic device and the VE requires update rates of commonly around 1 kHz. Two different strategies have been employed, which are briefly introduced in the following.

Single-rate rendering The computation of force and object deformation is tightly coupled in physics-based deformation simulations (see Equation (2.5)). This motivates the single-rate rendering approach, where the force displayed to the user is extracted directly from the deformation simulation [92]. Fur this purpose, the temporal update rate of the deformation simulation rate has to match the high haptic rendering update rate. This has been achieved, for example, by applying modal analysis on a linear FEM model [115], [117], by using simplified mass-spring models simulated on a Graphics Processing Unit (GPU) [137], or parallel processing methods [138].

Multi-rate rendering The multi-rate rendering approach, originally proposed in [139] and employed, for example, in [104], [140]–[144], separates the deformation simulation and the haptic rendering into different threads running at different temporal update rates. The key idea is that the force rendering uses a simplified intermediate model to achieve the required high temporal update rate.



(a) Planar intermediate model (b) Desired force feedback (c) Force artifacts

Figure 2.7: Multi-rate haptic rendering of the interaction with a deformable object with a planar intermediate model. (a) The planar model is updated at a low temporal rate, together with the FEM simulation. (b) The hypothetical force rendered for the given device trajectory, if the proxy algorithm would use the exact surface representation. (c) The high-rate proxy algorithm uses the planar intermediate model, which is updated at the time steps t_1, t_5 and t_9. This leads to errors in the force direction, force magnitude and also amplitude/direction jumps.

In [140], the low-rate deformation simulation extracts simple geometric shapes, for example a plane or a sphere, to approximate the area around the user interaction as illustrated in Figure 2.7a. A constraint-based force rendering algorithm (see Section 2.3.4) uses the intermediate model to calculate force feedback. The intermediate model is static during one deformation simulation time step. This eventually leads to jumps of the intermediate model once the model is updated and, as a consequence, to distortions in the rendered force feedback. Linear interpolation is commonly applied on the intermediate model position/rotation to smoothly converge from the current to the new state [142]. This multi-rate rendering approach generally supports multi-user interactions and does not depend on the underlying type of deformable object simulation.

If the intermediate model does not accurately approximate the object surface, the low-rate model update leads to errors in the force direction and amplitude [139], [144]. This problem is illustrated in Figure 2.7b and Figure 2.7c. The work presented in Chapter 3 and Chapter 4 also relies on an implementation using intermediate models that are updated at a low temporal rate. Exact cutouts of the original surface mesh are used as intermediate models instead of an approximation with simple local shape models to avoid these artifacts.

The authors of [145], [146] do not rely on a static intermediate model at all and propose to locally approximate a non-linear deformable object with a local intermediate linear mass-spring model, which is simulated at the haptic update rate and re-generated after each simulation step of the global non-linear deformation simulation. The difficulty in this approach lies in the dynamic selection of the local mass-spring model parameters to accurately describe the physical behavior of the deformable object. Similarly, the algorithm in [144] extracts a mechanical contact representation at a low rate, which is used to locally approximate the object dynamics and can be calculated at the required high rate.

2.4 Haptic communication: Shared Haptic Virtual Environments

2.4.1 Applications and challenges

Shared Haptic Virtual Environments enable the collaboration between users, which are geographically distributed and connected over a communication network. This enables applications, like, for example, collaborative surgical training [30]–[32], assembly simulation [33], virtual prototyping [34] or gaming [25], [36]. Several studies have shown that the inclusion of haptic feedback in such applications is beneficial, as it increases the realism [22], leads to better task performance [23], [24] and also increases the sense of togetherness between users [24], [147]. The implementation of SHVEs, however, faces several issues and challenges, which are summarized in the following.

Stability Analogous to the implementation of a standalone haptic virtual environment, which is explained in Section 2.3, a stable coupling between the haptic device and the VE of every user is a key design goal [101]. The data transmission over packet-switched networks introduces delay, which is not negligible for the long-distance communication between geographically distributed users and might lead to stability problems. Unwanted force oscillations and excessive force magnitudes can hamper the user experience, destroy the hardware or even harm the human user.

Fidelity and transparency Achieving fidelity, that is, enabling a high-quality haptic interaction in a SHVE, is another challenge. The IEEE standard for distributed interactive simulations defines fidelity as "the degree to which the representation within a simulation is similar to a real world object" [148].

In the context of bilateral teleoperation systems, where a human operator controls a robot in a remote environment and receives audio-visual-haptic feedback [37], the concept of transparency is often used as a measure of fidelity [149]. In an ideal and transparent teleoperation system, the operator feels as if interacting with the remote environment directly [150]. Common definitions of *ideal transparency* require that the mechanical properties of the environment (real or virtual) are displayed to the user without any distortion [149], that is, ideal transparency requires a match between the mechanical impedance of the remote environment and the locally displayed impedance, or require identical position and force signals at the local and the remote side [151]. Similarly, transparency describes the fidelity of the haptic interaction in a SHVE [152].

The concept of *perceived transparency* [153] alleviates the aforementioned definitions of ideal transparency. If the displayed impedance in a teleoperation system is not distinguishable from the impedance of the remote environment, the system is said to be perceived transparent. Similarly, if there is no perceivable difference between the local (non-delayed) interaction with the VE and the interaction through the distributed system, the SHVE is called as perceived transparent in the remainder.

Consistency As distributed users share the same virtual environment, the application state needs to be synchronized between them. Perfect consistency (also denoted as absolute consistency in [154]) implies that all users can experience the same state, that is, they can see, hear and feel the same virtual environment at all times. Perfect consistency in a distributed application is not achievable in the presence of communication delay [35], [154], [155]. Consistency maintenance schemes aim at providing the best-possible experience to all users and to resolve any temporal and spatial conflicts between them [154].

Communication The high sampling and rendering rates (see Section 2.2 and Section 2.3) lead to a high load on the communication network, which connects the geographically distributed users [156]–[159]. Furthermore, communication impairments such as delay, jitter and packet loss can lead to stability issues [39], [160]–[162], reduced transparency [43], [44], [163], [164] and a loss of consistency [165]–[168].

Computational complexity The simulation of a HVE is computationally demanding due to the high rendering rates and the complexity of physics-based deformation simulations (see Section 2.3). The computational complexity increases with the number of users interacting in the virtual environment, because collision detection and force rendering have to be performed for each user individually.

Scalability The computational complexity is one factor that hampers the scalability of SHVEs, that is, the number of users the SHVE can support. The transmission capacities required to transmit the information between different users is another limiting factor. For applications like, for example, collaborative surgical training, which involves probably only two or three users, scalability is less of an issue compared to multi-player haptic online games, which possibly connect hundreds of users [101].

2.4.2 Communication and control architectures

Different communication and control architectures have been conceived to implement SHVEs and to address the aforementioned challenges [29]. These architectures are categorized into centralized client/server and distributed peer-to-peer architectures. The following sections introduce three architectures, which research in the area of SHVEs has mainly focused on, and compares them with respect to the aforementioned challenges.

2.4.2.1 Position-force client/server architecture (CS-Force architecture)

The position-force client/server architecture (denoted as *CS-Force architecture* in the remainder), investigated for example in [38], [39], [169], relies on a centralized server, which fully handles the simulation of the virtual environment. This approach is conceptually similar to cloud gaming, that is, the rendering of interactive game applications in the cloud [170]. The users send their commands to the cloud-server and receive the rendered scene as a video stream. Similarly, the clients transmit their haptic device position $\boldsymbol{x}_{d,i}$ and/or velocity $\dot{\boldsymbol{x}}_{d,i}$

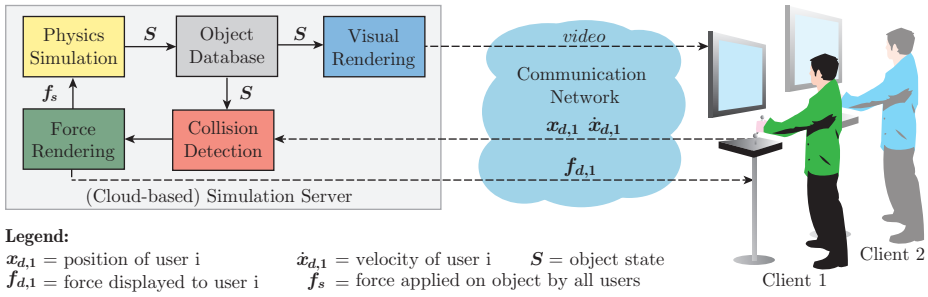


Figure 2.8: Illustration of the position-force client/server architecture (CS-Force architecture). Each client/server connection resembles a tradition bilateral teleoperation system.

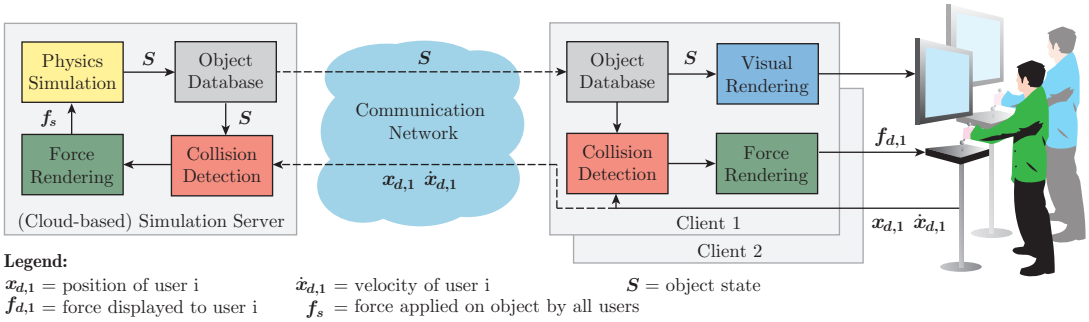


Figure 2.9: Illustration of the position-state client/server architecture (CS-State architecture). The server simulates the object movement and/or deformation by means of a centralized physics simulation and transmits the object state to the clients.

to the server and receive a video stream, together with the rendered haptic force feedback $f_{d,i}$ from the server as illustrated in Figure 2.8.

Each client/server connection resembles a traditional bilateral teleoperation system, where the power variables velocity and force are transmitted over the communication network [37]. The main drawback of this architecture are the instabilities caused by even a small delay in the communication between the client and the server as experienced in [38] and analyzed in [39]. Various control schemes have been proposed to guarantee the stability of bilateral teleoperation systems in the presence of communication delays (e.g., [171]–[173]). It is important to note that such control schemes can be adopted to stabilize the CS-Force architecture, but they generally gain stability at the cost of a loss in transparency [149], [153].

2.4.2.2 Position-state client/server architecture (CS-State architecture)

The aforementioned client/server approach is extended by maintaining a copy of the VE at the clients and executing visual rendering, collision detection and force rendering locally as illustrated in Figure 2.9. In this architecture (denoted as *CS-State architecture* in the remainder), the clients again transmit their haptic device states to the server, where a centralized physics engine simulates the dynamics (movement and/or deformation) of virtual objects. The resulting object states (summarized in S) are returned to the clients to update the local copies of the VE. Force feedback is rendered locally at the required high rate and without delay in the loop between the haptic device and the local VE. This architecture has been

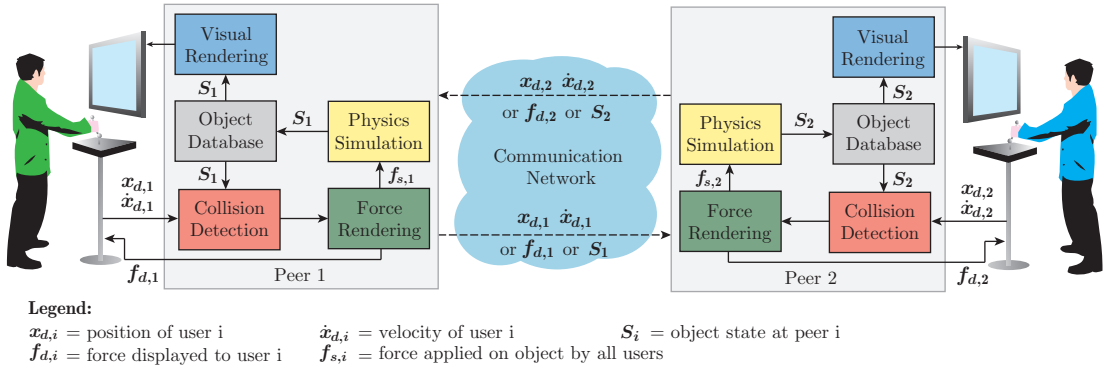


Figure 2.10: Illustration of the peer-to-peer architecture (P2P architecture). The simulation is fully distributed.

studied, for example, in [36], [43], [44], [156]–[158], [163], [164], [174].

The CS-State architecture corresponds to the communication architecture used in traditional online games [175] and is conceptually similar to model-mediated teleoperation (MMT) [7], [176]. In MMT, a model of the environment is estimated at the remote side (the server side) and transmitted to the operator side (the client side) to render force feedback locally, which makes the system robust to communication delays [177]. Different to MMT, the model estimation is not necessary for SHVEs, as the simulated VE inherently includes the environment model.

The VE copies at the clients need to be updated once the simulated object states have changed. It requires a full round-trip-time until the object state, which is calculated only on the centralized server, arrives at the client [33], [165]. Communication delay, hence, reduces the responsiveness of the local VE. This constitutes the main drawback of this architecture.

2.4.2.3 Peer-to-peer (P2P) architecture

The fully-distributed peer-to-peer architecture (denoted as *P2P architecture* in the remainder) relies on local physics simulations at each user, as illustrated in Figure 2.10. Each peer fully simulates the VE locally, which guarantees a high responsiveness.

The haptic device position $\boldsymbol{x_{d,i}}$ is transmitted to the other peers to facilitate the distributed simulation of the object dynamics. Alternatively, the locally rendered force $\boldsymbol{f_{d,i}}$ or even the simulated object state $\boldsymbol{S}$ can be transmitted to reduce the computational complexity at the other peers [178]. In the presence of communication delay, each peer will use outdated information about his or her peers, which leads to a loss of synchronization between the physics simulations at different peers. This constitutes the main drawback of the P2P architecture. Consistency maintenance schemes, which are studied, for example, in [39], [160]–[162], [166], [179], become necessary. Alternatively, the result of the physics simulation, that is, the object state, is transmitted to the other peers, where consistency maintenance schemes resolve conflicting states [33], [168].

	CS-Force	CS-State	P2P
Stability	Instabilities already for several milliseconds of communication delay [39] (e.g., 10 ms lead to instabilities in [38])	Robust due to the local force rendering (e.g., no instabilities are reported for a delay of 60 ms in [43] or 100 ms in [156])	Robust due to the local physics simulation and force rendering
Fidelity and transparency	In the presence of communication delay, stability-ensuring control schemes introduce artifacts and reduce the transparency [149], [153]	Communication delay leads to reduced transparency during the interaction with movable and deformable objects [43], [44]	High fidelity due to the local force rendering and physics simulation [33], [35]
Consistency	The centralized physics simulation ensures a consistent object state [33], [35]	The centralized physics simulation ensures a consistent object state [33], [35]	Communication delay and packet loss lead to a loss of consistency between the peers [33], [35]
Communication load	Video, audio and haptic signals; high packet rate and small payload for haptic data (typically 1000 packets per second for position/velocity and force) [41], [42]	Device pos./vel.: high packet rate, low payload [41], [42]; rigid object state: high packet rate, low payload [157], [180]; deformable object state: lower packet rate, but high payload [181]	Device position, velocity or forces: high packet rate and low payload. Object state (if used): same as for the CS-State architecture
Computational complexity	Complex server simulation; thin clients which display only the received signals	Complex server simulation; higher complexity at the clients when compared to the CS-Force architecture	Requires complex simulation at all peers
Scalability	Server is a bottleneck	Server is a bottleneck	Local simulation becomes the bottleneck

strength/advantage neutral weakness/disadvantage

Table 2.3: Comparison of three architectures, which are commonly used to implement a SHVE.

2.4.2.4 Comparison of the architectures conceived to implement a SHVE

The three aforementioned architectures have all strengths and weaknesses regarding the challenges listed in Section 2.4.1. Table 2.3 summarizes and compares them qualitatively.

Generally, all architectures struggle with communication delay. In the CS-Force architecture, delay leads to instabilities [38], [39]. This problem is alleviated in the CS-State architecture by the local force rendering at the clients. Communication delay in this case reduces the fidelity and leads to a loss of transparency [43], [44]. In the P2P architecture, it impairs the consistency between the different peers [33], [165].

It should be highlighted that the local force rendering in the CS-State and the P2P architecture does not ensure stability, but increases the robustness against communication delay, which is the major source of instabilities in the CS-Force architecture. The local VEs at the clients/peers require updates as soon as they receive (delayed) object state update packets from the server, or consistency maintenance schemes resolve conflicting object states. Such updates can artificially inject energy into the coupling between the haptic device and the

local VE and destabilize it. Control schemes to ensure stability, especially for the two-user interaction with a shared rigid object, have been investigated in [39], [160]–[162], [166], [167]. Communication delay, however, has no effect during the interaction with static objects in the CS-State and the P2P architecture.

All architectures struggle with the amount of communication resources required to transmit the information between the geographically distributed users. Data reduction and traffic control schemes that reduce the required transmission capacities are introduced in Section 2.4.3.

The server-based architectures had been attributed with scalability issues (e.g., in [33], [166], [182]), because the computation and communication load on server increases with the number of users connected to it, and because the server constitutes a single point of failure. The CS-based architectures, however, can benefit from the emerging trend of cloud computing [45]. A powerful cloud-based simulation server can be used instead of individual local physics-based simulations at each user. This entails the general advantages of cloud-based services like scalability, flexibility or availability [45].

Furthermore, a centralized server ensures a consistent object state. Although schemes to maintain consistency in the P2P architecture have been studied extensively (e.g., in [33], [39], [160]–[162], [166], [168], [179]), achieving consistency especially for complex interactive simulations like FEM-based deformable object simulations remains a significant challenge [35], [183], [184]. Existing approaches often simplify the interaction scenario, for example, to a rigid object, which is movable only along one dimension [160]–[162], [166], or to very simplistic deformable objects [168].

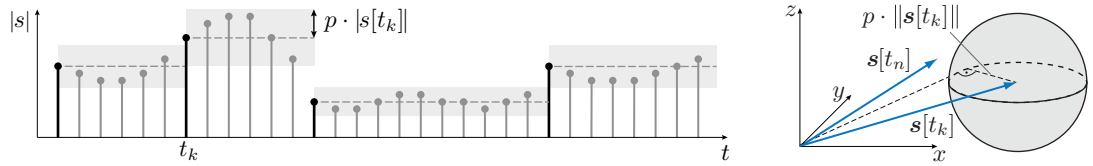
Depending on the type of the application, the computational complexity and the network conditions, either the CS-based, the P2P or even hybrid solutions may be preferred [33], [35]. This thesis studies the CS-State architecture, which ensures consistency but lacks responsiveness in the presence of communication delay, as discussed in more detail in Section 2.4.4.

2.4.3 Data reduction and traffic control

This section reviews data reduction and traffic control schemes that can be employed in SHVEs to reduce the required transmission capacities. Each client/server connection in the CS-Force architecture necessitates the transmission of position/velocity signals from the clients to the server, and force signals vice versa. Hence, they resemble bilateral teleoperation systems [37]. The return channels from the server to the clients differ in the CS-State architecture. The server transmits object state updates instead of forces and/or torques to the clients.

2.4.3.1 Position, velocity and force signals

The communication of position, velocity, force, and torque signals is characterized by high packet rates and low-delay requirements. Blockwise processing and retransmission of data samples is unfeasible because delay can deteriorate the system performance and, hence, the coding delay should be kept as small as possible [42]. For this reason, packets are typically



(a) 1D deadband example (b) 3D spherical deadzone

Figure 2.11: Illustration of the perceptual deadband principle [41]. (a) The perception thresholds, i.e., the boundaries of the gray zones in the 1D example, are a function of the stimulus intensity. Samples that fall within these deadbands can be dropped. (b) Extension of the 1D deadband to a 3D isotropic deadzone (Figures (a), (b) are reproduced and adapted from [41], © 2011 IEEE).

sent immediately once new sensor readings are available [42]. The User Datagram Protocol (UDP) is used in IP-based packet-switched network scenarios to transmit the data samples. This results in a high packet load on the network and also leads to substantial data overhead due to the transmission of packet header information. At a packet rate of 1 kHz, the overhead can even dominate the actual payload data [40]–[42].

Downsampling-based packet rate reduction Shahabi *et al.* [185] have compared different downsampling approaches and also proposed to use predictive coding techniques and quantization to compress the payload of position and force packets. The adaptive sampling approach monitors the acquired signals, identifies the Nyquist sampling rate within a signal window and adapts the sampling rate at the sender accordingly. This adaptive sampling procedure, however, introduces unwanted delay due to the window-based analysis.

Deadband-based packet rate reduction The packet rate reduction approach for networked control systems proposed in [186] relies on a time-domain instead of a frequency-domain signal analysis and does not introduce delay. New signal updates are triggered only if the difference between the most recently sent data sample and the current input data sample exceeds a fixed threshold. This fixed threshold is denoted as the deadband. The receiver holds the value of the most recently received data sample until a new sample is received. A similar sampling strategy is also evaluated in [187].

Perceptual deadband-based packet rate reduction The fixed threshold used in [186] ignores the human operator with its strong limitations in terms of perceivable signal changes (see Section 2.1). Perceptual data reduction schemes, which employ a mathematical model of the human haptic perception to keep the signal distortion below the perception thresholds, have been proposed to account for this [40]–[42], [69], [169], [188]–[194].

The so-called perceptual deadband (PD) data reduction approach exploits Weber's Law (see Section 2.1.3) to reduce the number of force and velocity data packets in bilateral teleoperation systems. The principle of perceptual deadbands is shown in Figure 2.11a for a 1D signal. Samples with black dots represent the output of the data reduction scheme. They define perceptual deadbands which are illustrated as gray zones and describe the human perceptual limitations. Samples that fall within the currently defined deadband (grey sam-

ples) can be dropped because the associated signal change is too small to be perceptible when compared to the currently displayed signal.

The extension of this 1D example to 3D forces/velocities leads to spherically shaped deadzones if an isotropic perception model is assumed (see Figure 2.11b). If the tip of the current sample vector $\boldsymbol{s}[t_n]$ falls within the volume of this deadzone, which has its origin at the tip of the most recently transmitted sample vector $\boldsymbol{s}[t_k]$ ($t_n > t_k > 0$), the current sample vector is not transmitted [40], [41], [190]:

$$p \cdot \|\boldsymbol{s}[t_k]\| \begin{cases} < \|\boldsymbol{s}[t_n] - \boldsymbol{s}[t_k]\| \Rightarrow \text{transmit sample } \boldsymbol{s}[t_n] \\ \geq \|\boldsymbol{s}[t_n] - \boldsymbol{s}[t_k]\| \Rightarrow \text{no transmission} \end{cases} \tag{2.8}$$

The parameter p is denoted as the deadband parameter, which controls the introduced distortion and the packet rate required to transmit the signal. It can be tuned with subjective user tests to find a good trade-off between the introduced signal distortion and the packet rate reduction. For example, the original rate of 1000 packets per second could be reduced by about 90 % without afflicting the performance of the teleoperation system in [40].

The receiver applies a zero-order-hold (ZOH) strategy to upsample the signal to the required rate of the local control loop. As this interpolation can be non-passive, a modified and passive ZOH strategy has been presented in [188]. Signal-based predictive coding, for example, based on a first-order linear predictor, can be used at the sender and receiver to further improve the performance of the PD data reduction scheme [40], [189], [190], [192], [193].

The PD-approach has been refined with velocity-dependent force thresholds [191] and an anisotropic deadzone [69]. This deadzone has the shape of a truncated cone to account for results of perceptual studies, which found a strong evidence for the anisotropy in human kinesthetic perception (see Section 2.1.3) and to further improve the performance. Furthermore, an error-resilient PD scheme to reduce the impact of packet losses has been proposed [195], [196].

In the architectures discussed in Section 2.4.2, the PD-approach can be used to transmit the haptic device velocity information [40], [189], [190], [197] from the clients to the server and between the peers, or to transmit the forces from the server to the clients in the CS-Force architecture.

2.4.3.2 Rigid object state

The CS-State architecture requires the transmission of the object state from the server to the clients. For rigid objects, the object state $\boldsymbol{S}$ is described by the position $\boldsymbol{x_o}$ and rotation $\boldsymbol{r_o}$.

Quantization and entropy coding The communication scheme proposed in [156] uses differential coding, quantization and Huffman coding to reduce the payload of packets transmitted between the server and the clients in the CS-State architecture. The payload was

compressed by a factor of 4 without deteriorating the subjective quality significantly[2]. The entropy coding of differential data is effective in this case, because the distribution of the position/rotation difference data is very concentrated. This approach, however, does not address the high packet rates, which lead to a significant overhead due to the packet header information.

Dead reckoning The number of state updates can be reduced with the concept of *dead reckoning (DR)*, supported by the *IEEE Standard for Distributed Simulations* [148]. DR locally predicts object states using the last received updates. The server in the CS-State architecture applies the same prediction as the clients. The simulated state and the predicted state are compared at the server. Based on the position and rotation error of objects, new state updates are triggered and transmitted to the clients.

In the area of SHVEs, fixed position and rotation error thresholds have been used to reduce the number of object state updates and, hence, the packet rate (e.g., in [29], [163]). These thresholds have to be tuned carefully according to the simulation's characteristics to achieve a good trade-off between packet rate reduction and rendering fidelity. The clients in the CS-State architecture update their local object state databases as soon as a new state update arrives. The objects' positions, rotations, and velocities will change as a result of these updates. This eventually leads to a perceivable force change for the user who is currently haptically interacting with a certain object.

The magnitude of this force change is influenced by different factors. One of them is the discrepancy between the current state at the client and the newly received state update. A sudden change of the object's position and/or rotation may lead to a sudden change in rendered force, as constraint-based force rendering algorithms commonly use penetration depth and Hooke's Law as a basis for force computation (see Section 2.3.4). Hence, also the object stiffness k_o determines if a force change caused by an object state update can be perceived or not (see Equation (2.6)). As a result, the force feedback quality achieved with fixed position/rotation-based thresholds heavily depends on the applied virtual coupling (spring and damper values) in the local force rendering. Chapter 3 introduces a data reduction scheme that incorporates human perception models to find a good trade-off between fidelity and packet rate reduction.

Additionally, a proper extrapolation mechanism to predict object states until the reception of a new update has to be chosen. First-order (e.g., in [156]) or second-order (e.g., in [163]) extrapolation based on the last received position state updates as well as velocity, or velocity plus acceleration have been used. Obviously, the better the prediction, the fewer update packets have to be transmitted. The prediction quality heavily depends on the human interaction and, thus, a higher-order predictor does not necessarily perform better. The use of acceleration in position extrapolation can lead to under or overshooting and an undesirable oscillatory behavior [198]. Furthermore, numerical calculation of the first or

[2] The work in [156] considers a double precision floating point representation, which requires 8 bytes for each coordinate. Taking single precision as a reference, the compression factor reduces to 4, instead of 8 as mentioned in [156].

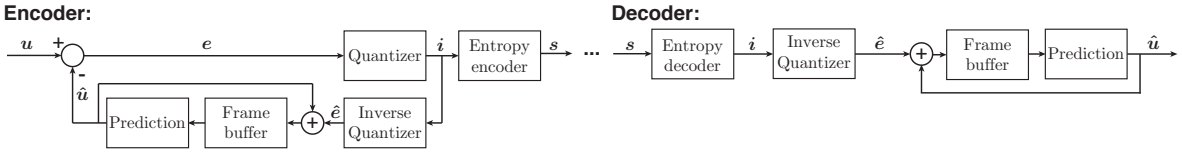


Figure 2.12: Illustration of a prediction-based mesh compression scheme using entropy coding. The decoder structure is included in the encoder. Different time-only, space-only or time-space predictors can be adopted. The structure is very similar to traditional Differential Pulse Code Modulation.

second-order derivative of discrete position signals amplifies or even introduces noise in the signals.

Dead reckoning can also be used on the device state (e.g., used in [12]) as an alternative to the deadband-based transmission of velocity information. The lossy PD-approach leads to a drift between the device positions at the client and the server if the position is not corrected regularly [40]. The DR-approach, on the other hand, bounds the position mismatch by explicitly considering the position prediction error in the decision to trigger a new device state update.

2.4.3.3 Deformable object state

The state of a deformable polygon mesh is described by the displacement of its vertices from their undeformed rest positions. The compression of static and deformable meshes has been subject of research in the area of computer graphics for many years. A comprehensive survey of the state-of-the-art can be found in [199]. Many of the available algorithms consider a complete animation, that is, a temporal sequence of mesh states, which is known beforehand and can be compressed jointly or blockwise (e.g., [200]–[203]). This does not hold, however, for real-time applications like SHVEs, where the next deformation state is not known in advance and the encoder can use only the current and the past states.

The key requirement for any feasible compression scheme in this context is a very low encoding and decoding time. This is due to the fact that the end-to-end delay should be kept as small as possible, as delay leads to a loss of transparency (see Chapter 4). For this reason, mesh compression schemes as for example presented in [181], [202], [204], [205] that cluster vertices into groups and estimate transformations that represent each group's vertex displacements, for example, by solving a least squares optimization problem, are not applicable in the considered scenario due to their computational complexity.

Mesh compression schemes with lower computational complexity are often based on the assumption of strong spatiotemporal redundancy in a small neighborhood of each vertex [199], [206]. The 3D displacement of a vertex is predicted using neighboring vertices (spatial prediction) and previous displacements (temporal prediction). Only the differences (the residuals) are transmitted after being quantized (e.g., in [181], [204], [206]–[208]) and compressed using entropy coding (e.g., in [204], [207], [208]) as illustrated in Figure 2.12.

The goal of mesh compression algorithms to be applied in computer graphics applications is to keep the introduced distortion below human visual perception thresholds, or to keep it at an acceptable level [209]. In a SHVE, it is not only the visual quality, but also the

haptic feedback quality that has to be maintained. Chapter 3 introduces a low-delay compression scheme for polygon mesh deformation data, which jointly considers the visual and haptic modalities to achieve a good trade-off between visual-haptic quality and achieved compression.

2.4.4 Fidelity and transparency

All architectures introduced in Section 2.4.2 struggle with communication delay. In the CS-State architecture, it has been found that communication delay leads to increased force magnitudes and a loss of transparency during the interaction with rigid objects [43], [44], [163], [164]. There is a very similar effect for the interaction with deformable objects.

The effect of communication delay If the user pushes against an object and communication delay exists, the object will not move or deform immediately because the physics simulation is performed on the server and the new object state has to be transmitted to the client over the communication network. The human user might recognize the delayed object state updates visually and haptically.

In the local force rendering at the client, the penetration of the virtual probe into the object increases as the user further presses against the object while waiting for the update. In turn, the force displayed to the user increases as constraint-based force rendering algorithms use the penetration depth to calculate the force feedback (see Section 2.3.4). For this reason, communication delay, that is, delayed object state updates, can lead to a loss of transparency and perceivable artifacts in the force feedback at the client.

It was found that a RTT of no more than 60 ms [43] or 100 ms [156] can be allowed before communication delay deteriorates the feedback quality significantly. These numbers, however, are only rough estimates from experimental data and heavily depend on the implementation, the application, and especially the stiffness k_o used in the local force rendering. Chapter 4 of this thesis systematically analyzes the effect of communication delay on the force magnitude rendered at the clients during the interaction with movable rigid objects and deformable objects. It derives guidelines on the maximum tolerable communication delay, which is the delay boundary for which the effect of delay becomes haptically perceivable. These guidelines incorporate the simulation parameters, the user interactions and the limitations of human perception.

Compensation schemes To compensate for the increased forces due to constant communication delay, Fujimoto *et al.* propose to reduce the spring coefficient k_o used in the local force rendering at the clients [164]. A simple heuristic is used to calculate the reduced stiffness. Following this idea, the algorithm proposed in [44] chooses a reduced spring coefficient based on a single-user LTI model of the CS-State architecture. Therein, the server-side physics simulation is roughly approximated with a 1D mass-damper system, which makes it difficult to model various contacts between objects and surfaces. This prevents the generalization of this scheme to complex multi-user interactions with simulated virtual objects in the CS-State architecture.

Chapter 4 introduces more detailed models of the CS-State architecture for the interaction with simulated rigid and deformable objects, which are necessary to generalize previous work. Based on these mathematical models, schemes to compensate for the effect of communication delay in the CS-State architecture are presented, which follow the paradigm proposed in [164] and reduce the stiffness used in the local force rendering at the clients.

2.5 Summary

This chapter has covered important background in the areas of haptic perception (human haptics, Section 2.1), haptic devices and interfaces (machine haptics, Section 2.2) and the rendering of haptic virtual environments (computer haptics, Section 2.3), which is necessary for the reader to understand the challenges and related work in the area of Shared Haptic Virtual Environments.

Section 2.4 compared three different architectures that can be conceived to implement a SHVE. The discussion of the advantages and shortcomings of these architectures motivates the research presented in the following chapters. This thesis focuses on a client/server-based architecture, where the dynamics of virtual objects are simulated by a centralized physics simulation on the server. Local copies of the VE are used to render the scene visually and haptically at the clients.

Important challenges in this architecture and the limitations of the state-of-the-art are summarized as follows:

- The haptic device position/velocity data and the object state data between the clients and the server has to be transmitted efficiently. It is important to note that:
  - State-of-the-art technology can be used to efficiently transmit the position/velocity data.
  - State-of-the-art technology to transmit object states ignores the human operator and the limitations of the human perception system.
- The delay in the communication between the clients and the server leads to changes in the force magnitudes rendered at the clients, that is, a loss of transparency. It is important to note that:
  - Related work focused only on the interaction with movable rigid objects.
  - Schemes to compensate for the effect of communication delay have been proposed for single-user interactions with simple virtual environments, which are based on either heuristics or over-simplified client/server models.
  - The state-of-the art is difficult to generalize towards arbitrary multi-user interactions with complex virtual environments due to the modeling assumptions that have been made.

This thesis addresses these challenges. Chapter 3 addresses the perception-based data reduction for the transmission of rigid and deformable object states in the considered client/server architecture. Chapter 4 systematically analyzes the effect of communication delay on the force magnitude rendered at the clients during the interaction with movable and deformable objects and presents schemes to compensate for it.

Chapter 3

Perception-based data reduction in client/server-based SHVEs

The CS-State architecture requires the transmission of the device state (position, velocity) from the clients to the server and the transmission of the simulated object state (position, rotation, deformation) back to the clients. This leads to a significant communication load on the network. As delay in the communication between the clients and the server can deteriorate the system fidelity, the delay introduced by data reduction and compression schemes should be kept at a minimum. For this reason, the blockwise processing and coding of device and object states is not feasible. Instead, new state updates should be packetized and transmitted once available.

The data reduction schemes presented in this chapter are designed to introduce only negligible delay into the communication from the server and the clients. Distinct from the work reviewed in Section 2.4.3.2 and Section 2.4.3.3, they incorporate models of the human perceptual limitations to allow for a good trade-off between system fidelity and data reduction.

The transmission of object states differs between the interaction with rigid and deformable objects. The following Section 3.1 summarizes the differences. Section 3.2 introduces a perception-based data reduction scheme to reduce the number of object state update packets transmitted from the server to the clients during the interaction with rigid objects. Section 3.3 introduces a low-delay lossy compression scheme for the transmission of 3D polygon mesh deformation data from the server to the clients during the interaction with deformable objects. Both data reduction schemes consider the visual and the haptic modality individually to find a balance between data reduction and visual-haptic fidelity.

The ideas and contributions developed in this chapter have been published in parts in [3], [8], [11], [13]. The compression scheme for 3D polygon mesh deformation data was developed in collaboration with Wolfgang Freund [210].

3.1 Traffic characteristics

The rigid body physics simulation in a HVE normally runs at the haptic rate of 1 kHz. A new object state, which includes only the object's position and the rotation, is ready to be transmitted every millisecond [157]. Hence, a high packet rate and a low payload charac-

	Packet rate	Payload	Bit rate[1]	Header to payload ratio[1]
Forward channel				
Device position	1000	12 byte [2]	320 kbit/s	2.33:1
Backward channel (CS-State)				
Single rigid object state	1000	24 byte [3]	416 kbit/s	1.17:1
Deformable object state	60	8568 byte [4]	4.12 Mbit/s	1:306
Backward channel (CS-Force) [5]				
Force	1000	12 byte [6]	320 kbit/s	2.33:1

[1] The calculations of the bit rate and the header to payload ratio assume the transmission with UDP (8 bytes header) over IPv4 (20 bytes header).
[2] 3D position vector with 4 byte per coordinate (single precision floating point)
[3] Position vector ($3 \cdot 4$ bytes) and Euler angles ($3 \cdot 4$ bytes). No velocity data is required if the object state is transmitted at a rate of 1 kHz.
[4] 714 vertices with 3D displacement vector (12 bytes) per vertex (see Figure 2.4b)
[5] Video and audio streams are not considered here.
[6] 3D force vector (12 bytes) with 4 bytes per coordinate

Table 3.1: A high packet rate and a small payload characterize the transmission of the device state in all communication architectures conceived to implement a SHVE. The transmission of forces in the CS-Force architecture shares the same characteristics. In the CS-State architecture, the packet rate and the payload differ between the transmission of rigid and deformable objects states. The numbers given for the deformable object state are only exemplary.

terize the transmission of rigid object states. This leads to a high packet header to payload ratio. The size of the payload increases with the number of object states included in each update packet. The user interacts haptically, however, only with a very limited number of objects at a time. The high update rate is required only for these objects and the state of the other objects, which are rendered only visually at the clients, can be updated at a lower frequency. For these reasons, the efficiency of schemes that compress the payload and neglect the high packet rate, briefly surveyed in Section 2.4.3.2, is limited. The scheme presented in Section 3.2 focuses on packet rate reduction and exploits visual and haptic thresholds to avoid unnecessary state updates.

The state of a deformable polygon mesh is described by the position of its vertices. Hence, the amount of information to be transmitted increases with the size of the mesh and is much higher as compared to the state of a rigid object. As the simulation of object deformations is computationally more complex than the simulation of rigid body dynamics (see Section 2.3.2), the achievable temporal simulation rate is lower. Compared to the transmission of rigid object states, a lower packet rate and higher payload characterize the transmission of deformable object states if the simulation is based on a multi-rate rendering approach (see Section 2.3.4.2). This requires the compression of the packet payload instead of reducing the packet rate. The compression scheme presented in Section 3.3 applies predictive coding and entropy coding to reduce the size of the payload. It groups vertex updates based on their influence on the haptic and visual modalities and adapts the coding parameters accordingly.

Table 3.1 summarizes the traffic characteristics in the CS-State architecture and compares

them to the CS-Force architecture. Traffic in the CS-Force architecture equals the traffic in bilateral teleoperation systems. It is important to note that the CS-Force architecture requires the transmission of video (and possibly also audio) streams in addition to the haptic data listed in Table 3.1.

3.2 Perception-based data reduction for the interaction with movable rigid objects

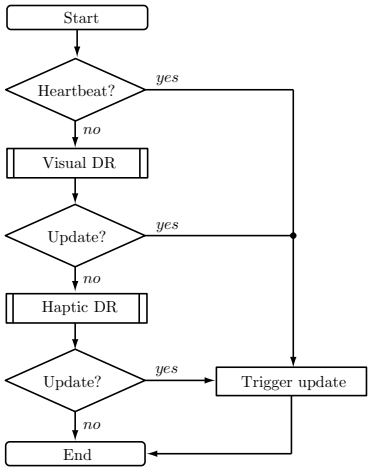
The perception-based data reduction scheme, which is presented in the following, extends the dead reckoning (DR) scheme explained in Section 2.4.3.2. It reduces the number of rigid object state updates transmitted from the server to the clients. Analogous to the scheme defined in the *IEEE Standard for Distributed Simulations* [148], it is based on the prediction of object positions and rotations using previous object states and triggers new state updates based on prediction errors. In contrast to the state-of-the-art, it incorporates a mathematical model of the human haptic perceptual limitations to dynamically adjust the prediction error thresholds used in the state update triggering. This allows us to find a good balance between packet rate reduction and force rendering fidelity.

3.2.1 Perception-based object state update triggering

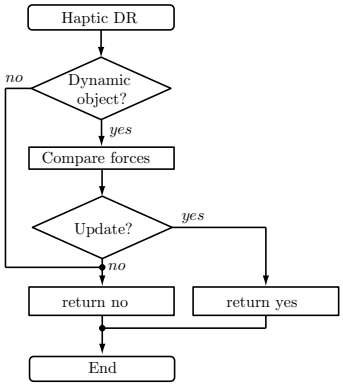
Each client locally predicts object states using the last received state updates. The server applies the same prediction as the clients and triggers new state updates eventually. The decision process used at the server to decide if a new object state update should be sent to the clients is illustrated in the flowcharts in Figure 3.1. There are three different update triggers.

The first two triggers, called *Heartbeat* and *Visual DR* (VDR) in the following, are inherited from the standard dead reckoning process defined in [148]. If a time-out since the last object state update occurs, a Heartbeat update is triggered. The VDR process uses individual thresholds for the position and rotation error. It compares the current true object state $\boldsymbol{S}$, containing the position $\boldsymbol{x_o}$ and rotation $\boldsymbol{r_o}$, calculated by the centralized physics simulation, to the predicted object state $\hat{\boldsymbol{S}}$ and triggers updates accordingly. The object state data, which is transmitted to the clients, also includes the object's translational and rotational velocities $\dot{\boldsymbol{x}}_{\boldsymbol{o}}$ and $\dot{\boldsymbol{r}}_{\boldsymbol{o}}$, respectively, to facilitate the prediction at the receiver.

The human visual and haptic sensory modalities differ in their spatial and temporal resolution capabilities. Human haptic perception is known to be sensitive to temporal force changes (see Section 2.1). Although the object trajectory looks visually smooth, the user interacting with an object might perceive artifacts in the force feedback caused by the object state updates. Constraint-based force rendering algorithms calculate interaction forces according to the penetration depth of the haptic device and a stiffness k_o assigned to the object (see Section 2.3.4). This stiffness value is chosen desirably as large as possible for the rendering of a rigid object, but is limited by the capabilities of the haptic device and stability constraints (see Section 2.2). At the same time, the stiffness k_o is the major factor influencing the haptic perceptibility of object state updates.



(a) Complete perception-based traffic control process including visual and haptic dead reckoning.



(b) Haptic dead reckoning in detail

Figure 3.1: Flowchart describing the proposed dead reckoning decision process (reproduced from [3]).

The third update trigger, the *Haptic DR* (HDR) process, checks for force changes, instead of relying solely on fixed position/rotation thresholds, to account for this. Figure 3.1b illustrates the HDR trigger. Obviously, this procedure is necessary only if the user currently interacts with a dynamic and movable rigid object. Updates for static objects are not necessary, as every client maintains a local copy of the VE. The HDR trigger decides whether to transmit an update to the client or not based on the perceptibility of the force change caused by the object state update. It is mathematically described by

$$p \cdot \|\boldsymbol{f}_{s,i}\| \begin{cases} < \|\Delta \boldsymbol{f}_{s,i}\| \Rightarrow \text{transmit object state update} \\ \geq \|\Delta \boldsymbol{f}_{s,i}\| \Rightarrow \text{no transmission} \end{cases} \tag{3.1}$$

where $\boldsymbol{f}_{s,i}$ denotes the currently rendered force vector for client i at the server and $\Delta \boldsymbol{f}_{s,i}$ is the estimated force error, that is, the force change the user would feel if the object state would be updated directly. It is calculated as

$$\Delta \boldsymbol{f}_{s,i} = \hat{\boldsymbol{f}}_{s,i} - \boldsymbol{f}_{s,i} \tag{3.2}$$

where $\hat{\boldsymbol{f}}_{s,i}$ denotes the estimated force vector, which depends on the predicted object state $\hat{\boldsymbol{S}}$.

The deadband parameter p (also referred to as *traffic control parameter* in the remainder) is inspired by Weber's Law (see Section 2.1.3). The isotropic haptic perception model applied in Equation (3.1) defines a spherical deadzone as discussed in Section 2.4.3 and introduced in [40]. Updates are triggered only if the tip of the three dimensional vector $\hat{\boldsymbol{f}}_{s,i}$ lies outside this spherical volume, defined by the reference vector $\boldsymbol{f}_{s,i}$ and the deadband parameter p. Figure 3.2 illustrates this procedure. Only the normal components of the rendered forces (parallel to the normal vector of the contacted triangle) are considered, since object state updates at the clients cause these components to change. Other force components are due to friction rendering, which is in any case rendered locally at the client.

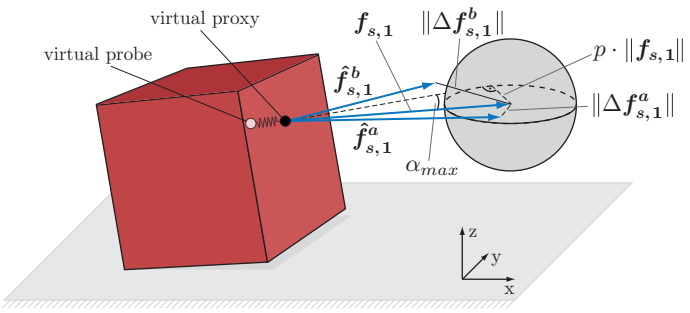


Figure 3.2: Illustration of the isotropic spherical deadzone for client 1. The estimated force $\hat{f}^a_{s,1}$ falls within the sphere defined by the reference force $f_{s,1}$ and the deadband parameter p and, thus, causes no update to be triggered. Contrarily, $\hat{f}^b_{s,1}$ does not comply with the deadzone and a new object state update has to be triggered (adapted from [3]).

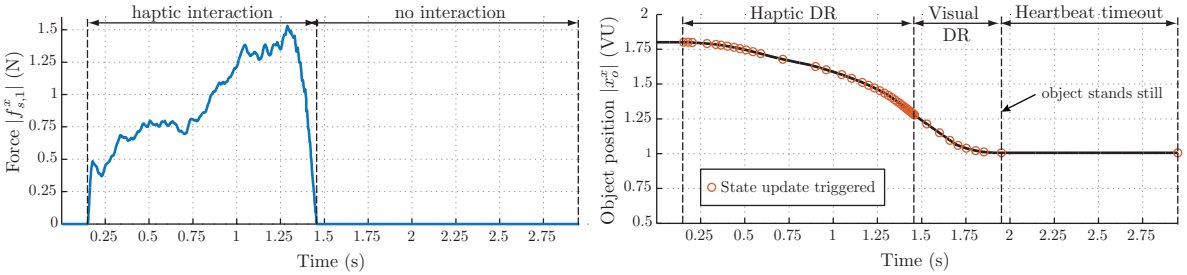
One important property of the isotropic spherical deadzone, which is assumed in the HDR trigger in Equation (3.1), is the maximum angle between two compared force vectors. The maximum angular error is reached if the estimated force $\hat{f}_{s,i}$ is tangential to the spherical deadzone and is calculated as [40]:

$$\alpha_{max} = \arcsin(p) \tag{3.3}$$

This implies that the HDR process triggers a new update as soon as the angular error between the actual and the estimated force exceeds α_{max}, independent of the currently rendered force magnitude (i.e., independent of $\|f_{s,i}\|$). The angular change of the rendered normal force is directly related to the change of orientation of the contacted triangle and, thus, to the orientation of the rigid object itself.

The forces, object position, and triggered state updates during a short interaction with a rigid cube have been recorded to illustrate the characteristics of the three-fold update decision process involving the Hearbeat, VDR, and HDR update triggers. The x-component of the rendered force signal and the object position are shown in Figure 3.3. The cube is pushed along the x-axis in the virtual environment. The rendered force is non-zero during the contact with the object. HDR triggers more updates to improve the haptic feedback displayed to the user at the client. At around $t = 1.3\,\mathrm{s}$, the user stops pushing, and now, only VDR triggers new state update packets and the update rate is obviously reduced. Finally, a first heartbeat update is triggered $1\,\mathrm{s}$ after the object stands still.

The parameter p, which is used in the HDR trigger, can be tuned with subjective user tests to find a good trade-off between perceptual transparency and packet rate reduction. It is expected to be within the range of the Weber-fraction for forces. By considering the resulting force changes in the dead reckoning decision process, the error thresholds are not fixed, but are dynamically adapted to the characteristics of the virtual environment, the haptic interface, the current user interaction and the limitations of the human haptic perception system. This enables perceptually robust traffic control of object state updates in the CS-State architecture.



(a) Recorded force signal at the server

(b) Object position and triggered state updates

Figure 3.3: Illustration of the perception-based traffic control procedure. Signals were recorded during an exemplary interaction with a rigid cube. The deadband parameter is set to $p = 8\%$. The user stops pushing after 1.5 s, but the object continues to move due to its inertia. Only x-coordinates are shown (adapted from [3]).

3.2.2 Experimental comparison between perception-based data reduction in the CS-State and CS-Force architecture

Both the CS-State and the CS-Force architecture require the high-rate transmission of information from the server to the clients during the interaction with rigid objects (see Section 3.1). If a new force sample in the CS-Force architecture or a new object state in the CS-State architecture is transmitted at the haptic rendering rate and there is no communication delay, the force signals at the server equal the force signals at the clients. In fact, the employed architecture does not influence the force and position signals at the clients and the server in this case. The only difference is the type of information transmitted between them.

The perceptual deadband data reduction scheme (see Section 2.4.3) can be used to reduce the number of transmitted force packets in the CS-Force architecture, where the force is rendered only on the server. On the other hand, the previously introduced perception-based traffic control scheme can be used to reduce the number of rigid object state updates in the CS-State architecture. A subjective test, denoted as the *architecture comparison experiment*, was performed to evaluate whether the difference in the type of transmitted information influences the achievable packet rate reduction, although both data reduction schemes rely on the same perception model.

3.2.2.1 Method

The experiment consisted of two sub-experiments:

1. Sub-experiment 1: Bases on the CS-Force architecture explained in Section 2.4.2.1. Perceptual deadband data reduction in combination with a first-order linear predictor [40] (see Section 2.4.3) was applied to reduce the number of force packets transmitted from the server to the clients.

2. Sub-experiment 2: Bases on the CS-State architecture explained in Section 2.4.2.2. The previously introduced data reduction scheme with a first-order linear predictor on the object position was applied to reduce the number of object state updates transmitted

from the server to the clients. Forces displayed to the users were rendered locally at the clients.

Apparatus The CS-State and the CS-Force architecture were conceived to implement a SHVE with two users. The server and clients ran on different workstations, which were connected over a local network. The communication delay was negligible. The force rendering rate at the server, and at the clients in the CS-State architecture, was fixed to 1 kHz. The object dynamics were simulated on the server by means of ODE, an open source physics simulation for rigid body dynamics [211]. ODE was set to simulate coulomb friction between the object and the supporting surface ($\mu_{static} = \mu_{kinetic} = 0.7$). Gravity in the VE was set to $[0, 0, 9.81\,\mathrm{m/s^2}]^T$. Both clients were equipped with a Geomagic Touch haptic device [79]. The seating posture and hand-device configuration were standardized across subjects. In order to prevent distractions due to noise emanating from the haptic device, subjects wore headphones playing music. In addition, distractions due to visual observation of the haptic device were avoided by blocking the view to the haptic device with a cardboard.

Task A pursuit tracking task, wherein two users collaboratively pushed/pulled a virtual cube to track the movements of a reference cube as closely as possible, was conceived for the evaluation. This was motivated by the bidirectional nature of haptics, which may lead to widely varying haptic interactions across subjects, and, thus, to widely varying subjective experiences. Note that the achieved packet rate reduction depends on the user interaction and, hence, might strongly vary between trials and subjects.

With two humans involved, this issue is compounded further. The specified reference signal, that is, the position of the reference cube as a function of time, helps to reduce the variability between different runs of the experiment and establishes a common basis for evaluation across subjects [212]. Additionally, one of the two humans involved in the task was the same throughout the whole experiment.

The pursuit tracking task consisted of a 1-DOF dynamic virtual environment (refer to Figure 3.4). In this VE, the “reference" cube had to be pursued as closely as possible with the “pursuit" cube. The reference cube served only as a visual reference. The subjects could haptically interact only with the pursuit cube and push or pull it. The pulling user was attached to the pursuit cube with a virtual “pull" spring. Contact between the user and the cube was lost if the spring force exceeded 2 N. This was done to ensure that a single user alone was not capable of executing the task successfully. The reference cube moved from right to left and back to the start position twice during a single run. Its velocity was controlled to have a sinusoidal profile (see Figure 3.4).

Procedure A training session was conducted at the beginning of the experiment to eliminate individual differences related to experience with the experimental setup. Herein, the task to be performed was explained and the subjects had the chance to get familiar with the task execution, the experimental setting and the parameter range. A within-subject experi-

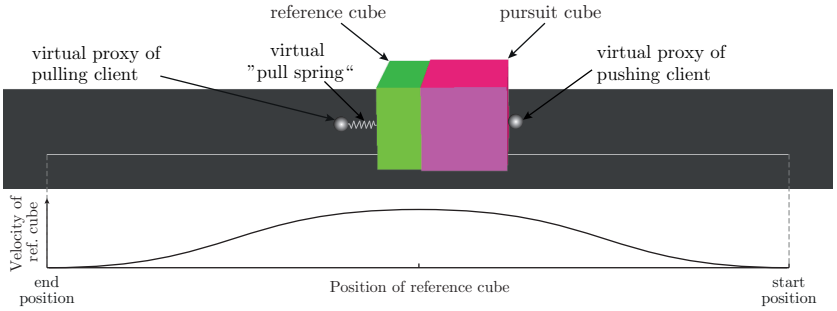


Figure 3.4: Illustration of the the dynamic virtual environment used for the two-user pursuit tracking task in the architecture comparison experiment (adapted from [11]).

mental design was used, with all participants executing the same haptic task under the same parameter settings.

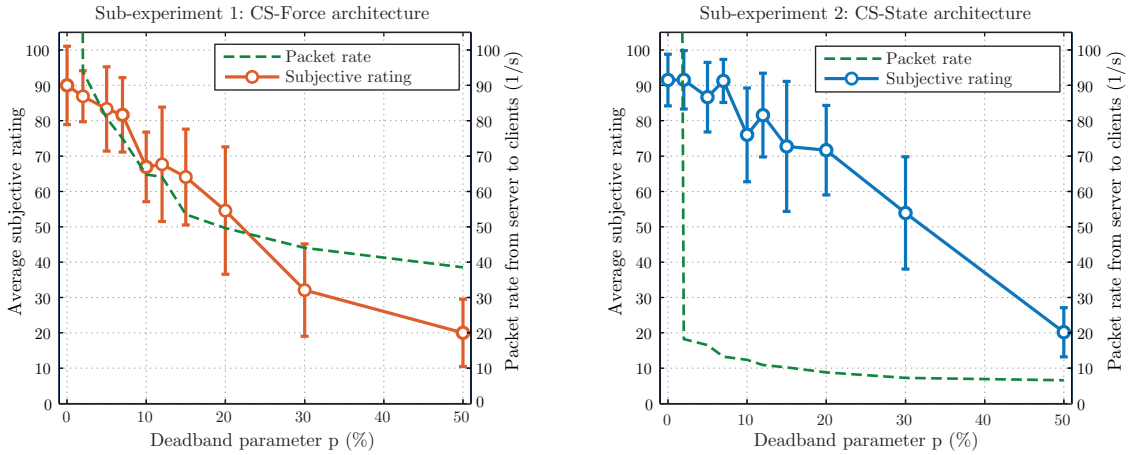
After the completion of every run, subjects were asked to rate the haptic feedback quality on a scale from 0 to 100. Two reference stimuli, corresponding to the ratings of 0 and 100, were displayed to the user at the beginning of every sub-experiment. The stimulus corresponding to the rating of 100 was the best that the system could offer (no packet rate reduction, i.e., no distortion or artifacts). The one corresponding to the rating of 0 was the worst (deviating the most from the best-possible quality, i.e., the highest packet rate reduction), while still allowing successful task completion.

Stimuli Only the haptic transmission rates from the server to the clients were measured in these experiments, neglecting the rate needed to transmit the device positions to the server and the additional rate needed for transmitting the video stream in sub-experiment 1. Each sub-experiment had 10 runs, within which the deadband parameter p was varied between 0 % and 50 %. The order of sub-experiments changed randomly from one subject to another, along with the order of the parameter values within each sub-experiment.

Participants Fourteen subjects participated in the evaluation, 1 of them female and 13 male. The participants' age was in the range of 25-43 with a mean of 29 years. Two of them were left-handed, the others right-handed. Four of them were researchers working in the area of haptic data compression and daily users of kinesthetic haptic devices. The remaining subjects have frequently participated in subjective tests for the evaluation of haptic compression schemes in the past and were therefore familiar with haptic devices.

3.2.2.2 Results and discussion

The results of the subjective evaluation for the tested deadband parameter range and the resulting mean packet rate from the server to the clients are shown in Figure 3.5. The perceived force feedback quality decreases with an increasing deadband parameter p. An increased deadband parameter reduces the required packet rate at the cost of a force feedback quality degradation.



(a) Results of sub-experiment 1

(b) Results of sub-experiment 2

Figure 3.5: Results of the subjective architecture comparison experiments. The error bars denote the mean and standard deviation of the subjective ratings between participants. The dashed lines denote the average packet rate from the server to the clients (adapted from [11])

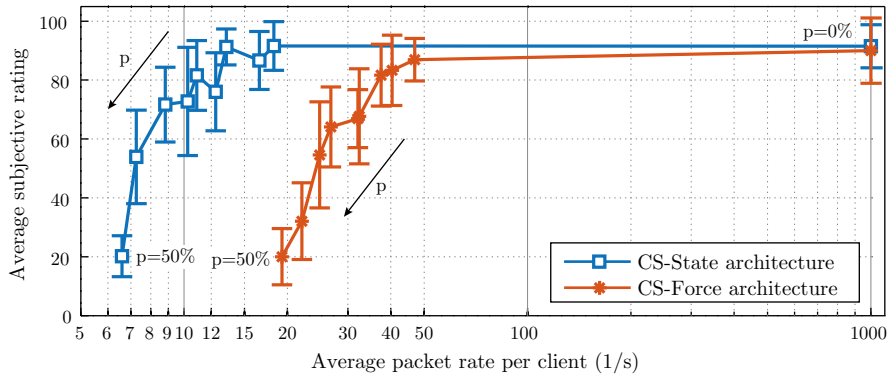


Figure 3.6: Resulting packet rates per client and user ratings (mean and standard deviation) in the two sub-experiments (adapted from [11]).

Figure 3.6 shows the resulting average packet rate per client for the evaluated deadband parameters and the corresponding subjective ratings on a semi-logarithmic scale. A desired operating point should lie as far up (high subjective rating) and as much towards the left as possible (low average packet rate).

Note that Figure 3.6, contrary to Figure 3.5, shows the average packet rate per client. The comparison of the two architectures is fairer in this case. The CS-State architecture has the advantage that multicasting can be used, as a single globally valid object state has to be transmitted to the clients to update the copies of the VE. The CS-Force architecture requires individual force samples to be transmitted to every client and, hence, the packet rate increases linearly with the number of connected clients.

It can be observed that the slope of the quality degradation is very similar for both architectures. This is due to the fact that the two applied perceptual data reduction schemes in the CS-Force and CS-State architecture are based on the same mathematical perceptual model and both deploy a first-order linear predictor.

The graph representing the CS-State architecture in Figure 3.6, however, is shifted to the left. The CS-State architecture requires less packets to achieve the same force feedback quality as the CS-Force architecture.

This gain can be explained as follows. When a user gets in contact with a rigid object, high-frequency transient forces lead to a large number of transmitted force samples in the CS-Force architecture [169], [213]. Also, the mass of the virtual object acts as a low-pass filter in the physics simulation. The object's position/velocity does not change significantly due to high-frequency low-amplitude force inputs. Hence, less state update packets have to be sent in the CS-State architecture.

In summary, although the main purpose of the local force rendering at the clients in the CS-State architecture is to increase the robustness against communication delay (see Section 2.4.2.4), it also reduces the load on the server's network interface. Less packets compared to the CS-Force architecture have to be transmitted to achieve the same haptic feedback quality, it requires no video streaming, and allows for the use of multicasting. For these reasons, the CS-Force architecture, which has been studied for example in [38], [39], [169], should be implemented only if very lightweight clients without any local rendering capabilities have to be used.

3.2.3 Experimental comparison between different object state interpolation techniques

The previously discussed perception-based data reduction scheme updates the object states at the clients immediately, as soon as a new object state has been received. This leads to sudden jumps in the rendered force feedback displayed to the user. These force changes are kept below human detection thresholds, as discussed in Section 3.2.1, to achieve satisfactory force feedback.

Interpolation methods to smoothly converge from the old to the new object state can be used at the clients in the CS-State architecture. This avoids abrupt changes in the force feedback and promises an additional packet rate reduction. Presumably, higher error thresholds can be chosen, if a proper interpolation method is adopted, while satisfactory force feedback is still maintained. Experimental evaluation, which is presented in Section 3.2.5, is necessary to prove this assumption.

Different approaches, based on first-order [148] or higher-order polynomials [198], [214] have been proposed to smoothly correct the object states for distributed applications without haptic feedback. There is, however, no well-accepted model for a perceptually convincing method, even for visual-only applications [215]. Similarly, there is a lack of knowledge of which method should be applied in SHVEs to achieve perceptually convincing haptic force feedback. This section presents an experimental comparison of three commonly used interpolation techniques to find the haptically preferred method.

3.2.3.1 Object state interpolation techniques

The three interpolation techniques, which are known from literature and used in computer graphics and networked games, are briefly introduced in the following. The formulas correspond to the interpolation of an object's position. Similarly, they can be used with Euler angles to correct rotation errors [148], [198].

Let $\boldsymbol{S}^* = \{\boldsymbol{x}_o^*, \dot{\boldsymbol{x}}_o^*\}$ denote an object state, where $\boldsymbol{x}_o^* \in \mathbb{R}^3$ is the position and $\dot{\boldsymbol{x}}_o^* \in \mathbb{R}^3$ the velocity in global coordinates.

The current object state at time $t = t_0$ when a new update is received at a client is denoted by $\mathbf{S}^0$, and the newly received object state by $\mathbf{S}^1$. The time used for the convergence from the old to the new position is denoted as t_N and is pre-decided before the interpolation starts. With this, the interpolation counter

$$i(t) = (t - t_0)/t_N \quad \text{with} \quad t_0 \leq t \leq t_0 + t_N \tag{3.4}$$

can be defined. The inter-arrival time between two consecutive object state updates received at a client is not constant, as the server non-uniformly triggers new state updates. In case an update arrives during an active interpolation process, the current interpolation should be stopped immediately and the new state applied directly. Otherwise the client falls behind the intended movement and the inconsistency between the server and the clients increases.

The interpolation interval t_N used to smoothly correct the inconsistency at the client can be chosen, for example, based on a simple heuristic as follows. The client measures the packet inter-arrival time of the M last packets, calculates its mean value and conservatively chooses $P < 100\,\%$ of this value as t_N. If an interpolation still needs to be aborted, the interpolation interval can be halved. The best subjective results in the experiments conducted in [13] have been achieved with $M = 5$ and $P = 75\,\%$. This heuristic is also applied in the remainder.

Assuming a linear prediction of object positions at the client-side, the predicted target position $\hat{\boldsymbol{x}}_o^1$ can be calculated with $\mathbf{S}^1$ as

$$\hat{\boldsymbol{x}}_o^1 = \boldsymbol{x}_o^1 + \dot{\boldsymbol{x}}_o^1 t_N \tag{3.5}$$

Linear interpolation The most straightforward interpolation between the current position $\boldsymbol{x}_o^0$ and the target position $\hat{\boldsymbol{x}}_o^1$ is a linear interpolation [148], which is given for the object position $\boldsymbol{x}_o(t)$ at time t as

$$\boldsymbol{x}_o(t) = \boldsymbol{x}_o^0 + \left[\hat{\boldsymbol{x}}_o^1 - \boldsymbol{x}_o^0\right]\, i(t) = \boldsymbol{x}_o^0\,[1 - i(t)] + \boldsymbol{x}_o^1 i(t) + \dot{\boldsymbol{x}}_o^1 t_N i(t) \tag{3.6}$$

Linear interpolation leads to abrupt changes in the object velocity at the times $t = t_0$ and $t = t_0 + t_N$. Thus, the object's trajectory in the virtual environment is not C^1-continuous.

Hermite interpolation Alternatively, the trajectory can be smoothed with a Hermite curve segment, which takes also the old and new velocity, $\dot{\boldsymbol{x}}_o^0$ and $\dot{\boldsymbol{x}}_o^1$, respectively, into account. Thereby C^1 continuity is maintained. With the common Hermite curve segment [214] and

the predicted target position $\hat{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{1}}$, the object position $\boldsymbol{x_o}(t)$ is given as

$$\begin{aligned}\boldsymbol{x_o}(t) = \boldsymbol{x_o^0}\left[2i(t)^3 - 3i(t)^2 + 1\right] + \boldsymbol{x_o^1}\left[-2i(t)^3 + 3i(t)^2\right] \\ + \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{0}}\, t_N\left[i(t)^3 - 2i(t)^2 + i(t)\right] + \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{1}}\, t_N\left[-i(t)^3 + 2i(t)^2\right]\end{aligned} \tag{3.7}$$

The drawback of the Hermite interpolation is the tendency to show an oscillatory behaviour [198]. Although the object trajectory is smooth, higher acceleration is needed during interpolation to correct the inconsistency within the interpolation time t_N.

Projective velocity blending The so-called projective velocity blending for virtual environments, proposed by Murphy [198], uses two combined linear interpolations. According to Murphy, this method leads to a visually convincing interpolation in networked games and tends to show less oscillations than the Hermite interpolation. The resulting curve segment, following the equations given in [198], can be calculated as

$$\boldsymbol{x_o}(t) = \boldsymbol{x_o^0}\left[1 - i(t)\right] + \boldsymbol{x_o^1} i(t) + \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{0}} t_N\left[i(t)^3 - 2i(t)^2 + i(t)\right] + \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{1}} t_N\left[-i(t)^3 + 2i(t)^2\right] \tag{3.8}$$

The first part of Equation (3.8) consists of a linear interpolation between the old and new object position, equal to the first two terms in Equation (3.6). The prediction is incorporated into the latter two terms as in Equation (3.7). It is a mixture of the two aforementioned approaches and C^1-continuity is not achieved. The velocity used for the extrapolation is blended from $\dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{0}}$ to $\dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{1}}$ compared to linear interpolation, where only $\dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\mathbf{1}}$ is used in the extrapolation term.

3.2.3.2 Method

The *interpolation preference experiment* was based on the methodology described in [216]. It used several pairwise comparisons of the previously described interpolation methods to determine a haptically preferred interpolation approach.

Apparatus The apparatus consisted of the same hardware as used in the architecture comparison experiment explained in Section 3.2.2. The SHVE was based on the CS-State architecture with only a single client connected to the server. This client transmitted the device position to the server at the full rate of 1 kHz. On the return channel from the server to the client, the perception-based data reduction scheme, presented in Section 3.2.1, was used to reduce the number of object state updates. A new state update was triggered if the expected force change due to a sudden object position change exceeded 25 % of the current force displayed to the user ($p = 25\,\%$, see Equation (3.1)). This parameter was determined in pilot tests, where it was found to be a reasonable trade-off between operability and perception of artifacts. According to the known human perceptual limitations, this force change is easily perceivable (see Section 2.1.3). During the experiments, the subjects wore headphones with active noise cancellation to prevent any auditory feedback from the haptic device or distractions from the environment.

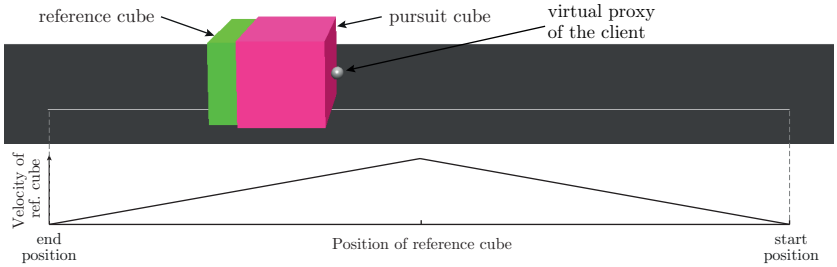


Figure 3.7: Illustration of the single-user pursuit tracking task with pursuit cube (pink, haptically active) and reference cube (green, only visual) conceived for the object state interpolation preference experiment (adapted from [13], © 2013 IEEE).

Task A simple to accomplish task, which avoids as much variability between runs and subjects as possible, is desirable to compare artifacts introduced by different interpolation schemes. Therefore, a single-user 1-DOF pursuit tracking task was conceived. In a single run, the subject had to push a rigid cube from a start location to an end location following a reference cube as closely as possible (see Figure 3.7). The reference cube's movement was experimentally controlled to show constant acceleration in the first half of the trajectory and constant deceleration afterwards. Additionally, the movement of the virtual probe in the VE was also artificially constrained to 1-DOF to further simplify the task execution.

Procedure and stimuli Two runs (denoted as run A and run B) were presented sequentially in each comparison. Afterwards, the subject was asked to select his preferred run. The subject also had the option to state that there was no perceived difference between both runs.

The comparisons included all possible combinations between the three interpolation methods, which are abbreviated as S_1 for linear interpolation, S_2 for cubic Hermite interpolation and S_3 for velocity blending. It also included comparisons of the three interpolation schemes to a reference without any data reduction (denoted as S_4, with 1000 object state updates per second transmitted from the server to the client, no interpolation applied) and a reference with perception-based data reduction, but no object state interpolation applied (denoted as S_0, with $p = 25\,\%$), that is, a setting as evaluated in Section 3.2.2. Additionally, there were comparisons with both runs A and B set to be identical. Each comparison was evaluated twice, leading to an overall number of 24 comparisons per subject.

The order of comparisons and runs within a comparison was selected randomly to avoid any systematic error in the results. Repetition of runs within a comparison or the repetition of the comparison itself on behalf of the subject was not allowed.

Participants Twelve subjects participated in this experiment. Two of them were novice users of haptic devices. Ten subjects already participated in previous haptic experiments in our lab. A training session was conducted with every subject before the actual experiment. One subject was excluded from the later analyses as the two repeated comparisons were judged differently by more than 50 %, indicating random answers.

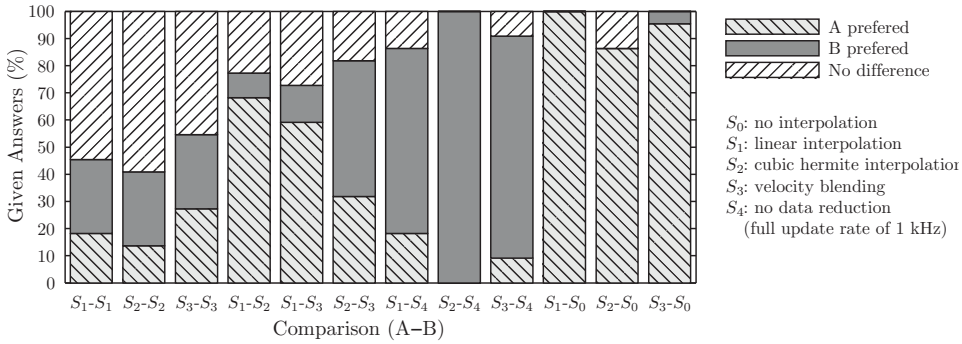


Figure 3.8: User answers in the *interpolation preference experiment* (adapted from [13], © 2013 IEEE)

3.2.3.3 Results

The user preference for all comparisons is shown in Figure 3.8. The first three bars show the comparisons wherein the runs A and B use the same interpolation technique. The subjects correctly chose the *no difference* answer only in 50 % to 60 % of the cases, showing the subjects' difficulties of comparing two separate runs. Besides the uncertainty of every subjective experiment, the following characteristics could be a reason for this. Although the experiment was built around a simplified task to reduce the variation between different experimental runs, a human is not able to exactly repeat a run and to perfectly follow the reference cube. Occasionally, a subject realizes during task execution that the error between the user cube and the reference cube is growing too big and quickly adapts his movement accordingly. Due to the nature of the applied perception-based data reduction scheme, new state updates are triggered only if the linear client-side prediction fails, meaning whenever sudden accelerations in movement occur. New updates lead to new interpolations at the client and, as a result, to perceived artifacts in the force feedback displayed to the user.

The first three comparisons were explicitly included in the experiment to counteract this problem. They served as identity norms as suggested in [216]. Statistical analysis using Pearson's chi-squared test was applied to investigate if other comparisons differ from them significantly. This allowed conclusions if any interpolation mode performs differently compared to the corresponding norm. The results were further investigated for significant user preferences by applying the statistical method described in [216].

The comparisons S_1-S_2, S_1-S_3 and S_2-S_3 in Figure 3.8 are direct pairwise comparisons between the three interpolation methods. Linear interpolation seems to be clearly preferred compared to the other two interpolation modes. Compared to cubic Hermite interpolation (S_1-S_2), it was preferred in 68 % of the comparisons, while the cubic approach was preferred in 9 %, and the subjects perceived no difference in 23 % of the runs (in short 68:9:23). Pearson's chi-squared test indicated a significant difference compared to the identity norm S_1-S_1 ($\chi^2(2, N = 22) = 37, p < 0.001$), indicating that method S_1 and S_2 do not perform equally. The follow-up analysis showed a significant preference for linear interpolation ($\chi^2 = 10.89, p = 0.0008$ with $\alpha = 0.05$). Similar results were found for the comparison to velocity blending (S_1-S_3) with a significant difference to the identity norm (59:14:27, $\chi^2(2, N = 22) = 24.75, p < 0.001$) and a preference for linear interpolation

($\chi^2 = 6.70, p = 0.009$). The comparison between cubic Hermite interpolation and velocity blending (S_2-S_3) was significantly different from the identity norm S_2-S_2 (32:50:18, $\chi^2(2, N = 22) = 15.73, p = 0.0003$). The follow-up analysis indicated no clear preference for one of these two methods ($\chi^2 = 0.90, p = 0.34$).

The comparisons of the three methods to the two references are shown in the last six bars in Figure 3.8. Clearly, the reference case with 1000 packets per second was preferred in nearly all of the runs (compare S_1-S_4, S_2-S_4, S_3-S_4 bars). Similarly, every interpolation scheme was preferred to the reference without interpolation (compare S_1-S_0, S_2-S_0, S_3-S_0). These results are not surprising, as a relatively large traffic control parameter was chosen for the data reduction in the experiment.

3.2.3.4 Discussion

The results of this experiment encourage the use of linear interpolation for object positions in the proposed perception-based data reduction scheme. One reason for the user's preference for the linear interpolation is probably the acceleration needed in both cubic methods to correct the inconsistency. During linear interpolation, there is a sudden change in velocity only at the beginning and in the end of the interpolation. The other two schemes adapt more severely to the old (new) velocities at these time instances. Thus, they need to catch up during the interpolation itself. This acceleration was reported after the experiment to be a driving factor in the subjective preference decisions. Although, for example, projective velocity blending might lead to a convincing visual object movement as stated in [198], in terms of displayed force feedback, the users prefer a trajectory with less acceleration.

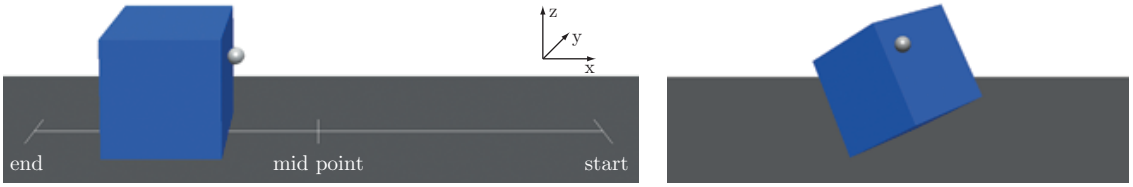
Similarly, spherical linear interpolation (SLERP) [217] uses constant rotational velocity during the interpolation and, thus, is comparable to the linear interpolation of positions. It can be adopted to correct the rotation error in the proposed data reduction scheme.

3.2.4 Experimental traffic control parameter identification

The previous experiment has shown that linear interpolation of object states leads to the haptically most convincing force feedback. The traffic control parameter setting that leads to a satisfying force feedback quality, however, has not been investigated so far. Two experiments, denoted as the *1-DOF* and *6-DOF parameter adjustment experiments*, are presented in the following. The goal is to find parameter settings p^* for which the users cannot distinguish between traffic control enabled or disabled while interacting with a rigid object. The experiments also test the effect of linear object state interpolation on the proposed perception-based data reduction. Higher values for p^* are expected if object state interpolation is enabled, that is, fewer object state updates have to be transmitted from the server to the clients.

3.2.4.1 Method

Apparatus The 1-DOF and 6-DOF parameter adjustment experiments used the same apparatus as the interpolation preference experiment (see Section 3.2.3). Again, a single client was connected to the server and the transmission of haptic device state updates from the



(a) VE used in the 1-DOF parameter adjustment experiment. The rigid cube is only movable along the line from the start point to the end point. At the mid point, the user has to reduce the movement velocity.

(b) VE used in the 6-DOF parameter adjustment experiment. The rigid cube can translate and rotate in 3 degrees-of-freedom each.

Figure 3.9: Virtual environments used for the subjective evaluation of the deadband parameter p in the parameter adjustment experiments (adapted from [3]).

client to the server was kept at the full rate of $1\,\mathrm{kHz}$. Position encoder readings from the haptic device are known to be noisy. A moving average filter with a filter length of 8 was applied on the position signal at the client to denoise the signal before transmitting the position samples to the server. The proposed perception-based data reduction scheme (see Section 3.2.1) was implemented on the server. Linear extrapolation was used between two state updates to extrapolate the object movement. Linear interpolation of object states was either enabled or disabled to test its effect on the data reduction. No communication delay was emulated in the communication between the server and the client as the goal of the experiments was to investigate solely the effect of the proposed scheme on the packet rate reduction and the force feedback displayed to the user.

Even in a single-user stand-alone haptic application using the maximum stiffness that the haptic device can display and also adopting the aforementioned filter on position signals, some high-frequency noise in the rendered force feedback was perceivable, similar to the effect described by Choi *et al.* [131]. Such noise potentially compromises the collected experimental data. To counteract this problem, the object stiffness in the constraint-based force rendering algorithm was reduced to $630\,\mathrm{N/m}$. The weight of the virtual object was set to $m_o = 0.15\,\mathrm{kg}$.

Position and rotation thresholds for the visual dead reckoning were set to 0.01 (in virtual world coordinates) and $4°$, respectively. The heartbeat timeout was set to $1\,\mathrm{s}$. The interpolation time was chosen based on the heuristic using the most recent packet inter-arrival times as described in Section 3.2.3, but also clipped to a minimum of $5\,\mathrm{ms}$ and a maximum of $100\,\mathrm{ms}$.

Task (1-DOF parameter adjustment experiment) In a single run, the participant had to move the rigid cube from the start point to the end point along the x-axis (see Figure 3.9a). The object movement was artificially constrained to 1-DOF and, thus, the object could not rotate or flip. The movement of the virtual probe, representing the haptic device position in the VE, was constrained to the x-z plane to further simplify the task. The participants were instructed to slow down the movement around the middle and the end of the complete movement. Three markers (start point, mid point and end point) were drawn on the ground in the VE to give the users visual guidance. They were asked to adapt their move-

ment to these markers and to choose an acceleration they feel comfortable with. They were also advised to be consistent in their movement between different runs.

The desired trajectory enforced the users to accelerate and decelerate the object. The client extrapolates the object movement between two state updates linearly and, thus, new state updates need to be triggered by the server only if the object movement is non-linear. Only object state updates might lead to perceivable artifacts in the force feedback, as the interaction forces are rendered locally at the client.

As soon as the object reached the end point, it was automatically reset to the start point and the user had the option to adapt the parameter p or to select a reference for comparison in the next run.

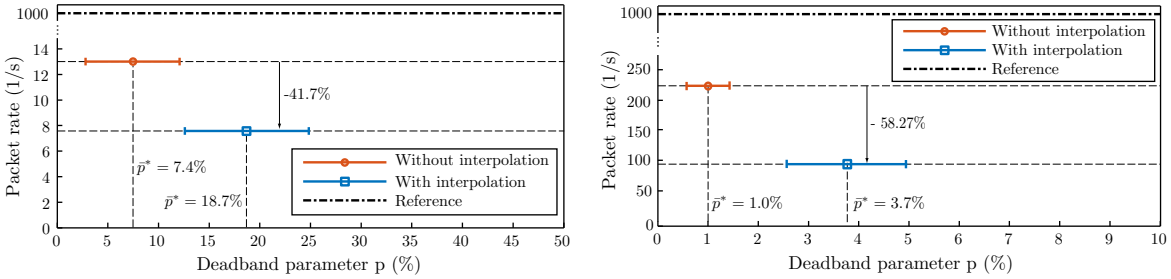
Task (6-DOF parameter adjustment experiment) The user had to rotate and flip the object in the 6-DOF experiment. All motion constraints were removed and the object could translate and rotate in 3-DOF each. The user was asked to keep in contact with the cube while rotating it to allow for force feedback quality evaluations. The markers on the ground (compare Figure 3.9a and Figure 3.9b) were removed. The participant was instructed to freely rotate the object, but to be consistent between different trials.

Procedure The aim of both experiments was to determine a suitable traffic control parameter setting for which the users perceive the haptic feedback quality to be the same as with uncompressed communication. The *method of adjustment* [61] was adopted for this purpose. Starting from an initial parameter setting, the participant could increase/decrease p in steps of 0.5 % by using the right and left arrow keys on the keyboard. The participants were instructed to find the parameter setting p^*, for which they just perceive the current setting to be the same as the reference and to stop the current trial as soon as she/he thinks this setting is found by pressing enter on the keyboard. Data reduction was disabled in the reference condition and, thus, 1000 object state updates per second were transmitted to the clients.

The mean of the final adjusted traffic control parameter (the mean of all p^* across all subjects) serves as an estimate of the suitable traffic control parameter ($\bar{p}^*$). This corresponds to a rough estimate of the point of subjective equality (PSE) [61]. The PSE is the point on the psychometric function for which the subjects perceive the reference and the stimulus to be the same.

Each experiment consisted of two different conditions (with and without object state interpolation), which were repeated three times each. The participant was forced to take a 30 seconds break between two trials. The currently active condition and parameter setting was unknown to the user to avoid any influence and bias between the trials. A distortion in the force feedback due to the data reduction was always perceivable in the beginning of a trial, meaning that the traffic control parameter p was initially set to $50\,\%$. This selection ensured that the participant was clearly aware of the difference at the beginning of a trial. Pre-tests suggested that this leads to more consistent user performance.

At any time, the reference could be selected by pressing a button on the stylus. As soon as the button had been pressed again, or the parameter p had been changed, the reference



(a) Results of the 1-DOF experiment.

(b) Results of the 6-DOF experiment.

Figure 3.10: Subjectively adjusted deadband parameter p together with the resulting packet rates for the 1-DOF and 6-DOF adjustment experiments. The horizontal error bars show the mean for all participants and the standard deviation (adapted from [3]).

was disabled, that is, data reduction was turned on again.

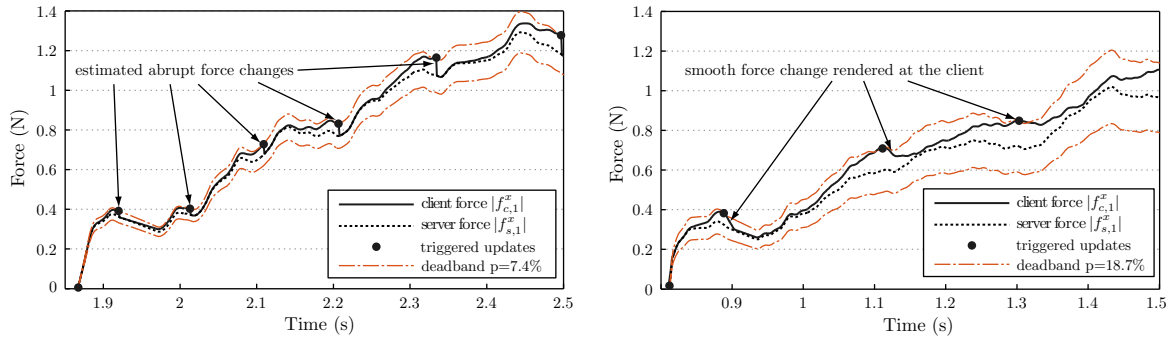
The participants wore headphones playing colored noise to avoid the influence of noise emanating from the haptic device. On average, the users tried 31.1 different settings per trial (excluding reference settings) and spent around 35 minutes for completing the 1-DOF experiment. In the 6-DOF experiment, they tried 29.9 different settings and spent 33 minutes on average.

Participants The same twelve participants (ten male, two female), with an age between 25 and 31, participated in both experiments. Four participants were frequent users of haptic devices, while the others have participated in past experiments in our lab. A training phase including proper posture and grip of the stylus was performed with each participant. The training also ensured that everyone was familiar with the experimental procedure, the haptic device and especially the tasks to be performed. The 1-DOF experiment was conducted three weeks before the 6-DOF experiment.

3.2.4.2 Results of the 1-DOF parameter adjustment experiment

The results for the parameter $\bar{p}^*$ found in the 1-DOF experiment are shown in Figure 3.10a, together with the resulting average packet rate, which was measured only while the user was in contact with the object. If there was no contact, only the Visual DR scheme (compare Section 3.2.3) triggered new state updates, not depending on the currently set parameter p. Haptic DR triggers additional updates on top of the visual updates to ensure a haptically smooth interaction with the object.

If no object state interpolation was applied, the participants adjusted the deadband parameter to an average value of $\bar{p}^* = 7.4\,\%$. As soon as the estimated force change due to the abrupt object state update exceeded $7.4\,\%$ of the intended force magnitude, a new update was triggered. In the used 1-DOF task, this led to 13 packets per second being sent on average from the server to the client. As the object movement was constrained to 1-DOF and, hence, the rendered force was always in the direction of the normal surface of the contacted triangle and object rotation was not possible, artifacts perceived by the participants were inherently related only to changes in force magnitude. A perceivable force magnitude change



(a) Without object state interpolation.

(b) With object state interpolation.

Figure 3.11: Comparison of the rendered forces without/with interpolation of object states at the client. Both forces were measured during acceleration of the cube (adapted from [3]).

of $\bar{p}^* = 7.4\,\%$ falls exactly into the range of Weber-fractions found in several psychophysical studies (compare Table 2.1).

With linear interpolation enabled at the client, the participants perceived the current setting and the reference to be the same for a deadband parameter of $\bar{p}^* = 18.7\,\%$. This reduced the packet rate on average by $41.7\,\%$ to 7.5 object state updates per second, when compared to the case without interpolation. The object state interpolation obviously allowed for the use of an increased traffic control parameter p. Higher force differences could be tolerated, which were then corrected gradually. This is in line with the experimental data reported in [218], where it was found that the JND for kinesthetic forces decreases the slower the force magnitude changes to a new value.

To illustrate this, Figure 3.11 shows exemplary force signals during the initial acceleration of the object for both found deadband configurations. The abrupt force changes at the time instances when an update arrives are clearly visible in the rendered client force (Figure 3.11a). Contrarily, these jumps are avoided and the rendered force is changing gradually if object state interpolation is used (Figure 3.11b).

3.2.4.3 Results of the 6-DOF parameter adjustment experiment

The results of the 6-DOF parameter adjustment experiment, shown in Figure 3.10b, are quite different compared to the settings found in the 1-DOF experiment. If no interpolation was used, the users adjusted p on average to $\bar{p}^* = 1\%$. Including object state interpolation, the adjusted parameter was $\bar{p}^* = 3.7\,\%$. Both parameter settings are much lower than the ones found in the 1-DOF experiment. Object state interpolation reduced the average packet rate from 230 to 96 updates per second (reduction of 58 %). Note that the packet rate was again measured only while the user was in contact with the object.

3.2.4.4 Statistical analysis

The collected data is normally distributed according to the D'Agostino-Pearson omnibus K2 normality test. Hence, the data was further analyzed with parametric tests. A repeated measures two-way analysis of variance (ANOVA) [219] was used (significance level of 5%).

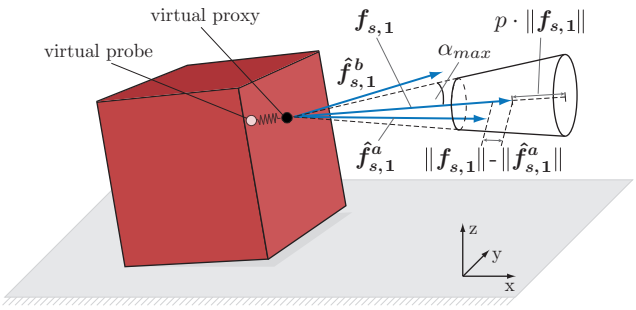


Figure 3.12: Illustration of the non-isotropic deadzone. The force vector $\hat{\boldsymbol{f}}^{\boldsymbol{a}}_{\boldsymbol{s,1}}$ falls within the truncated cone defined by the reference force $\boldsymbol{f_{s,1}}$, the deadband parameter p and α_{max} and, thus, no update is triggered. Contrarily, $\hat{\boldsymbol{f}}^{\boldsymbol{b}}_{\boldsymbol{s,1}}$ does not comply with the deadzone and a new object state update is triggered (adapted from [3]).

A significant effect of interpolation (F(1,11)=199.08, p-value<0.001) and task (F(1,11)=45.30, p-value<0.001) can be reported. Additionally, there is a significant interaction between the interpolation method and the task (F(1,11)=44.09, p-value<0.001). Post-hoc analysis using Bonferroni correction indicates that statistically higher values for p^* were observed in the translation task than in the rotation task (p-value<0.001). Statistically lower values of p^* were found in both tasks, compared to the cases with interpolation, if no interpolation was used (p-value<0.001).

3.2.4.5 Discussion

The results of the 1-DOF and the 6-DOF adjustment experiments show that object state interpolation helps to hide the force changes caused by object state updates. As a result, it allows for further packet rate reduction. The found traffic control parameter for the 1-DOF and the 6-DOF task, for which the participants perceive the reference and the current setting to be the same, are, however, quite different.

This can be attributed to the following reasons. First of all, the 1-DOF task includes only object position changes in a single direction, without any rotational changes. The linear prediction at the clients performs better in this simplified task. As a consequence, fewer object state updates are needed and, thus, more time is available to smoothly correct the object state. Additionally, the many object state updates in the 6-DOF experiment, each of them correcting only a small error, lead to high-frequency noise in the rendered force feedback. This is especially evident for the rotation task without object state interpolation. In the direct comparison to the reference, which feels very smooth, this noise is perceivable. As a consequence, the participants further decrease the traffic control parameter p to remove this noise. In other words, low traffic control parameters are needed to fully match the perception to the reference. On the other hand, this perceivable high-frequency noise might not be disturbing to the user. This motivates further evaluation, presented in the following Section 3.2.5, where the degradation of force feedback quality, rather than perceptual equivalence is investigated.

The results of the 1-DOF and 6-DOF experiments also motivate to reshape the isotropic deadzone explained in Section 3.2.1. Object state interpolation and a traffic control parameter

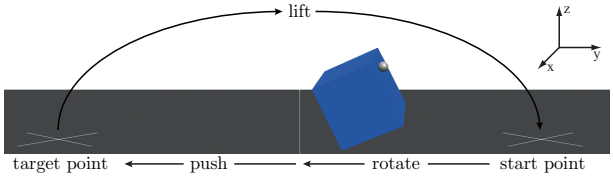


Figure 3.13: VE used for the force feedback quality experiment. The task consists of rotating, pushing and lifting the object (adapted from [3]).

of $3.7\,\%$ lead to a performance loss if the object is translated linearly, rather than rotated, as more updates are triggered than necessary. On the other hand, using $p = 18.7\,\%$ could lead to unsatisfying force feedback quality, if the object is rotated. According to Equation (3.3), $p = 18.7\,\%$ corresponds to a maximal directional change of $10.8°$. The parameter setting $p = 3.7\,\%$, however, defines a maximal directional change of $2.1°$. Instead of using just a single parameter p, which defines the maximal force magnitude change as well as directional change, the change in magnitude and direction can be considered individually.

The resultant deadzone has the shape of a truncated cone, as introduced by Kammerl *et al.* in [69] and illustrated in Figure 3.12. The reference force $\boldsymbol{f_{s,1}}$, together with the deadband parameter p and the angle α_{max} define the volume of the deadzone. The parameter p is now used to compare only the change in force magnitude (contrary to force magnitude and direction as for the isotropic deadzone). The angle between the two forces is compared to α_{max} and if one of the two conditions is violated, a new update is triggered. Subjective evaluation is needed to investigate the usefulness of the reshaped deadzone. For this reason, both the isotropic and the non-isotropic deadzones are included in the force feedback quality experiment, which is explained in the following.

3.2.5 Subjective evaluation of the haptic force feedback quality

The results of the previous experiments motivate the evaluation of the force feedback quality degradation depending on different traffic control parameter settings. For this purpose, another experiment, denoted as the *force feedback quality experiment*, has been performed.

3.2.5.1 Method

Apparatus This experiment used the same apparatus as the previously explained 1-DOF and 6-DOF experiments (see Section 3.2.4.1).

Task The task involved rotating, pushing and lifting a rigid cube as illustrated in Figure 3.13. At the beginning of a trial, the cube was positioned at the start point. The user had to move the cube approximately towards the center, denoted by the line on the ground, by mainly rotating it. From here on, the cube had to be pushed until the target point was reached. Lastly, the user lifted the cube back to the starting position. This was done by touching the surface of the cube and pressing a button on the stylus of the haptic device. A soft spring (stiffness of $68\,\mathrm{N/m}$) between the virtual probe and the surface contact point was modeled to render interaction forces during lifting. The task was finished as soon as the cube

Rating	Description
7	No difference to reference stimulus S_{R7}
6	Not disturbing
5	Slightly disturbing
4	Moderately disturbing
3	Disturbing
2	Strongly disturbing
1	No difference to reference stimulus S_{R1}

Table 3.2: Applied user rating scheme in the force feedback quality experiment (adopted from [3]).

was returned to the starting point. The movement of both the object and the virtual probe were not restricted.

Procedure The user was asked to perform the task and to rate the force feedback quality according to the rating scheme shown in Table 3.2, where 7 corresponds to the best force feedback quality and 1 to the lowest quality, also represented by two references (denoted as S_{R7} and S_{R1} as explained below). If a difference to the reference stimuli S_{R7} or S_{R1} was perceived, the subject was also asked to rate how disturbing this difference, with respect to S_{R7}, was.

A training phase was conducted with each participant to ensure proper knowledge of the task. The task was first demonstrated by the experimenter, before the participants practiced with the reference S_{R7} until they could handle the task properly. Then, the references S_{R7} and S_{R1} were twice shown alternately. The participants were also instructed to hold the stylus of the haptic device tightly and similar to a pen.

The tested traffic control parameter settings were presented sequentially to the participant, but their order was randomized. After finishing the task, the participant had the option to give a rating, to try the references again, or to repeat the current setting. As soon as the current setting had been rated, the experiment continued with the next setting. Ratings and commands were given orally to the experimenter, who controlled the experimental procedure. There was a break of 30 seconds after all settings had been rated. In a second round, all settings were presented again in the same order. The gathered knowledge about the tested settings now helped the user to rate them more precisely. The rating, given in the first round, was shown to the user and he/she had the option to re-calibrate and change the ratings if necessary. Only the adjusted ratings given in the second round were used as results. The participants wore headphones playing colored noise to avoid the influence of noise emanating from the haptic device.

Stimuli Twelve different traffic control settings, listed in Table 3.3, were investigated. Nine settings used the proposed perception-based data reduction scheme including Haptic DR and Visual DR. The VDR position and rotation error thresholds (denoted as Δx and Δr) were tuned in a pre-test so that the object movement was perceived as visually smooth during the

Stimulus name	HDR	VDR (Δx,Δr)	Interpolation
S_{R7}	No traffic control, i.e., 1000 object state updates per second		
S_{R1}	-	$(0.006, 2°)$	no
S_4^{NI}, S_8^{NI}, S_{18}^{NI}, S_{30}^{NI}	$p = \{4\,\%, 8\,\%, 18\,\%, 30\,\%\}$	$(0.006, 2°)$	no
S_4^{WI}, S_8^{WI}, S_{18}^{WI}, S_{30}^{WI}	$p = \{4\,\%, 8\,\%, 18\,\%, 30\,\%\}$	$(0.010, 4°)$	yes
S_{NIso}	$p = 18\,\%$, $\alpha_{max} = 2.1°$	$(0.010, 4°)$	yes
S_{DR}	$(0.0015, 2°)$	$(0.010, 4°)$	yes

Table 3.3: The twelve dead reckoning settings tested in the force feedback quality experiment (adopted from [3]).

task. Settings adopting object state interpolation can use higher VDR thresholds compared to settings without interpolation.

Eight of the nine perception-based settings used the isotropic deadzone defined by the deadband parameter p, with p varying between 4 % and 30 %. Object state interpolation was either disabled (S_p^{NI} in Table 3.3) or enabled (S_p^{WI} in Table 3.3).

The non-isotropic setting S_{NIso} used the reshaped deadzone as explained in Section 3.2.4.5. The two parameters p and α_{max} were chosen based on the results presented in Section 3.2.4.2 and Section 3.2.4.3 for the conditions adopting object state interpolation.

State-of-the-art dead reckoning as suggested by the IEEE Standard for Distributed Simulations [148] was also tested (S_{DR} in Table 3.3). The setting S_{DR} also adopted object state interpolation and used different thresholds for the visual and haptic modality, but all of them were fixed. The fixed haptic thresholds were tuned in pre-tests to achieve a force feedback quality, which is comparable to the non-isotropic S_{NIso} setting.

Additionally, there were two references (S_{R7} and S_{R1} in Table 3.3), representing the highest and lowest ratings in the scale shown in Table 3.2. S_{R7} did not apply any data reduction and, thus, 1000 object state updates per second were transmitted from the server to the clients. This corresponded to the best achievable force feedback quality. S_{R7} was equivalent to the reference in the previous 1-DOF and 6-DOF parameter adjustment experiments. S_{R1} triggered only visual updates with fixed thresholds, neglecting additional haptic updates, and did not use object state interpolation.

Participants Twelve subjects participated in this experiment (10 male, 2 female, aged between 25 and 32). Five of them were frequent users of haptic devices, the others already participated in former experiments in our lab. Eight of the subjects participated also in the previous 1-DOF and 6-DOF experiment.

3.2.5.2 Results and statistical analysis

The user ratings and the resulting packet rates for all twelve tested traffic control settings are shown in Figure 3.14. Packet rates were measured during the complete task. This is in contrast to the 1-DOF and 6-DOF adjustment experiments, where the packet rate was measured only while the user was in touch with the object. User ratings and packet

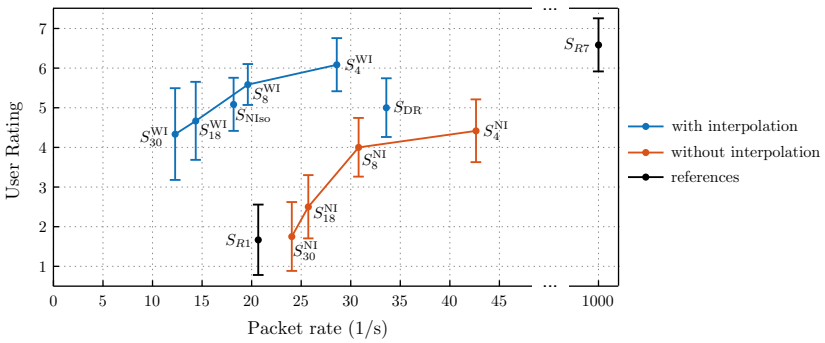


Figure 3.14: Resulting packet rates and user ratings for the tested settings. The error bars show the mean and standard deviation between participants. The blue color denotes settings wherein object state interpolation was enabled. Red refers to settings where it was disabled. Shown in black (S_{R1} and S_{R7}) are the references (adapted from [3]).

rates were averaged between participants. The result of the non-parametric Kruskal-Wallis test [220] (significance level of 5 %) indicates a significant difference in the user ratings ($\chi^2(11) = 110.57, p - value < 0.001$). Post-hoc statistical analysis using Bonferroni correction was used to determine significantly different settings.

The tested settings without object state interpolation (S_4^{NI}, S_8^{NI}, S_{18}^{NI}, S_{30}^{NI}) perform worse than the other stimuli using the proposed data reduction scheme. Even a low deadband parameter of $p = 4\,\%$, triggering 42.6 state update packets per second on average, was not enough to achieve a force feedback quality close to the reference S_{R7}. This conforms to the results presented in the 6-DOF parameter adjustment experiment (Section 3.2.4.3), where a deadband parameter $\bar{p}^* = 1\,\%$ was found necessary to achieve subjective equality to S_{R7}. Statistical analysis also indicated a significant difference between S_4^{NI} and S_{R7}.

The average rating for S_{30}^{NI} was very close to the reference S_{R1} (no statistically significant difference). Only few additional updates, compared to S_{R1} without HDR, were triggered with the high deadband parameter $p = 30\,\%$. Therefore, the force feedback quality was not improved substantially. Force feedback quality ratings were low in this setting, although on average at least nearly 25 updates per second were transmitted to the client. It is generally known that around 25 to 30 frames per second are needed to display a visually convincing scene. These results show that this rate is not enough for SHVEs. This also emphasizes the usefulness of the multi-modal decision process in the proposed traffic control scheme, which considers the visual and the haptic modality separately.

The proposed traffic control scheme including object state interpolation (S_4^{WI}, S_8^{WI}, S_{18}^{WI}, S_{30}^{WI}) was consistently rated higher and also reduced the average packet rate compared to the settings without interpolation. S_{30}^{WI} and S_4^{NI} were rated comparably (no significant difference), but there were only 12.3 updates per second triggered for S_{30}^{WI} compared to 42.6 for S_4^{NI} (packet rate reduction of 71.13 %).

The best average user rating was achieved with S_4^{WI}, which resulted on average in 28.6 update packets per second. Compared to the reference S_{R7}, this corresponds to a decrease of 97.14 %. Statistical analysis indicates no significant difference in the user ratings to the reference S_{R7}.

The comparison to the S_{DR} setting shows that the proposed traffic control scheme, taking characteristics of the VE, the haptic device, and the limitations of the human haptic perception system into account, outperforms traditional dead reckoning. The average user rating for S_{DR} was five, while the resulting packet rate was 33.6 packets per second. Compared to the non-isotropic S_{NIso} setting, which was rated comparably (no statistically significant difference), the packet rate increased by 84.6 %.

3.2.5.3 Discussion

The degradation in user ratings is less steep for the settings including object state interpolation (compare the lines connecting the mean ratings in Figure 3.14). This indicates that the artifacts, introduced by the object state updates for high traffic control parameters, were less disturbing if object state interpolation was used. The packet rate can be further reduced, while at the same time the force feedback is less distorted.

The non-isotropic deadzone did not reduce the packet rate significantly, while, at the same time, achieving convincing force feedback quality. The reason might be the relatively short pushing phase during the task. As more updates are generally needed during the rotation in the beginning (see Section 3.2.4.3), the gain achieved during pushing did not lead to a big decrease in the packet rate averaged over the complete task. Additionally, the parameters of the non-isotropic deadzone were determined in separate experiments, where force magnitude and direction changes were decoupled. The object movement in the beginning of the task, however, included rotation and translation at the same time. This means that using $p = 18\,\%$ (found in Section 3.2.4.2) and $\alpha_{max} = 2.1^\circ$ (found in Section 3.2.4.3) was not necessarily the optimal choice.

The evaluation has shown that the proposed data reduction scheme outperforms standard dead reckoning, as a higher force feedback quality is achieved with fewer packets being transmitted from the server to the clients. Even S_4^{WI}, which was rated best among all tested traffic control settings, triggered less object state updates than the S_{DR} setting. Remember that the S_{DR} setting used different, but fixed thresholds for the visual and haptic modalities and, hence, goes beyond the standard defined in [148], which uses only visual thresholds. If a single threshold pair is used without distinguishing between the visual and haptic modalities, the more precise modality dominates the threshold selection. Smaller values need to be chosen and the performance decreases. To cope with this, the proposed dead reckoning scheme uses separate thresholds and a mathematical model of the human perceptual limitations to adapt the haptic error thresholds to the current interaction. This is especially evident during lifting of the object, when higher error thresholds can be allowed due to the small stiffness of the spring. The state-of-the-art dead reckoning algorithms do not automatically adapt to such scenarios.

In summary, the results show that proposed scheme achieves a packet rate reduction of 97 % in the presented experiments without deteriorating the haptic feedback quality significantly. If some small disturbing artifacts in force feedback are accepted, the update rate can be further reduced. The achievable packet rate reduction also depends on the task and the user interactions involved. Obviously, if the object state prediction is less accu-

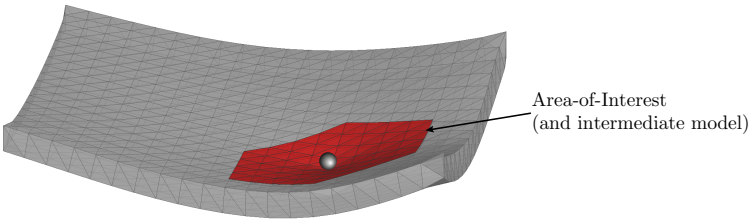


Figure 3.15: Exemplary deformable object polygon mesh consisting of 714 vertices and 1424 triangles. The red area defines the Area-of-Interest (AOI) during the haptic interaction. The vertices within the AOI also describe the intermediate model, which is used in the multi-rate haptic rendering algorithm. These vertices are prioritized during the compression of the deformation, as they are deemed to influence directly the haptic force feedback quality (adapted from [8], © 2016 IEEE).

rate, for example, due to interactions that lead to fast-changing and unpredictable object position/rotations, the performance reduces.

The perceptibility of artifacts in the force signal also depends on the way the signal is presented to the user. The haptic device and its capabilities, for example, its force resolution or inherent physical damping, the way the user interacts with the device, or the part of the body that is stimulated are factors that might lead to different parameters as found in the experiments presented in this chapter.

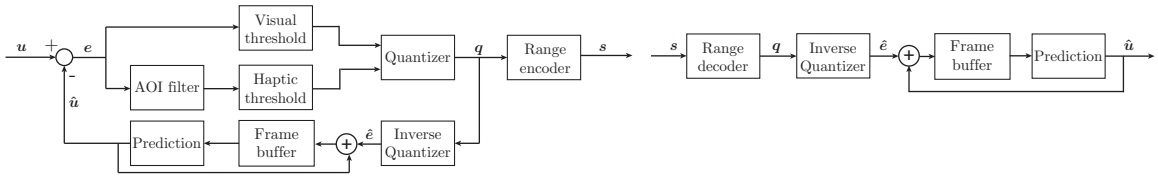
3.3 Perception-based data reduction for the interaction with deformable objects

The previously discussed data reduction scheme has been shown to reduce the number of packets transmitted from the server to the clients during the interaction with rigid objects. The transmission of deformable object states differs significantly. Large payload and lower packet rates characterize the traffic during the interaction with deformable objects in the CS-State architecture (see Section 3.1). As a result, the compression of polygon mesh deformation data, that is, the payload, becomes necessary to reduce the required transmission capacity.

As an example, consider the deformable object shown in Figure 3.15. It consists of 714 vertices and 1424 triangles. The FEM simulation running on the server calculates the displacement of each vertex $\boldsymbol{u_i}$ ($i = 1, \ldots, N = 714$), summarized in the vector $\boldsymbol{u} \in \mathbb{R}^{3N}$, at a temporal update rate of about 60 Hz. The uncompressed transmission of the deformation data requires 8568 bytes payload per packet[2], leading to a bit rate of 4.1 Mbit/s.

Existing compression schemes for the transmission of such polygon mesh deformation data are tailored for vision-only applications (see Section 2.4.3.3). Parameters are fine-tuned to visually hide the introduced distortion from the user or to find the best-possible trade-off between visual quality and achieved compression, that is, bit rate. The human visual and haptic sensory modalities, however, are known to differ in their spatial and temporal resolution capabilities. Also, the more precise modality typically dominates during the integration of different sensory information [47].

[2] Each vertex coordinate is considered as a 4 byte floating point number.



(a) Encoder. (b) Decoder.

Figure 3.16: Schematic block diagram of the lossy encoder/decoder structure for the compression of 3D polygon mesh deformation data. Signals marked with a hat ($\hat{\cdot}$) denote signals distorted due to the applied thresholds and quantization (adapted from [8], © 2016 IEEE).

For these reasons, a compression scheme for the efficient transmission of polygon mesh deformation data in visual-haptic applications should consider the visual and haptic modalities individually, similarly to the previously discussed data reduction scheme for rigid object states. The following section presents a low-delay compression scheme for 3D polygon mesh deformation data that follows this paradigm.

3.3.1 Perception-based compression of polygon mesh deformation data

Figure 3.16 shows the encoder and the decoder structure of the proposed perception-based low-delay compression scheme for 3D polygon mesh deformation data. It consists of the common building blocks of most state-of-the-art mesh compression algorithms: prediction, quantization and entropy coding.

The main difference compared to the state-of-the-art is the pre-selection of vertices based on the *Area-of-Interest* (AOI), which describes the area around the current haptic user interaction as illustrated in Figure 3.15. The vertices within the AOI form the intermediate model mesh (see Section 2.3.4.2) and are used in the high-rate force rendering at the client. For this reason, the deformation updates of these vertices are deemed specifically important for the force feedback quality.

The encoder structure shown in Figure 3.16 allows for the individual compression of vertex deformation updates by applying different mechanisms and parameters based on their influence on the haptic or the visual modality. In other words, coding noise can be introduced in areas of the mesh where the distortion is barely perceivable or even below haptic or visual detection thresholds.

The following section introduces the building blocks of the encoder and the decoder. Their general structure is similar to the algorithms proposed in [206], [207], which are suited for real-time compression of polygon mesh deformation data, but do not consider the visual and haptic modalities individually.

3.3.1.1 Encoder structure

Prediction The *prediction block* performs the prediction of vertex displacements to exploit the correlation between adjacent updates. In this thesis, a zero-order time-only predictor is used due to its simplicity. The zero-order time-only predictor adds the most recently received displacement error of a vertex i (received at time step t_l), $\hat{\boldsymbol{e}}_{\boldsymbol{i}}[t_l]$, to the last displace-

ment, $\hat{\boldsymbol{u}}_{\boldsymbol{i}}[t_{l-1}]$, and holds this value until the next update arrives. Hence, the displacement at time step t_n is

$$\hat{\boldsymbol{u}}_{\boldsymbol{i}}[t_n] = \hat{\boldsymbol{u}}_{\boldsymbol{i}}[t_{l-1}] + \hat{\boldsymbol{e}}_{\boldsymbol{i}}[t_l] \tag{3.9}$$

Note that any other prediction scheme, for example, time-only, space-only or time-space predictors [206], can be adopted instead. More-sophisticated (and computationally more expensive) time-space predictors promise an increased compression performance than the zero-order time-only predictor, especially if the prediction errors are quantized with a high resolution. If the prediction errors are quantized to a small number of bits as in the prototype implemented for this thesis, the gain due to better prediction is small [206].

AOI filter The *AOI filter* selects the vertices from the polygon mesh that lie within the AOI and, hence, are close to the current user interaction. The AOI is redefined at the rate of the deformation simulation. At each time step, the triangle that is closest to the virtual proxy denotes the center of the AOI. The triangles within a neighborhood of m triangles around this center build the intermediate model and also denote the AOI. The list of vertex indices that describe these triangles are prestored for all possible center triangles to efficiently implement this AOI filter. The vertices inside the AOI are then processed individually as described in the following.

Visual and haptic thresholds A *visual threshold* and a *haptic threshold* are introduced to further reduce the number of vertex deformation updates to be included in the next packet. An update transmitted to the client(s) includes a vertex only if the prediction error for this vertex exceeds a certain threshold. The motivation behind this approach is that small displacement errors at the client, where the same prediction is performed, are not perceivable or disturbing. This technique is known as *dead reckoning* and is similarly used to reduce the number of object state updates for rigid objects in the considered CS-State architecture (see Section 2.4.3.2 and Section 3.2). The AOI and these thresholds distinguish the proposed scheme from common mesh compression approaches (compare Section 2.4.3.3).

The predicted position error of vertex i at time step t_n is compared to a fixed and constant visual threshold v:

$$\text{if}: \quad \|\boldsymbol{e}_{\boldsymbol{i}}[t_n]\| \geq v, \quad \text{then}: \text{include vertice} \tag{3.10}$$

The compression scheme additionally compares the predicted position error of vertex i, if it is included in the AOI, to a variable haptic threshold $h[t_n]$:

$$\text{if}: \quad \|\boldsymbol{e}_{\boldsymbol{i}}[t_n]\| \geq h[t_n], \quad \text{then}: \text{include vertice} \tag{3.11}$$

The variable haptic threshold is calculated at each time step using Hooke's Law as

$$h[t_n] = p\frac{\|\boldsymbol{f}_{\boldsymbol{s}}[t_n]\|}{k_c} \tag{3.12}$$

The vector $\boldsymbol{f}_{\boldsymbol{s}}[t_n]$ describes the force rendered at the server. The stiffness k_c is the stiffness that is used in the penalty-based force rendering algorithm at the client. The deadband parameter p is inspired by the Weber-fraction c, which describes the limitation of human force perception.

The force rendering at the client is based on the penetration depth of the haptic device into the polygon mesh. The vertex position change also leads to a change in the displayed force. The threshold $h[t_n]$ is an approximation of the vertex displacement that leads to a perceivable force change if the deadband parameter p is chosen to be in the range of the experimentally found Weber-fraction for force magnitudes.

The proposed filtering of vertex updates necessitates the transmission of a list of vertex indices included in each update. This leads to additional overhead, which can be reduced by applying differential coding and entropy coding to the sorted index list.

Quantization The prediction and filtering steps result in a reduced list of vertices to be included in the next deformation update. The displacement errors of these vertices are uniformely quantized with a resolution of q bits (2^q quantization levels). This leads to positive integer representations of the deformation data and reduces the number of bits per vertex from 96 (assuming 4 byte floating point numbers for the x, y, z coordinates) to $3q$. Ibarria *et al.* [206] state that the subsequent entropy coding is especially effective if the deformation data is quantized to a small number of bits, typically between 8 and 12. The implementation evaluated in the following uses a resolution of 8 bits, because pre-tests suggested that the introduced quantization noise is sufficiently small to be not perceptible.

The quantization range is adapted to the differential deformation data included in each update. This ensures that the uniform quantization with the 2^q levels well represents the processed data, but requires the transmission of the quantization range along with the quantized data to allow for a correct reconstruction at the receiver.

Entropy coding The last stage of the encoding process is to apply lossless entropy coding on both the differentially encoded vertex index list and the quantized displacement data. Specifically, a so-called range coder [221], which is an integer-arithmetic variant of a general arithmetic coder, is used for speed considerations. Such a range coder has been adopted successfully, for example, also for real-time point cloud compression [222].

The statistical model in the range coder is adaptive, that is, no probability table for the quantized vertex displacements is calculated for each update packet in advance. Instead, the probabilities are adapted during the encoding/decoding process. This approach does not require the transmission of the probability table, which constitutes a significant overhead in the considered application, but the performance is slightly worse when compared to a static encoder/decoder using the exact probabilities [223].

3.3.1.2 Decoder structure

The decoder first decodes the received bitstream, then inversely quantizes the deformation data and finally applies the same prediction as the encoder (see Figure 3.16b). The decoded displacement $\hat{\boldsymbol{u}}[t_n]$ is not identical to the input $\boldsymbol{u}[t_n]$, because the thresholds and the quantizer introduce coding noise. Hence, the proposed encoder/decoder structure is lossy. Note that the encoder includes the decoder structure to calculate the displacement error $\boldsymbol{e}[t_n]$ with respect to the decoded displacement at the client to avoid error propagation.

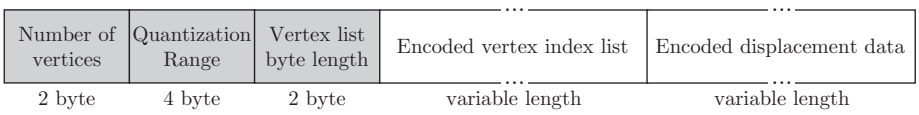


Figure 3.17: Packet format of a deformation update transmitted to the clients.

3.3.1.3 Packet structure

The deformation data and the corresponding vertex index list can be packetized and transmitted individually for the AOI and the remaining polygon mesh, that is, there are separate visual and haptic update packets. Alternatively, all selected vertex updates are included in a single update. The individual transmission has the advantage that the haptic packets are smaller and can be handled more efficiently by the network, for example, by prioritizing haptic updates. This comes, of course, at the cost of an increased packet rate.

The data structure in Figure 3.17 represents the packet format in both cases. The filtering of vertices based on visual-haptic thresholds requires the transmission of the number of vertices that are included in the packet and, of course, also a list of vertex indices so that the receiver can reconstruct the displacement data properly. The vertex list byte length field is necessary to point to the end of the variable-length encoded vertex index list in each packet.

3.3.2 Implementation

In the following, the implementation used for the evaluation of the proposed compression scheme is introduced.

The CS-State architecture for the remote haptic interaction with deformable object was implemented on top of Chai3D [224]. The Geomagic Touch haptic [79] device was used to display force feedback to the users, which was calculated at the clients with a penalty-based force rendering algorithm (see Section 2.3.4) at a rate of 1 kHz using intermediate models (see Figure 3.15). The intermediate models consisted of exact cutouts of the original mesh instead of simple geometric shapes, for example, planes, to avoid the rendering problems explained in Section 2.3.4.2. The intermediate model was updated at the client once a new deformation update has been received. To mitigate jump effects due to sudden vertex position updates, vertex positions were corrected smoothly by applying linear interpolation.

Collision trees based on Axis-Aligned Bounding Boxes (see Section 2.3.3) were used to detect collisions between the virtual probe and the intermediate model, and to find the closest vertex to the virtual probe. They were continuously updated during the interaction (see Section 2.3.3). Note that the tree update and the force rendering at the clients require a temporal update rate of 1 kHz to achieve a stable coupling between the haptic device and the VE (see Section 2.2.2). The intermediate model, which represents only a small subset of the complete polygon mesh, reduced the complexity of the update procedure and allowed to achieve this rate.

The FEM deformable object simulation on the server used the co-rotational linear elasticity model [113] (see Section 2.3.2.2) implemented by the VegaFEM library [225]. The membrane-like object with inhomogeneous material properties, which was used in the eval-

uation, consisted of 714 vertices, 1424 triangles and 1920 tetrahedral simulation elements (see Figure 3.15). The simulation update rate on the server (quad-core i7 CPU, 8GB RAM) was about 60 Hz. Forces applied by the clients were rendered on the server at the rate of the FEM simulation. The reduced temporal update rate, when compared to the haptic rendering at the client, is possible because there is no physical haptic device connected directly to server.

A single client connected to the server has been considered for the evaluation of the proposed compression scheme with objective and subjective tests, as the compression performance is independent of the number of clients. The server implemented the proposed compression scheme to transmit the object deformation data back to the client. The client transmitted its device position to the server at the full rate of 1 kHz. The server and the client were running on different workstations, which were connected over a local network. The communication delay between the two machines was negligible.

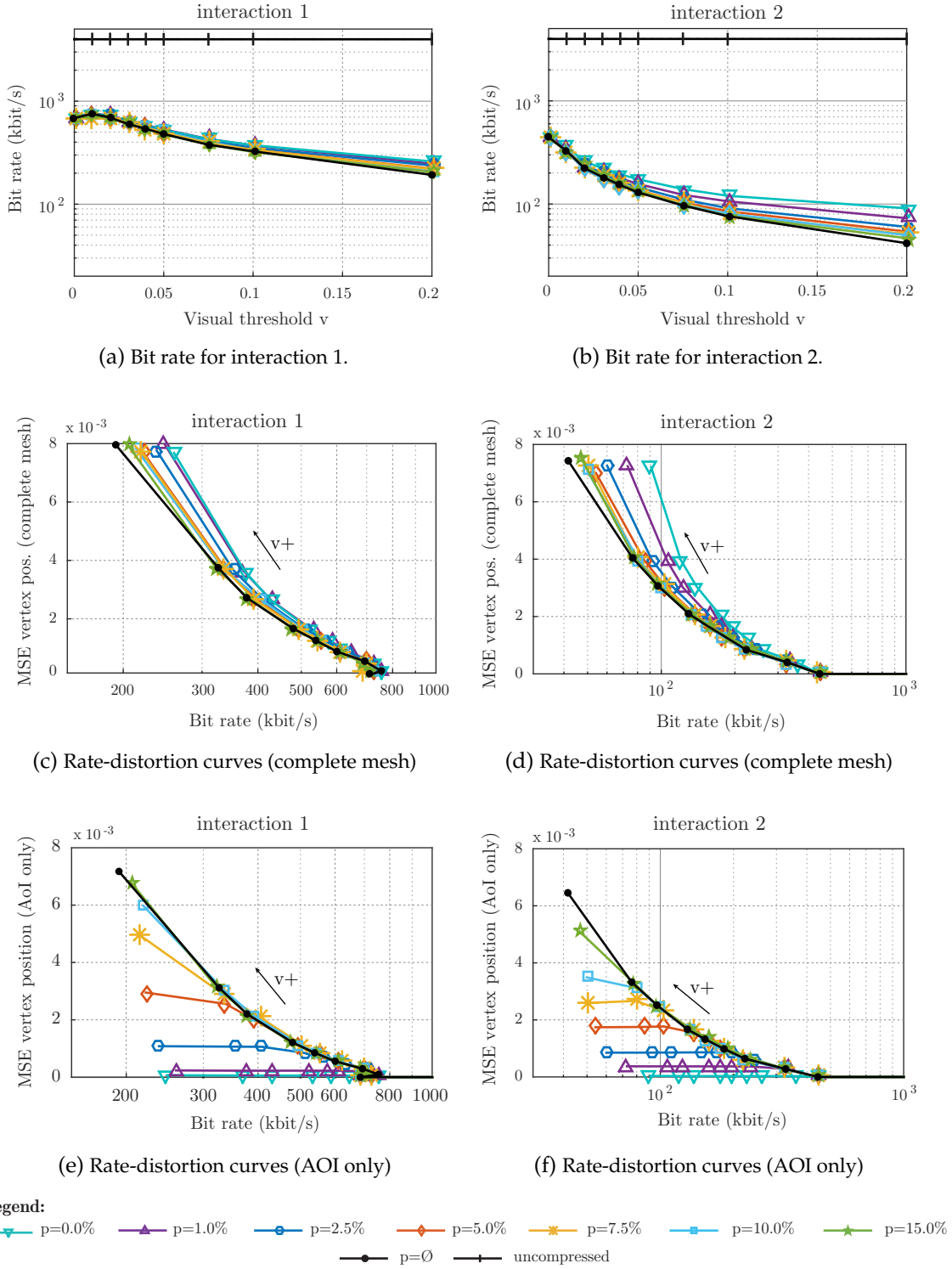
3.3.3 Objective evaluation of the compression performance

Procedure Two different user interactions, that is, haptic device position trajectories, were recorded at the client while the compression scheme was turned off. The first interaction involved larger object deformations compared to the second interaction. The prerecorded device trajectories were replayed repeatedly and transmitted to the server during the objective evaluation. The server varied the coding parameters v and p and recorded the resulting bit rates and the distortion (position error) introduced to the polygon mesh in terms of mean squared error (MSE). Note that v is an absolute threshold measured in virtual position units, while the deadband parameter p defines the varying haptic threshold h as given in Equation (3.12).

Results Figure 3.18 summarizes the objective evaluation results. About 4.1 Mbit/s are required to transmit the polygon mesh deformation data to the client if no compression is applied (see Figure 3.18a and Figure 3.18b). The average compression performance depends on the user interaction. More vertex updates need to be transmitted if the interaction leads to larger object deformations (compare Figure 3.18a and Figure 3.18b).

If no thresholds are applied (v=0, p=0 %), that is, all 714 vertex displacements are transmitted to the client after each simulation step, quantization and entropy coding reduces the bit rate to 685 kbit/s on average for interaction 1 (compression ratio 5.8:1) and to 444 kbit/s on average for interaction 2 (compression ratio 9:1). The difference can be explained by the lower entropy of the deformation data for interaction 2 compared to interaction 1. Interaction 2 involved less deformations and, hence, the displacement for many vertices was zero and quantized to the same level.

The proposed filtering of vertex updates further reduces the bit rate. One can observe from Figure 3.18a and Figure 3.18b that the visual threshold parameter v is the main factor influencing the bit rate. This is intuitive as the haptic threshold h is applied only within the AOI but the visual threshold v on the complete mesh. Only for increasing values of v, combined with low values of p, the additional bit rate for the vertex updates within the AOI updates becomes significant. For interaction 2 and $v = 0.2$, for example, the transmis-



(a) Bit rate for interaction 1.

(b) Bit rate for interaction 2.

(c) Rate-distortion curves (complete mesh)

(d) Rate-distortion curves (complete mesh)

(e) Rate-distortion curves (AOI only)

(f) Rate-distortion curves (AOI only)

Figure 3.18: Compression performance for varying coding parameters. (a) and (b): The resulting bit rate for the transmission of a test polygon mesh (714 vertices, see Figure 3.15) as a function of v and p and two different user interactions. (c) and (d): Rate-distortion curves for the two interactions with the vertex position MSE shown for the complete polygon mesh. (e) and (f): Rate-distortion curves with the MSE shown only for the haptic AOI (adopted from [8], © 2016 IEEE).

sion requires 42 kbit/s on average if no additional haptic updates are triggered. The settings wherein no additional haptic updates are triggered, are denoted as $p = \varnothing$ in the following. If all vertices within the AOI are always transmitted to the client ($p = 0\,\%$), the resulting bit rate is 91 kbit/s on average.

The average bit rate for interaction 1 and $v = 0.01$ slightly increases compared to $v = 0.0$ (see Figure 3.18a). The required list of vertex indices that are included in the current packet (see Section 3.3.1.3) sacrifices the gain due to only a few vertices being excluded from the update. In this case, the overhead could be reduced by sending a list of vertex indices that are not included in the current update.

One can observe from Figure 3.18c and Figure 3.18d that v is also the main parameter influencing the introduced mesh distortion, measured as the MSE of the vertex positions. The bit rate increases for small values of p, but the MSE does not decrease significantly. The reason is again that p influences only a small subset of the complete polygon mesh. In fact, the MSE introduced into the complete mesh is nearly independent of p. Note that all curves in Figure 3.18c and Figure 3.18d end at the same maximum bit rate for $v = 0$, independent of the parameter p. The reason is that the parameter setting of $v = 0$ updates all 714 vertices in each iteration. Hence, only quantization and entropy coding reduces the bit rate in these cases.

The effect of p is observable in Figure 3.18e and Figure 3.18f. The distortion within the AOI increases with increasing values of p, independent of v. The two separate thresholds enable the individual control of the coding noise, that is, the vertex position error, which the compression scheme introduces into the AOI and the rest of the mesh. Hence, the visual and haptic feedback quality can be controlled individually with the parameters v and p.

3.3.4 Subjective evaluation of the visual-haptic feedback quality

The objective evaluation results do not allow to draw conclusions about the visual and haptic feedback quality, which is achieved for different coding parameters. The subjective test described in the following was performed to fill this gap and to find suitable values for the traffic control parameters v and p.

Apparatus The CS-State architecture was implemented as described in Section 3.3.2. The membrane-like deformable object (compare Figure 3.15) was simulated based on the co-rotational linear elasticity model [113] with inhomogeneous material properties. The parameters of the linear elasticity model were set to a mass-density of $10\,000\,\mathrm{kg/m^3}$ and a Poisson's ratio of 0.45. Young's modulus was set to $10\,\mathrm{kN/m^2}$ in the left half and $2.5\,\mathrm{kN/m^2}$ in the right half of the object. Hence, the object was softer in the right half than in the left half. The positions of the eight vertices at the corners of the membrane were fixed during the simulations, that is, they denoted boundary conditions in the FEM simulation. The stiffness of the intermediate model used in the force rendering at the clients was set to $500\,\mathrm{N/m}$.

Task In each trial, the task for the subject was to circularly interact with the membrane-like deformable object and to rate the perceived visual and haptic quality on a seven-point rating

scale (1 to 7). A rating of 7 corresponded to the best-possible visual/haptic feedback quality, whereas a rating of 1 corresponded to the worst quality.

Procedure All subjects participated in a training phase prior to the experiment to familiarize themselves with the task, the rating scale, and the range of stimuli that were included in the experiment. The experimenter explained the procedure and highlighted two references, which corresponded to a rating of 7 and 1 as explained below. The subjects were informed that both references are included in the experiment and that the visual and haptic quality might differ within a single trial. The subjects could compare the current trial to both references at any time by pressing a key on the keyboard. There was no time limit for entering the ratings using the keys 1 to 7 on the keyboard.

Stimuli The server randomly selected a parameter pair (p, v) in each trial (denoted as stimulus $S_{(p,v)}$) from the following parameters: $p = \{2.5\%, 5.0\%, 7.5\%, 10\%, 15\%, \varnothing\}$ and $v = \{0.025, 0.05, 0.075, 0.10, 0.2\}$. A setting of $p = \varnothing$ means that the server does not transmit additional haptic updates for vertices that lie within the AOI.

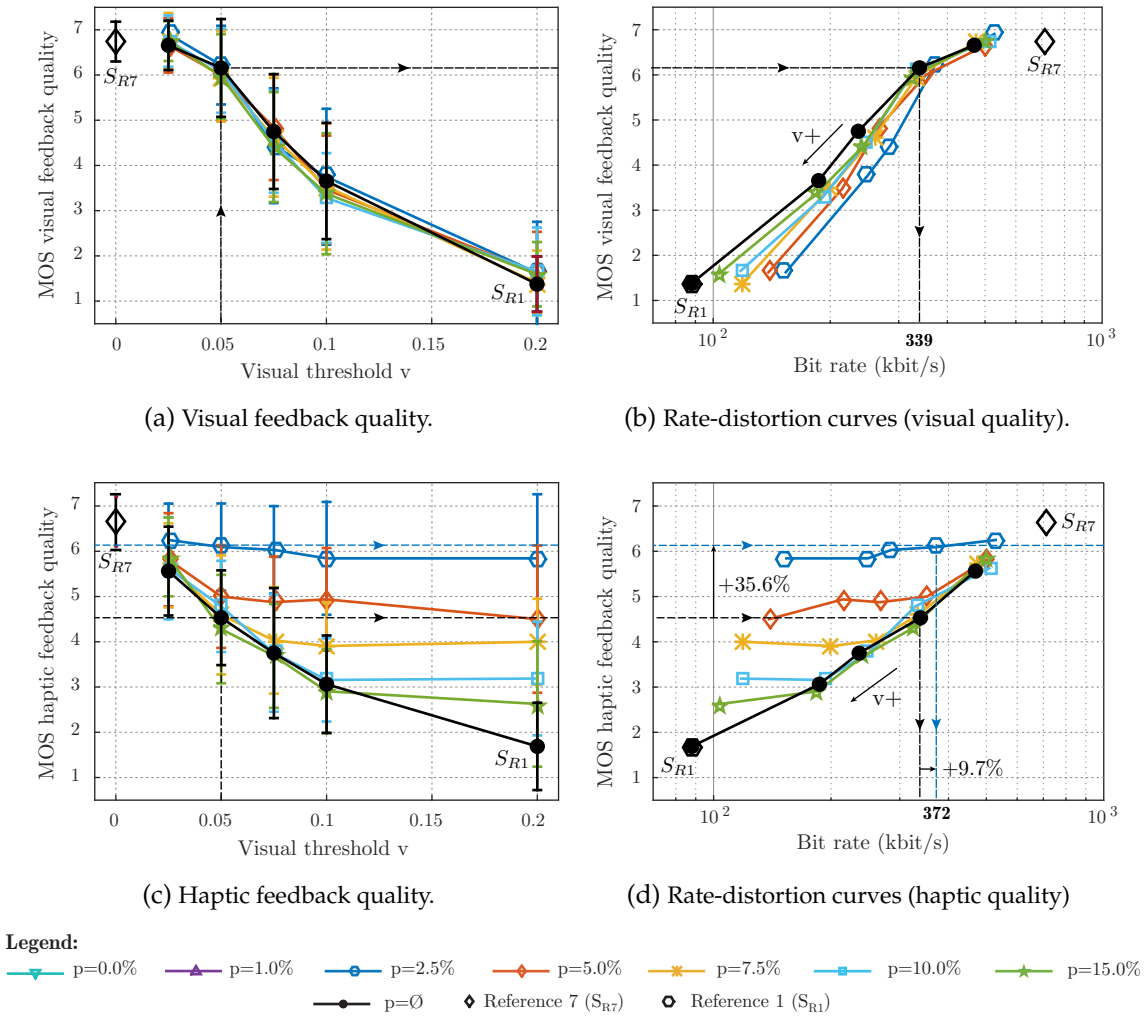
The pair $(\varnothing, 0.2)$ denoted the reference S_{R1} and corresponded to a rating of 1. It represented the worst visual and haptic quality. The additional parameter pair $(0.0, 0.0)$ served as reference S_{R7} and represented a rating of 7. The displacements of all vertices were transmitted after each FEM simulation step to the client in S_{R7}, compressed only using quantization and range coding. Range coding is lossless, but quantization of the displacement data is lossy. The deformation data in the reference S_{R7}, hence, was also lossy compressed. The introduced quantization noise, however, was not perceivable when compared to an uncompressed communication between the server and the client. The reasons are as follows. First, human perception capabilities are limited. Second, the imperfect mechanical components of the haptic device introduce some distortions, perceivable as high-frequency noise [131], which might mask the effects due quantization.

The reference S_{R7} corresponded to a common mesh compression algorithm, which uses only prediction, quantization and entropy coding, but does not use the additional AOI-based visual and haptic thresholds explained in Section 3.3.1.1. The presented subjective test evaluates if these thresholds help to further reduce the bit rate without distorting the visual-haptic feedback quality compared to state-of-the-art realtime mesh compression algorithms.

In total, there were 31 different parameter settings, including the two reference settings. Each setting was tested twice, that is, there were 62 trials per subject.

Participants Sixteen voluntary subjects participated in the experiment (12 male, 4 female, aged between 23 and 46). Five were frequent users of haptic devices, six had already participated in former experiments in our lab, and the others were novice users.

Results Figure 3.19 summarizes the results of the subjective evaluation. The user ratings for the haptic and visual feedback quality are not normally distributed according to the chi-squared goodness of fit test. Hence, non-parametric tests (significance level of 5 %) were used for the statistical analysis of the data [220].



(a) Visual feedback quality.

(b) Rate-distortion curves (visual quality).

(c) Haptic feedback quality.

(d) Rate-distortion curves (haptic quality)

Figure 3.19: Subjective evaluation results. The error bars denote the mean and the standard deviation between all trials performed by the subjects (adopted from [8], © 2016 IEEE).

One can observe from Figure 3.19a that the visual feedback quality decreases with increasing values of the visual threshold v. The impact of p on the visual mean opinion score (MOS) seems negligible. Friedman's analysis of variance also suggests a statistically significant effect of v, but no statistically significant effect of p on the visual MOS.

The haptic MOS decreases for an increasing visual threshold v in Fig. 3.19c if no additional updates are transmitted for the AOI (solid black line for $p = \varnothing$). The haptic MOS for values of $v = 0.025$ are independent of p. It slightly increased only for $p = 2.5\%$. The reason is that the haptic threshold does not trigger additional updates if the visual threshold is small enough.

With increasing values of v the additional updates triggered by the haptic threshold helped to maintain the haptic feedback quality at a level depending on p. This clearly demonstrates the usefulness of the proposed compression scheme, which considers the visual and the haptic modality separately. It offers a high flexibility in adjusting the compression performance depending on the desired visual and haptic feedback quality.

The quality ratings were analyzed with the Kruskal-Wallis test and a post-hoc statistical analysis using Bonferroni correction to determine parameter settings that lead to visual and haptic MOS values that are not significantly different from the reference S_{R7}. In other words, the goal was to identify the parameter combination that leads to satisfying visual and haptic feedback quality, while at the same time minimizes the required bit rate.

A statistically significant difference in the visual MOS compared to S_{R7} can be reported for all parameter pairs with $v > 0.05$. For $(\varnothing, 0.05)$, that is, no additional haptic updates were transmitted, the compression scheme achieved a satisfying visual feedback at a bit rate of 339 kbit/s (see Figure 3.19b).

No statistically significant difference of the haptic MOS compared to S_{R7} was found for all settings with $v = 0.025$ and for all settings with $p = 2.5\%$. The setting of $(\varnothing, 0.05)$, hence, did not achieve a satisfying haptic feedback quality (MOS 4.5, see Fig. 3.19c), although the visual quality was satisfying (MOS 6.1, see Figure 3.19a).

Parameter sets with $p = \varnothing$ constitute the baseline in the evaluation, as they correspond to traditional compression schemes considering the visual modality only. Compared to this baseline, the proposed compression scheme could improve the haptic feedback quality by including additional vertices inside the AOI in the updates. The setting $(2.5\%, 0.05)$ led to a visual MOS of 6.5 and a haptic MOS of 6.1 and required a bit rate of 372 kbit/s (see Figure 3.19d). Both the visual and the haptic MOS were not statistically significantly different from S_{R7} for this setting. As a comparison, if the traffic control parameters were set as $(\varnothing, 0.025)$, the visual MOS of 6.6 and the haptic MOS of 5.5 were also not statistically significantly different to S_{R7}. The haptic MOS, however, was slightly reduced, while the bit rate increased to 469 kbit/s.

3.3.5 Discussion

The objective evaluation revealed that the visual threshold v is the main parameter influencing the introduced mesh distortion and the required bit rate. The haptic parameter p enables the individual control of the coding noise (the vertex position error) that the compression scheme introduces into the AOI. This allows us to fine-tune the visual and haptic feedback quality independently using the two traffic control parameters v and p.

The subjective evaluation suggested that a parameter setting of $v = 0.05$ and $p = 2.5\,\%$ is a good trade-off between the visual and the haptic distortion as well as the required bit rate. The achieved average compression ratio for this setting was 11:1. Figure 3.20 illustrates how the individual coding mechanisms, namely quantization, visual/haptic thresholds and range coding, contributed to the compression of the deformation data. When compared to the reference S_{R7}, which used only quantization and lossless range coding, the visual/haptic thresholds further reduce the bit rate by 47.8 %.

Compared to the visual-only setting of $(\varnothing, 0.05)$, which achieved a satisfying visual feedback quality, but no satisfying haptic feedback quality, the haptic feedback quality increased by 35.6 % for the setting of $(2.5\,\%, 0.05)$, while the bit rate increased by only 9.7 %. Compared to the visual-only setting of $(\varnothing, 0.025)$, the haptic quality increased by 10.9%, while the bit rate was reduced by 20.7 %. These results clearly confirm the assumption that the visual

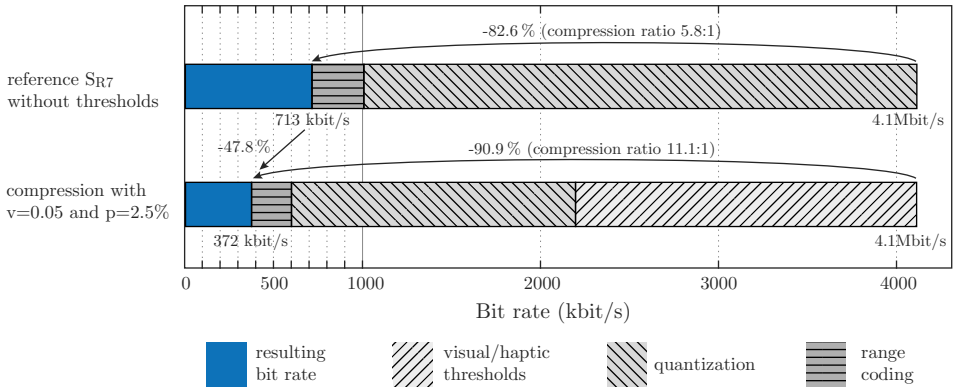


Figure 3.20: Average bit rate reduction achieved by the different coding mechanisms during the subjective experiment for a setting of v=0.05 and p=2.5 %. The compression ratio is 11:1. The resulting visual MOS is 6.5 and the haptic MOS is 6.1.

and haptic modalities should be considered individually to achieve a good trade-off between visual/haptic feedback quality and compression performance.

The reason is that the haptic modality requires more updates than the visual modality due to its perceptual sensitivity. Note that the haptic feedback quality depends on the stiffness used in the penalty-based force rendering at the client. If vertices are updated based on a fixed position threshold, larger stiffness values lead to larger irregularities in the force feedback. The proposed variable haptic threshold for vertices within the AOI considers this relationship by employing the stiffness in the threshold calculation. Furthermore, if fixed thresholds are applied, larger stiffness values require more fine-grained updates and smaller thresholds. These small thresholds, however, are unnecessary for the visual modality. Hence, the performance gain due to the individual consideration of the visual and haptic modalities in the compression algorithm is expected to further increase for higher stiffness values.

3.4 Chapter summary

This chapter has presented two contributions of this thesis, namely two data reduction schemes for the efficient transmission of rigid and deformable object state updates from the server to the clients in the CS-State architecture. The novelty of these schemes lies in the approach to individually consider the visual and haptic modalities and to exploit their respective limited perceptual capabilities.

The data reduction schemes are built on two fundamentally different structures to cope with the characteristics of the traffic in SHVEs that are built on top of the CS-State architecture. These characteristics are summarized as follows:

- High packet rates and low payload characterize the transmission of the device position/velocity on the forward channel from the clients to the server.
- High packet rates and low payload characterize the transmission of rigid object states, but lower packet rates and high payload characterize the transmission of deformable object states on the backward channel from the server to the clients.

While perception-based data reduction schemes to reduce the number of position/velocity packets on the forward channel exist and have been proven to achieve an excellent trade-off between perceptual robustness and packet rate reduction, state-of-the-art approaches for the communication of state updates on the backward channel in the CS-State architecture ignore the human user and his/her perceptual limitations. Hence, they do not fully achieve the potentially possible performance.

The proposed schemes address these limitations by incorporating mathematical models of the human perceptual limitations into the data reduction. The CS-State architecture requires the transmission of object states, which are used to render the virtual environment at the client visually and haptically. Hence, a single data stream influences both the visual and haptic modalities. This is in contrast to traditional bilateral teleoperation systems, where individual haptic (force/torque) and video streams are transmitted on the backward channel. The proposed data reduction schemes consider the visual and haptic modalities individually to avoid the transmission of unnecessary data and to achieve a good trade-off between visual-haptic fidelity and data reduction.

During the interaction with rigid objects, the packet rate needs to be decreased to reduce the load on the communication network. The proposed perception-based data reduction scheme reduced the packet rate from initially 1000 packets per second to 28.6 packets per second during the task used in the presented evaluation, without deteriorating the haptic feedback quality significantly. This corresponds to a reduction of 97 %. It also outperformed state-of-the-art dead reckoning and achieved a better feedback quality with less updates being transmitted from the server to the clients.

The experimental evaluations with rigid objects presented in this chapter also lead to the following conclusions:

- The CS-State architecture requires less packets to be transmitted from the server to the clients compared to the CS-Force architecture. The reason is that the virtual object's mass and inertia act as a low-pass filter and avoid high-frequency signal changes, which trigger many updates in prediction-based coding schemes such the perceptual deadband data reduction scheme for forces.

- At the client-side, object state interpolation can be used to gradually correct inconsistencies due to the reduced number of object state updates. Linear interpolation has been haptically preferred in the subjective comparison of three state-of-the-art methods.

During the interaction with deformable objects, data reduction requires the compression of the payload of each update packet instead of reducing the number of transmitted packets itself. The proposed data reduction scheme uses prediction, quantization and entropy coding to achieve this. Additionally, visual and haptic thresholds are applied to avoid the transmission of vertex updates that are not perceivable at all. The coding scheme applies variable fine-grained thresholds on vertices that are in the neighborhood of the current haptic interaction to consider the different capabilities of the human visual and haptic perception system.

The subjective evaluation has shown that for the tested scenarios, the bit rate can be reduced from originally 4.1 Mbit/s to 372 kbit/s (compression ratio 11:1) without deteriorating the visual and haptic feedback quality significantly. Compared to a compression scheme that does not distinguish between the visual and the haptic modalities, the subjective haptic feedback quality could be improved by 10.9 %, while, at the same time, the bit rate was even reduced by 20.7 %.

Chapter 4

Compensating the effect of delay in client/server-based SHVEs

The previous chapter presented schemes that reduce the amount of data, which is transmitted from the server to the clients in the CS-State architecture, by exploiting human perceptual limitations. Communication delay, which is not negligible, for example, in the communication between geographically distributed users, has not been considered so far.

All architectures conceived to implement SHVEs and discussed in Section 2.4.2 struggle with communication delay. The centralized physics simulation in the CS-State architecture, which is considered in this thesis, ensures a consistent object state between clients. The main drawback is the round-trip-time required until the object states at the clients are updated. This leads to a loss of transparency during the interaction with movable rigid and deformable objects as already explained in Section 2.4.4

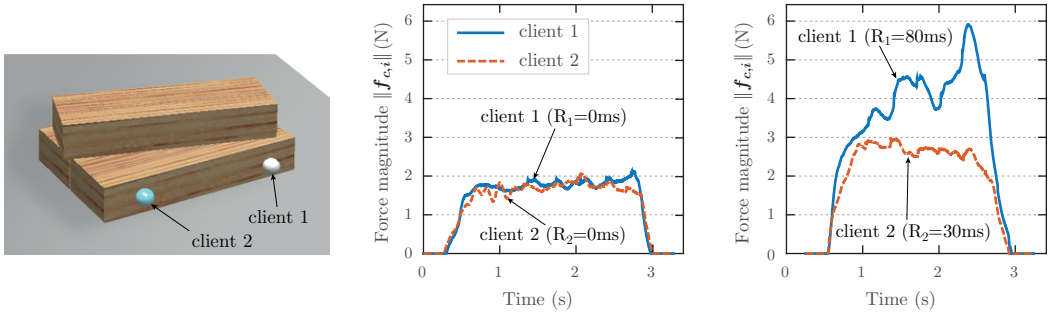
The client/server-based SHVE is called haptically perceived transparent if the user cannot perceive a difference between a fully locally simulated interaction and the interaction through the distributed system, where the physics simulation is running only on the server. A loss of transparency can have critical implications for human decision making, for example, in haptically enhanced virtual surgical training [153].

This chapter investigates the transparency of the CS-State architecture. First, Section 4.1 details the effects of communication delay on the force magnitude at the clients. Section 4.2 and Section 4.3 analyze the haptic transparency and present compensation schemes to achieve perceptual transparency during the interaction with rigid and deformable objects in the CS-State architecture, respectively.

The ideas and contributions developed in this chapter have been published in parts in [1], [2], [9], [10].

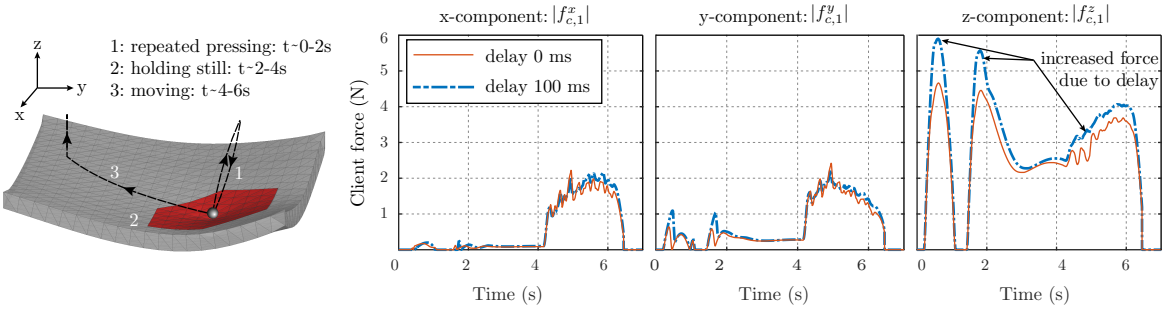
4.1 Problem statement

Figure 4.1 illustrates the effect of delayed state updates on the forces at the clients during a collaborative interaction with rigid objects in the CS-State architecture. The force magnitude at the client i ($\|\boldsymbol{f_{c,i}}\|$) increases with increasing round-trip-time, denoted with R_1 and R_2 in Figure 4.1 for client 1 and client 2. The higher the communication delay, the bigger the effect



(a) Two-user interaction (b) Zero delay reference (c) In the presence of delay

Figure 4.1: (a) Two users cooperatively push rigid objects over a frictional surface. (b) Force signals at the clients for zero communication delay. (c) Force signals at the clients in the presence of communication delay. The communication delay obviously leads to increased rendered forces at the clients (adapted from [2]).



(a) Single-user interaction (b) Recorded force signals at the client

Figure 4.2: (a) Single-user interaction with a membrane-like deformable object. (b) Communication delay leads to an increased force, especially while pressing on the object in the z-direction normal to the surface. There is no effect of delay while holding still and only a small effect while moving along the surface (adapted from [1], © 2016 IEEE).

on the rendered force. The increased force magnitude at the clients leads to a loss of haptic transparency of the SHVE.

The reason is as follows. If the user pushes against an object and communication delay exists, the object will not move immediately because the object motion is simulated on the server and has to be transmitted to the client. The penetration depth of the virtual probe into the object increases as the user further tries to move the object while waiting for the object state update from the server. In turn, the force displayed to the user increases [43], [44], [156].

Studies about the human perception system have observed that the more precise modality often dominates during the integration of various human sensory modalities [47]. While the user might not visually recognize the delayed object state updates for low communication delay, she/he will very likely still notice the increased feedback forces. For the interaction illustrated in Figure 4.1a, the increased force magnitudes lead to the effect of increased perceived weight of the rigid objects as mentioned in [43], [156], [163], [164].

The communication delay has a similar effect during the interaction with simulated deformable objects in the considered CS-State architecture. During the exemplary interaction with the deformable object shown in Figure 4.2a, the rendered force increases, especially

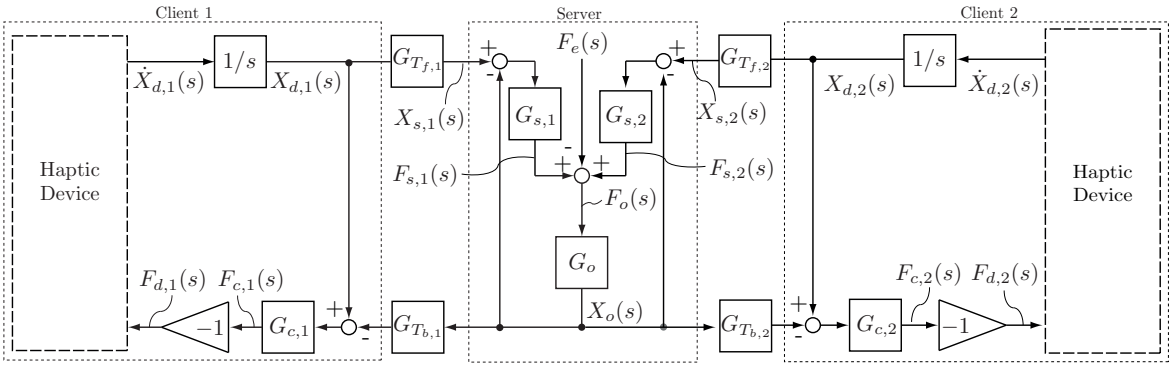


Figure 4.3: LTI model representing a two-user client/server architecture for the interaction with rigid objects. $G_{c,i}$ and $G_{s,i}$ denote the force rendering algorithms at the clients and the server for client i, respectively. The centralized rigid body physics simulation is denoted by G_o (adapted from [2]).

while pressing on the object in the z-direction normal to the surface (see Figure 4.2b). There is no force error while holding still. The delay also leads to a slightly steeper force reduction while removing the haptic device from the object. As humans perceive the compliance of an object based on sensory signals conveying position and force signals (see Section 2.1.2), the human perceives a stiffer object while tapping on the deformable object in the presence of communication delay.

The analyses of the effect of communication delay on the rendered forces at the clients presented in this chapter reveal guidelines on how much delay is tolerable before its effect becomes haptically perceivable. These guidelines incorporate the simulation parameters, the user interactions and the limitations of human perception. The proposed compensation schemes can be seen as a method to achieve perceptual transparency if this tolerable communication delay is exceeded.

4.2 Interaction with movable rigid objects

4.2.1 Two-user client/server model

In the following, the LTI model of the CS-State architecture shown in Figure 4.3 is derived. The intention is to establish a mathematical representation of multi-user interactions with movable rigid objects that allows for an analysis of the effect of communication delay on the forces rendered at the clients.

The focus first lies on a 1-DOF LTI system to simplify the analysis. Subsequently, the obtained 1-DOF results are generalized to point-based 3-DOF haptic interactions. The presented model is an extension of the single-user model used in [44] to a two-user model with a generalized model of the physics simulation on the server.

4.2.1.1 Client/server LTI model

In the LTI model in Figure 4.3, $\dot{X}_{d,i}(s)$ and $X_{d,i}(s)$ denote the velocity and position of the user's virtual probe (the representation of the haptic device in the VE) at the client i in the

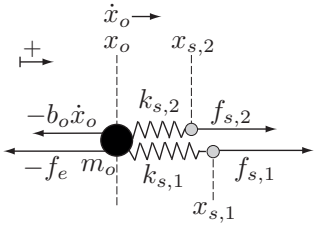


Figure 4.4: Mechanical model of the point mass on the server in the steady state. Both users apply a force in the same direction, leading to an object movement in this direction. The damping in the physics simulation ($-b_o\dot{x}_o$) and the external (friction) force ($-f_e$) act against the movement. The damping elements $b_{s,i}$ have no effect in the steady state and can be neglected (adapted from [2]).

Laplace domain. $X_{s,i}(s)$ is the same, but delayed position at the server.

$G_{c,i}(s)$ represents the constraint-based force rendering algorithm at client i (see Section 2.3.4.1). It implements a spring-damper-based virtual coupling between the virtual probe and the virtual proxy (see Section 2.2.2):

$$G_{c,i}(s) = k_{c,i} + b_{c,i}s \quad \text{with} \quad k_{c,i} > 0,\ b_{c,i} > 0 \tag{4.1}$$

At the clients, the locally calculated interaction forces, $F_{d,i}(s)$, are displayed to the human via the haptic device. Similar to other transparency analyses (e.g., in [153]), the following derivations do not include the haptic device, that is, an ideal haptic device is assumed. This allows us to investigate the effect of communication delay on the rendered forces magnitudes, which is independent of the device dynamics.

At the server, $G_{s,i}(s)$ denotes the constraint-based force rendering algorithm to calculate the interaction force of the client i:

$$G_{s,i}(s) = k_{s,i} + b_{s,i}s \quad \text{with} \quad k_{s,i} > 0,\ b_{s,i} > 0 \tag{4.2}$$

Figure 4.4 illustrates the mechanical model of the shared object. The physics simulation is represented by $G_o(s)$ in Figure 4.3 and modeled as a mass-damper:

$$G_o(s) = \frac{1}{m_o s^2 + b_o s} \quad \text{with} \quad m_o > 0,\ b_o > 0 \tag{4.3}$$

Here, m_o is the object's point mass and b_o represents the non-physical damping of velocities in the simulation (see Section 2.3.2).

Compared to the server model used by Lee *et al.* [44], the dependent input force $F_e(s)$ is added to better incorporate possible contact scenarios between surfaces and objects. $F_e(s)$ corresponds to f_e in the mechanical model in Figure 4.4. It summarizes the external forces acting against the object movement, for example, due to frictional contacts between surfaces. For the contact scenario shown in Figure 4.1a, for example, $F_e(s)$ corresponds to the Laplace transform of the friction force caused by the objects' weight while they are pushed over the ground surface.

Lee *et al.* [44] model the physics simulation only with the mass-damper system $G_o(s)$, which makes it difficult to model various contacts between surfaces. The parameter b_o has

to be pre-determined based on the user interaction and the simulation parameters (e.g., the friction coefficient) in this case. This can be achieved, for example, by measuring the input-output behavior of the SHVE system for various inputs. In contrast, the model presented in this chapter does not require such measurements. Hence, the proposed model supports the generalization towards various interaction scenarios.

The calculated object position on the server, $X_o(s)$, is transmitted back to the clients over the communication network, denoted by the delay elements $G_{T_{f,i}}(s)$ with delay $T_{f,i}$ on the forward channels to the server and $G_{T_{b,i}}(s)$ with delay $T_{b,i}$ on the backward channels to the clients. The round-trip-times are denoted by $R_1 = T_{f,1} + T_{b,1}$ and $R_2 = T_{f,2} + T_{b,2}$ for client 1 and client 2, respectively.

The following equations are derived for client 1 from the LTI model in Figure 4.3. The derivation for client 2 is identical due to the symmetry of the model. At first, the interaction of user 1 with a movable object is considered, while client 2 is not interacting with the same object. In this case, $F_{s,2}(s)$ in Figure 4.3 is zero and only the left half of the LTI model has to be considered:

$$\begin{aligned} F_{c,1}^1(s) =& \tfrac{1}{s} G_{c,1}(s) \left(1 - e^{-s(R_1)} \tfrac{G_{s,1}(s)G_o(s)}{1+G_{s,1}(s)G_o(s)}\right) \dot{X}_{d,1}(s) \\ & - G_{c,1}(s) e^{-s(T_{b,1})} \tfrac{G_o(s)}{1+G_{s,1}(s)G_o(s)} F_e(s) \end{aligned} \tag{4.4}$$

The superscript 1 denotes that only client 1 is considered. If the second user interacts with the same object, only the force applied by client 2 on the server, $F_{s,2}(s)$, may influence the object movement. For client 1, hence, only $F_{s,2}(s)$ and not $\dot{X}_{d,2}(s)$ has to be considered. $F_{s,2}(s)$ can be seen as an additional input similar to $F_e(s)$. The following transfer function represents the effect of this interaction on the force rendered at client 1:

$$F_{c,1}^2(s) = -G_{c,1}(s) e^{-s(T_{b,1})} \tfrac{G_o(s)}{1+G_{s,1}(s)G_o(s)} F_{s,2}(s) \tag{4.5}$$

In the following, $b_{c,i} = b_{s,i} = 0$ is assumed to simplify the calculations. Section 4.2.2 analyzes the steady state, in which the penetration depth of the haptic device into the virtual object remains constant. As a result, the damping elements have no impact on the rendered forces in the steady state and can be set to zero for the analysis without loss of generality.

The delay elements are approximated by a first-order Padé series for low-frequencies:

$$G_{T_{(\cdot)}} = e^{-sT_{(\cdot)}} \approx \frac{1 - \frac{1}{2}T_{(\cdot)}s}{1 + \frac{1}{2}T_{(\cdot)}s} \quad \text{for} \quad \omega < \frac{1}{3T_{(\cdot)}} \tag{4.6}$$

The first-order Padé approximation limits the validity of the transfer functions derived in the following to input frequencies of $\omega < \frac{1}{3T_{(\cdot)}}$ [153]. Again, this limitation does not impair the analysis, which considers only the steady state ($s = j\omega = 0$).

The force at the client is derived by inserting Equations (4.1), (4.2), (4.3), and (4.6) into Equation (4.4) and Equation (4.5). This results in

$$\begin{aligned} F_{c,1}^1(s) =& \frac{s^2(\frac{k_{c,1}m_oR_1}{2}) + s(m_ok_{c,1} + \frac{b_ok_{c,1}R_1}{2}) + k_{c,1}k_{s,1}R_1 + b_ok_{c,1}}{s^3(m_o\frac{R_1}{2}) + s^2(m_o + b_o\frac{R_1}{2}) + s(k_{s,1}\frac{R_1}{2} + b_o) + k_{s,1}} \dot{X}_{d,1}(s) \\ & + \frac{-k_{c,1}\frac{T_{b,1}}{2}s + k_{c,1}}{s^3(m_o\frac{T_{b,1}}{2}) + s^2(m_o + b_o\frac{T_{b,1}}{2}) + s(k_{s,1}\frac{T_{b,1}}{2} + b_o) + k_{s,1}} F_e(s) \end{aligned} \tag{4.7}$$

and

$$F^2_{c,1}(s) = -\frac{-k_{c,1}\frac{T_{b,1}}{2}s+k_{c,1}}{s^3(m_o\frac{T_{b,1}}{2})+s^2(m_o+b_o\frac{T_{b,1}}{2})+s(k_{s,1}\frac{T_{b,1}}{2}+b_o)+k_{s,1}}F_{s,2}(s) \tag{4.8}$$

Due to the linearity of the model, $F^1_{c,1}(s)$ and $F^2_{c,1}(s)$ can be summed to get the force rendered at client 1 for a cooperative two-user interaction with a shared object:

$$F_{c,1}(s) = F^1_{c,1}(s) + F^2_{c,1}(s) \tag{4.9}$$

4.2.1.2 Parameters of the LTI model

Most parameters that describe the LTI model in Figure 4.3 are known. The force rendering parameters $k_{c,i}$, $b_{c,i}$, $k_{s,i}$ and $b_{s,i}$ are explicitly set in the haptic application. The delay between the clients and the server can be measured, for example, using time stamps. Hence, $T_{f,i}$, $T_{b,i}$ and also $R_i = T_{f,i} + T_{b,i}$ are assumed to be known.

The mass m_o is set as a parameter in the physics simulation. As mentioned in Section 2.3.2, the physics simulation applies a general damping of velocities for stability reasons. The SHVE prototype implemented for this thesis uses the ODE physics simulation [211]. It multiplies the object velocities with a defined factor $0 < d < 1$ after each integration time step Δt. The kinetic energy lost due to this non-physical damping can be calculated as:

$$\Delta E = \frac{1}{2}m_o\left[\dot{x}_o(t)^2 - (\dot{x}_o(t)d)^2\right] = \frac{1}{2}m_o\dot{x}_o(t)^2(1-d^2) \tag{4.10}$$

The viscous damper b_o in $G_o(s)$ (see Equation (4.3)) is employed to model this energy dissipation. The object velocity is constant during a time step $t \rightarrow t + \Delta t$. A viscous damper dissipates the energy

$$E_{b_o} = b_o\dot{x}_o(t)\Delta x_o = b_o\dot{x}_o(t)^2\Delta t \tag{4.11}$$

during this period. By equating Equation (4.10) and Equation (4.11), the parameter b_o is calculated as:

$$b_o = \frac{m_o(1-d^2)}{2\Delta t} > 0 \tag{4.12}$$

In summary, all parameters required to model the virtual environment are known or can be calculated. $\dot{X}_{d,1}(s)$ and $\dot{X}_{d,2}(s)$ are the user inputs. F_e is calculated within the physics simulation on the server.

4.2.2 Transparency analysis

In this section, the transparency of the CS-State architecture for the interaction with movable rigid objects is discussed based on the previously presented LTI model. Transparency is investigated for client 1 in the steady state ($s = j\omega = 0$). This practice is also used in the analysis of teleoperation systems (e.g., in [226]). The analysis for client 2 is identical. The steady state implies a constant input velocity of client 1 and, as a consequence, the same velocity for the object and client 2. Otherwise the spring's elongation would change, leading to a change in the rendered force and, hence, to an acceleration/deceleration of the object.

It can be shown with the Routh-Hurwitz stability criterion [227] that the transfer functions in Equation (4.7) and Equation (4.8) are asymptotically stable for $b_o > 0$, independent of the delay elements and, hence, a steady state exists. This does not imply stability of the complete haptic application, as the presented model does not include the haptic device and the human user.

4.2.2.1 Ideal transparency

A common definition of ideal transparency in the context of teleoperation systems requires the position and force signals on the local and remote side to be identical [151]. Similarly, the positions and forces at the clients and the server need to be identical for ideal transparency of the SHVE.

The steady-state force response at the client 1 in the time-domain ($f_{c,1}$) for a unit-step input ($\dot{X}_{d,1}(s) = \dot{x}_{d,1}\frac{1}{s}$, $F_e(s) = f_e\frac{1}{s}$, $F_{s,2}(s) = f_{s,2}\frac{1}{s}$) can be calculated with the Equations (4.7), (4.8), (4.9) and the final value theorem as:

$$f_{c,1} = \lim_{t\to\infty} f_{c,1}(t) = \lim_{s\to 0} sF_{c,1}(s) = \frac{k_{c,1}}{k_{s,1}}(k_{s,1}R_1 + b_o)\dot{x}_{d,1} + \frac{k_{c,1}}{k_{s,1}}f_e - \frac{k_{c,1}}{k_{s,1}}f_{s,2} \tag{4.13}$$

Here, f_e denotes the amplitude of the steady-state external force, $\dot{x}_{d,1}$ the amplitude of the steady-state device velocity of client 1, and $f_{s,2}$ the amplitude of the steady-state force of client 2 applied on the object on the server. Equation (4.13) is rewritten as

$$f_{c,1} = \frac{k_{c,1}}{k_{s,1}}\left(R_1 k_{s,1}\dot{x}_{d,1} + b_o\dot{x}_o + f_e - f_{s,2}\right) \tag{4.14}$$

because $\dot{x}_{d,1}$ equals $\dot{x}_o$ in the steady state. The steady-state force rendered on the server for client 1 is known from the LTI model (Figure 4.3) or the mechanical model (Figure 4.4) and given as:

$$f_{s,1} = b_o\dot{x}_o + f_e - f_{s,2} \tag{4.15}$$

Ideally, the server force is also displayed to the user at the client. For zero communication delay (R_1 = 0 ms), the client force $f_{c,1}$ in Equation (4.14) equals the desired server force $f_{s,1}$ in Equation (4.15) only if the stiffness coefficients $k_{s,i}$ and $k_{c,i}$ are set to be equal. This setting is denoted as the *original stiffness setting* with $k_{c,i}$ and $k_{s,i}$ set to $k^o_{c,i}$ (the superscript o denotes *original*). Note that $k^o_{c,i}$ is desirably chosen as large as possible while still guaranteeing a stable haptic interface (see Section 2.2.2 and Section 2.3.4.1). The rendered steady-state force at the client increases with increasing delay for a constant value for $k_{c,i}$:

$$f_{c,1} = R_1 k_{c,1}\dot{x}_{d,1} + b_o\dot{x}_o + f_e - f_{s,2} \tag{4.16}$$

Only the first term in Equation (4.16) depends on the RTT of client 1. If the original stiffness setting $k^o_{c,i}$ is used, the client i interacting with the virtual object perceives an increased force magnitude $|\Delta f_{c,i}|$ in the steady state of:

$$|\Delta f_{c,i}| = k^o_{c,i}R_i|\dot{x}_{d,i}| \tag{4.17}$$

This result is intuitive: $|\Delta f_{c,i}|$ is caused by the increased penetration depth ($R_i|\dot{x}_{d,i}|$) multiplied with the stiffness used in the force rendering at the client.

In summary, the requirement for ideal transparency of positions and forces being the same on the server and the client can obviously only be fulfilled for zero communication delay (R_i = 0 ms). In the presence of delay, the object positions, as well as the haptic device positions at the client and the server will differ. For the CS-State architecture, the difference in positions is an accepted trade-off for a globally consistent object state calculated at the server.

The delayed object state updates lead to an increased force feedback displayed to the user (see Equations (4.16) and (4.17)). This gives the user the impression of increased weight of an object, as observed in the experiments in [156], if the object is, for example, pushed over a surface with non-zero friction as illustrated in Figure 4.1a.

The human user might recognize the delayed object state updates visually and haptically. It has been observed that the more precise modality often dominates during the integration of various human sensory modalities [47]. While the user might not visually recognize the delayed object acceleration for low communication delay, she/he will very likely still notice the increased feedback forces.

4.2.2.2 Perceived transparency

It is known from psychophysics that human perception has limitations in terms of detection and discrimination thresholds (see Section 2.1.3). The concept of *perceived transparency* introduced by Hirche *et al.* [153], [228] in the context of bilateral teleoperation systems relaxes the condition for ideal transparency by considering human perceptual limitations. Similarly, the SHVE is assumed as haptically perceived to be transparent if the increased force rendered at the client (4.14) is not distinguishable from the ideally displayed force (4.15).

Perceivable communication delay The *perceivable communication delay* is the RTT that causes a loss of perceived transparency, that is, the RTT for which the effect of delay becomes potentially perceivable. Hence, the perceivable communication delay denotes a guideline for the tolerable RTT in the CS-State architecture. The increased force magnitude at the client is used to describe the loss of perceived transparency, because communication delay has a direct effect on the rendered forces as shown in Figure 4.1.

With the desired force from Equation (4.15), the Weber-fraction c, and the increased force magnitude $|\Delta f_{c,i}|$ from Equation (4.17), one can write for client 1 in the two-user scenario:

$$k^o_{c,1} R_1 |\dot{x}_{d,1}| = c\,|b_o \dot{x}_o + f_e - f_{s,2}| \tag{4.18}$$

The left part is the increased force magnitude due to delay. The right part is the just-noticeable force magnitude change. Rearranged to R_1, the maximum allowable RTT R_1^{max} for the client 1 is:

$$R_1^{max} = c\,\frac{|b_o \dot{x}_o + f_e - f_{s,2}|}{|\dot{x}_{d,1}| k^o_{c,1}} \tag{4.19}$$

If the RTT exceeds R_1^{max} and the user controls the haptic device around a constant operating point $\dot{x}_{d,1}$, which equals $\dot{x}_o$ in the steady state, the rendered force at the client exceeds the zero-delay reference by c percent in the steady state. Assuming that the Weber-fraction c correctly predicts the perceivable difference in force magnitude during the current user interaction, R_1^{max} also predicts the perceivable communication delay.

This is generalized for client i with N clients manipulating the same object by replacing the force applied on the object by client 2 with the sum of all user forces acting on the object except of the force of user i:

$$R_i^{max} = c\,\frac{|b_o\dot{x}_o + f_e - f_s|}{|\dot{x}_{d,i}|k_{c,i}^o} \quad \text{with} \quad f_s = \sum_{j;j\neq i}^{N} f_{s,j} \tag{4.20}$$

The Equations (4.19) and (4.20) reduce to the formulas derived in [44] if only a single user is considered and the external force f_e is calculated as a viscous friction force ($f_e = b\dot{x}_o$), that is, a viscous friction model between surfaces is assumed in the physics simulation.

Perceivable communication delay change The *perceivable communication delay change* from R_i to $R_i \pm \Delta R_i^{max}$ is derived similarly from Equation (4.16) and Equation (4.17) with ΔR_1 for client 1:

$$\pm\Delta R_1|\dot{x}_{d,1}|k_{c,1}^o = \pm\, c\,|R_1 k_{c,1}^o \dot{x}_{d,1} + b_o\dot{x}_o + f_e - f_{s,2}| \tag{4.21}$$

Rearranged, this leads to the perceivable delay change of:

$$\pm\Delta R_1^{max} = \pm c\,\frac{|R_1 k_{c,1}^o \dot{x}_{d,1} + b_o\dot{x}_o + f_e - f_{s,2}|}{|\dot{x}_{d,1}|k_{c,1}^o} \tag{4.22}$$

Again, this can be rewritten in the generalized form:

$$\pm\Delta R_i^{max} = \pm c\,\frac{|R_i k_{c,i}^o \dot{x}_{d,i} + b_o\dot{x}_o + f_e - f_s|}{|\dot{x}_{d,i}|k_{c,i}^o} \quad \text{with} \quad f_s = \sum_{j;j\neq i}^{N} f_{s,j} \tag{4.23}$$

In summary, the above formulas for the perceivable communication delay and the perceivable communication delay change are based on the original stiffness setting with $k_{c,i}$ set to $k_{c,i}^o = k_{s,i}$ and on a Weber-fraction c describing the human force magnitude difference threshold. They constitute guidelines on how much delay is acceptable in the CS-State architecture before the effect of delay becomes haptically perceivable. They incorporate the simulation parameters, a model of the limitations of human perception, and the current user interaction.

The formulas consider, however, only the kinesthetic force feedback independent of the visual feedback. For higher delay values, the user will also recognize the delay visually, which may also influence the perception of the kinesthetic force feedback. The results reported in [229] suggest that delayed state updates becomes visually perceivable at about 100 ms. The perceivable communication delay for force feedback is typically much lower.

The reason for the increased force magnitude at the client is the RTT in the communication with the centralized server. The penetration depth of the virtual probe into the object

increases proportionally to the device velocity during the round-trip-time. The fractions in Equation (4.20) and Equation (4.23) decrease accordingly, which suggests that also the perceivable communication delay decreases. At the same time, however, there is also evidence that c increases with increasing velocity during dynamic arm movements [191], [230]. This suggests that the perceivable communication delay increases again. Hence, the perceivable communication delay presumably does not decrease exactly inversely proportional to the device velocity.

Furthermore, if the delay varies drastically while the user is in contact with an object, other artifacts might become haptically and visually perceivable. This might lead again to a loss of perceived transparency as discussed Section 4.2.4.5.

4.2.3 Compensating the effect of delay

To compensate for the increased force feedback at the client due to communication delay, the authors of [164], [44] and [231] have proposed to reduce the stiffness used in the local force rendering at the client. The stiffness has been chosen for constant communication delay using a heuristic [164] or a simple single-user LTI model [44], which is, however, difficult to generalize as discussed in Sections 2.4.4 and 4.2.1. Suzuki *et al.* [231] adopted the heuristic proposed in [164] and have evaluated its usability for time-varying delay.

The algorithm presented in the following adopts the idea to reduce the stiffness at the clients to compensate the effect of delay. Based on the previously discussed LTI model (Section 4.2.1), this section shows how a reduced stiffness can be calculated and how this knowledge can be exploited at the clients to implement a scheme to compensate for the effect of constant and time-varying delay during multi-user interactions with movable rigid objects.

4.2.3.1 Stiffness adaptation for 1-DOF interactions

To reduce the effect of delay on the steady-state force at the client, the force rendering algorithm reduces the stiffness $k_{c,i}$ at the client i to a value denoted as $k^r_{c,i}$. The goal is that the steady-state force magnitude in the presence of communication delay matches (using a reduced stiffness $k^r_{c,i}$) the steady-state force for the zero-delay case (using the original stiffness $k^o_{c,i}$).

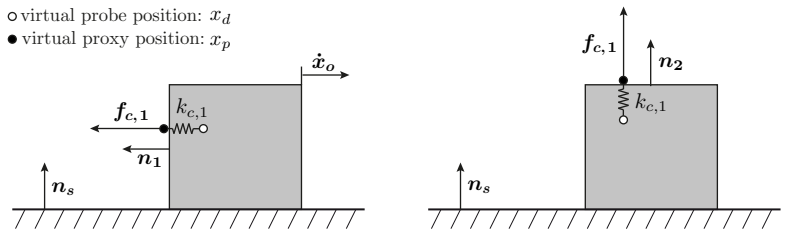
The following relationship is derived by equating Equation (4.14) with Equation (4.15) and setting $k_{c,1}$ to the reduced value $k^r_{c,1}$, while $k_{s,1}$ is kept at the original value of $k_{s,1} = k^o_{c,1}$:

$$\frac{k^r_{c,1}}{k^o_{c,1}}|R_1 k^o_{c,1}\dot{x}_{d,1} + b_o\dot{x}_o + f_e - f_{s,2}| = |b_o\dot{x}_o + f_e - f_{s,2}| \quad \text{with} \quad k^r_{c,1} \leq k^o_{c,1} = k_{s,1} \tag{4.24}$$

Rearranging leads to the reduced stiffness $k^r_{c,1}$ for client 1:

$$k^r_{c,1} = k^o_{c,1}\frac{|b_o\dot{x}_o + f_e - f_{s,2}|}{|b_o\dot{x}_o + f_e - f_{s,2} + R_1 k^o_{c,1}\dot{x}_{d,1}|} \tag{4.25}$$

For R_1 = 0 ms the fraction on the right hand side of Equation (4.25) is one and the stiffness $k^r_{c,1}$ to be used at the client 1 is the original stiffness value: $k^r_{c,1} = k^o_{c,1}$. With increasing delay, the stiffness $k^r_{c,1}$ decreases.



(a) Pushing from the side (b) Pressing on the top

Figure 4.5: Illustration of two different user interactions with a cubic virtual object. (a) The user pushes the cube, leading to object velocity $\dot{x}_o$. (b) The user presses from the top. The object movement is constrained by the ground (adapted from [10], © 2014 IEEE)

In the general multi-user form, one can rewrite Equation (4.25) as:

$$k^r_{c,i} = k^o_{c,i} \frac{|b_o \dot{x}_o + f_e - f_s|}{|b_o \dot{x}_o + f_e - f_s + R_i k^o_{c,i} \dot{x}_{d,i}|} \quad \text{with} \quad f_s = \sum_{j;j \neq i}^{N} f_{s,j} \tag{4.26}$$

In summary, if the communication delay exceeds the perceivable communication delay from Equation (4.20), the stiffness used in the proxy algorithm at the client should be set according to Equation (4.26) to match the steady-state forces. If the communication delay changes by more than ΔR_i^{max} from Equation (4.23) while the user is in contact with an object, the stiffness should be updated again. Abrupt stiffness updates introduce perceivable artifacts and jumps in the rendered force feedback. A moving average filter on the calculated stiffness value to smoothly converge to the desired value can be used as explained in Section 4.2.3.4.

4.2.3.2 Extended proxy algorithm

Normally, the clients apply the original stiffness $k^o_{c,i}$ in the local force rendering algorithm. The reduced stiffness $k^r_{c,i}$ is used to compensate for the increased resistive force while moving a rigid object in the presence of communication delay. This corresponds to the interaction sketched in Figure 4.5a. However, if the movement of the object is constrained by another surface (e.g., the ground in Figure 4.5b), the stiffness adaptation leads to a new artifact. The reduced stiffness might impair the illusion of an interaction with a rigid object for the user. Remember that the stiffness is desirably chosen as large as possible, while still guaranteeing stability of the haptic interface (see Section 2.3.4.1).

Equation (4.26) is extended as follows to overcome this issue. A new stiffness $k^r_{c,i}$ is chosen adaptively at the client i not only based on the delay, but also depending on the contacted triangle j (represented by its normal vector $\boldsymbol{n_j}$ in Figure 4.5) as well as on local object motion constraints (represented by the normal vector $\boldsymbol{n_s}$ of a constraining surface). For the two rather extreme cases shown in Figure 4.5, with the force applied on the object being either orthogonal or parallel to the surface constraint $\boldsymbol{n_s}$, the choice of the proper stiffness for force rendering is obvious. For the former (Figure 4.5a), $k_{c,1}$ should be set to $k^r_{c,1}$ given by Equation (4.26), while $k^o_{c,i}$ is appropriate for the latter (Figure 4.5b). For all cases in between

these two extremes, the angle between $\boldsymbol{n_j}$ and $\boldsymbol{n_s}$ defines the amount of stiffness reduction to be applied for the local force rendering. This can be described with the blending factor β ($0 \leq \beta \leq 1$) as:

$$\beta = \frac{2}{\pi} \arccos(|\boldsymbol{n_s} \cdot \boldsymbol{n_j}|) \tag{4.27}$$

Both $\boldsymbol{n_j}$ and $\boldsymbol{n_s}$ are unit vectors. The vector $\boldsymbol{n_j}$ is the normal vector of the contacted triangle and is determined by the proxy algorithm. Motion constraints, however, need to be determined additionally at runtime. The blending factor β is incorporated into Equation (4.26) to consider motion constraints in the stiffness adaptation procedure:

$$k^r_{c,i} = k^o_{c,i} \frac{|b_o \dot{x}_o + f_e - f_s|}{|b_o \dot{x}_o + f_e - f_s + \beta R_i k^o_{c,i} \dot{x}_{d,i}|} \quad \text{with} \quad f_s = \sum_{j;j \neq i}^{N} f_{s,j} \tag{4.28}$$

If the blending factor is one (i.e., $\boldsymbol{n_j}$ and $\boldsymbol{n_s}$ are orthogonal to each other), the stiffness is fully reduced. This corresponds to the stiffness required for the interaction in Figure 4.5a. If the blending factor is zero (i.e., $\boldsymbol{n_j}$ and $\boldsymbol{n_s}$ are parallel to each other), the fraction in Equation (4.28) becomes one and the original stiffness is used: $k^r_{c,i} = k^o_{c,i}$.

4.2.3.3 Extension from 1-DOF to 3-DOF

So far, only a one-dimensional interaction has been considered, where the user movement, the object movement and the considered forces are all aligned. The steady-state force at the client in Equation (4.16) describes the resistive force the user feels in the presence of communication delay while interacting with the object in this one-dimensional example. For 3-DOF, the forces acting on the object on the server ($\boldsymbol{f_e}$) and the forces applied by the other clients ($\boldsymbol{f_{s,j}}, j \neq i$) are obviously not necessarily in the same direction as the movement and the force of client i.

The direction of the increased force, however, is inherently aligned to the client's movement. The penetration depth increases along the direction of the device velocity until the object state at the client is updated, which leads to the increased force rendered at the client. The force vectors $b_o \dot{\boldsymbol{x}}_{\boldsymbol{o}}$, $\boldsymbol{f_e}$ and $\boldsymbol{f_s} = \sum_{j;j \neq i}^{N} \boldsymbol{f_{s,j}}$ are projected on the direction of the movement of user i to extend the 1-DOF equations to 3-DOF.

The resistive force against the direction of the movement for client i is then given as

$$\boldsymbol{f_{r,i}} = -\frac{(b_o \dot{\boldsymbol{x}}_{\boldsymbol{o}} + \boldsymbol{f_e} - \boldsymbol{f_s}) \cdot \dot{\boldsymbol{x}}_{\boldsymbol{d,i}}}{\|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\|} \frac{\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}}{\|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\|} = f_{r,i} \frac{\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}}{\|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\|} \tag{4.29}$$

where $(\cdot)$ denotes the dot product between two vectors and $\|\cdot\|$ the vector norm. The amplitude of the resistive force is denoted by $f_{r,i}$. For the case of the resistive force acting against the user movement, that is, the dot product in Equation (4.29) being positive ($f_{r,i}$ being negative), Equation (4.25) can be rewritten as:

$$k^r_{c,i} = k_{c,i} \frac{-f_{r,i}}{-f_{r,i} + \beta R_i k^o_{c,i} \|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\|} \tag{4.30}$$

With $f_{r,i}$ being negative, the fraction in Equation (4.30) is always positiv and smaller or equal to one. Equation (4.20) and Equation (4.23) are similarly rewritten for 3-DOF:

$$R_i^{max} = c\,\frac{-f_{r,i}}{\|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\|k_{c,i}^o} \tag{4.31}$$

$$\pm\Delta R_i^{max} = \pm c\,\frac{R_i k_{c,i}^o\|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\| - f_{r,i}}{\|\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}\|k_{c,i}^o} \tag{4.32}$$

4.2.3.4 Implementation

The following section explains how each client uses the previous formulas to calculate a reduced stiffness for the local force rendering algorithm.

Equation (4.30) includes $f_{r,i}$, which describes the amplitude of the resistive force the client will feel while pushing the object with constant velocity $\dot{\boldsymbol{x}}_{\boldsymbol{d,1}}$. The value of $f_{r,i}$ depends on the contact forces rendered in the physics simulation and the forces applied by other users on the server. Hence, the knowledge to calculate the reduced stiffness using Equation (4.30) is not directly available at the client.

To overcome this issue, the object states $\boldsymbol{S}$ transmitted from the server to the clients (see Section 3.2) include the forces acting on the objects in the VE: $\boldsymbol{S} = \{\boldsymbol{x_o}, \boldsymbol{r_o}, \dot{\boldsymbol{x}}_{\boldsymbol{o}}, \dot{\boldsymbol{r}}_{\boldsymbol{o}}, \boldsymbol{f_e}, \boldsymbol{f_u}\}$. The force vector $\boldsymbol{f_e}$ represents forces in the physics simulation due to gravity, frictional contacts and contacts with other objects (i.e., $\boldsymbol{f_e}$ corresponds to $F_e(s)$ in Figure 4.3). The set $\boldsymbol{f_u} = \{\boldsymbol{f_{s,1}}, ..., \boldsymbol{f_{s,N}}\}$ includes the forces on the object applied by the users at the server.

This information, together with the measured RTT at the client (R_i), the original stiffness of the object $k_{c,i}^o$, the locally rendered force in the last iteration $\boldsymbol{f_{c,i}}$, and the device velocity $\dot{\boldsymbol{x}}_{\boldsymbol{d,i}}$ are used in the algorithm below to calculate a reduced stiffness at the client for the case of a coulomb friction contact of the touched object with one or more surface(s).

There is no dynamic friction rendered in the physics simulation if the object is in contact with another surface, but currently not moving ($\|\dot{\boldsymbol{x}}_{\boldsymbol{o}}\| = 0.0$). The proposed stiffness reduction in Equation (4.30), however, is based on the resistive force during a constant movement of the object. Hence, at the initial contact, $\boldsymbol{f_e}$ is not known directly. In this case, the client estimates the expected resistive force as the magnitude of the normal force $\|\boldsymbol{f_n}\|$ multiplied with the friction coefficient μ (line 7). The normal force is included in the received $\boldsymbol{f_e}$, while the friction coefficient is set in the physics simulation and, thus, is known.

Note that the used physics simulation, ODE [211], does not distinguish between static and dynamic friction. It uses the same friction coefficient to implement static and dynamic friction in the virtual environment. If the object is not moving and the magnitude of the locally rendered force $\boldsymbol{f_{c,i}}$ is smaller than the estimated friction force (i.e., the static friction force is not exceeded), the client uses the original stiffness value $k_{c,i}^o$ to calculate the interaction force (line 8).

If the object is moving, that is, the client has received a non-zero object velocity $\|\dot{\boldsymbol{x}}_{\boldsymbol{o}}\|$, the resistive force is directly calculated with Equation (4.29) (line 9). The calculated resistive force f_r now represents the resisting force due to the frictional contact in the physics simulation and the desired stiffness $k_{c,i}^r$ is calculated in line 2 with Equation (4.30).

Algorithm 1: Algorithm to calculate the stiffness at the client

Algorithm `calculateObjectStiffnessAtClient` (R_i, $k^o_{c,i}$, $\dot{\boldsymbol{x}}_{\boldsymbol{d},\boldsymbol{i}}$, $\boldsymbol{f}_{\boldsymbol{c},\boldsymbol{i}}$, S)

1 $f_r \leftarrow$ `calculateResistiveForceAtClient` ($\dot{\boldsymbol{x}}_{\boldsymbol{d},\boldsymbol{i}}$, $\boldsymbol{f}_{\boldsymbol{c},\boldsymbol{i}}$, S)

2 **if** $f_r < 0$ **then** $k[t] \leftarrow k^r_{c,i}$ /* $k^r_{c,i}$ calculated with Equation (4.30) */

3 **else** $k[t] \leftarrow k^o_{c,i}$ /* Use original stiffness if force is not acting against the user's movement */

4 $\overline{k}[t] \leftarrow \lambda \cdot k[t] + (1-\lambda) \cdot \overline{k}[t-1]$ /* Apply an exponential moving average filter */

5 **return** $\overline{k}[t]$

Procedure `calculateResistiveForceAtClient` ($\dot{\boldsymbol{x}}_{\boldsymbol{d},\boldsymbol{i}}$, $\boldsymbol{f}_{\boldsymbol{c},\boldsymbol{i}}$, S)

/* Special case: the object is currently not moving! */

6 **if** $\|\dot{\boldsymbol{x}}_{\boldsymbol{o}}\| == 0.0$ **then**

7 $f_r \leftarrow -\mu\|\boldsymbol{f}_{\boldsymbol{n}}\|$ /* The resistive force is the static friction force */

/* The current force magnitude is smaller the than static friction force. We return 0 to keep the original stiffness: */

8 **if** $\|\boldsymbol{f}_{\boldsymbol{c},\boldsymbol{i}}\| < |f_r|$ **then return** 0

9 **else** $f_r \leftarrow f_{r,i}$ /* Standard case: calculate the resistive force $f_{r,i}$ with (4.29) */

10 **return** f_r

The desired stiffness $k^r_{c,i}$ might, however, change drastically with time-varying delay and varying device velocity. This leads to perceivable artifacts and jumps in the rendered force feedback. To mitigate this effect, the client uses an exponential moving average filter on the calculated stiffness value to smoothly converge to the desired value (line 4). A similar filter is also used by Suzuki *et al.* [231]. The stronger the low-pass characteristic of this filter, defined by the parameter λ, the slower the client adapts to the new stiffness value. While slow stiffness updates are desirable to avoid sudden force jumps, they also lead to a mismatch between the desired stiffness $k^r_{c,i}$ and the displayed stiffness. Hence, a trade-off between perceptually convincing and fast enough updates needs to be found. A value of $\lambda = 0.01$ was found as a reasonable value in the implemented SHVE.

Note that the client calculates a desired stiffness and updates the displayed stiffness using the moving average filter even if the delay change is below ΔR_i^{max} to avoid large stiffness updates and to use a stiffness value that follows the desired value as close as possible.

The proposed stiffness adaptation scheme sometimes needs to increase the stiffness while the user is in contact with the object, for example due to communication delay changes. This artificially creates energy in the local force rendering, leading to a non-passive behavior and potential instability if the applied damping at the client ($b_{c,i}$) does not dissipate enough energy. The aforementioned filter leads to slow stiffness updates so that instabilities due to sudden stiffness updates and excessive energy generation are avoided. Alternatively, a stability ensuring stiffness update method based on the concept of passivity as proposed in [7] can be used.

4.2.4 Evaluation

This section presents objective and subjective evaluation results of the previous analysis and the proposed compensation scheme. The communication delay was simulated by software to have the best-possible control over the runtime conditions. Furthermore, as the eval-

uations consider only the effect of communication delay, the communication between the clients and the server was not modified, that is, no data reduction schemes were applied and information was exchanged at the original rate of 1 kHz.

4.2.4.1 Simulation results for constant communication delay

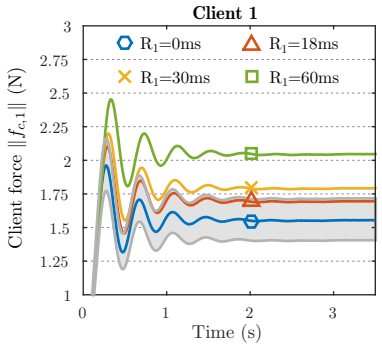
A 1-DOF user input with constant velocity was simulated to verify the two-user client/server LTI model (Section 4.2.1), the derived tolerable delay boundaries (Section 4.2.2), and the proposed delay compensation scheme (Section 4.2.3).

The forces recorded for four constant delay values are plotted in Figure 4.6. The increasing steady-state force at the client for increasing RTT can be clearly observed for both clients (Figure 4.6a and 4.6b). The Weber-fraction of $c = 10\%$, together with the reference force ($R_1 = R_2 = 0\,\mathrm{ms}$) defines the so-called *area of perceived transparency*. If the rendered force falls within this area, the user potentially does not perceive the difference to the reference force. Note that the applied Weber-fraction of 10 % lies in the range of static force discrimination thresholds as discussed in Section 2.1.3. The calculated perceivable communication delay can be seen as conservative, because the Weber-fraction increases with increasing hand/arm velocity [191], [230].

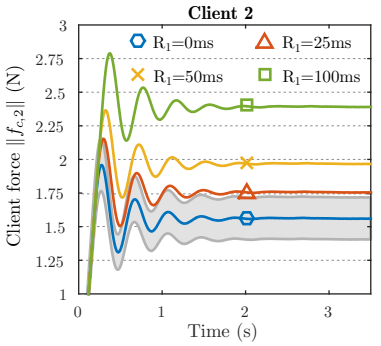
The estimated perceivable communication delay for the simulated input, calculated with Equation (4.31) and $c = 10\%$, is R_1^{max} = R_2^{max} = 18 ms. This delay value well predicts an increased steady-state force magnitude of 10 percent. This can be clearly observed in Figure 4.6a, where the steady-state force for R_1 = 18 ms falls on the border of the area of perceived transparency. For client 2 and R_2 = 25 ms in Figure 4.6b the steady-state force is slightly above the area of perceived transparency.

The proposed stiffness adaptation approach is verified with the same simulated input. The rendered force values at the clients are plotted in Figure 4.6c and Figure 4.6d. The steady-state force matches the zero-delay reference during the first time period for the constant RTT until the delay increases at around $t = 2.8\,\mathrm{s}$. The predicted perceivable delay change with Equation (4.32) is between 20 ms and 28 ms for the tested round-trip times. Hence, the delay increase of 20 ms should not lead to a force magnitude that exceeds the area of transparency. The simulation results for $t > 2.8\,\mathrm{s}$ confirm this theoretical results.

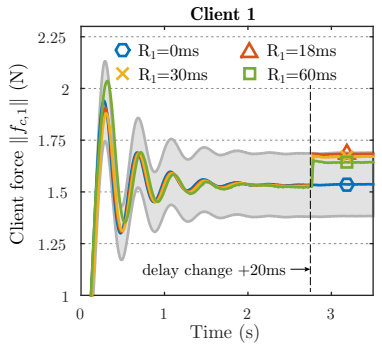
Remark Lawrence [149] defines bilateral teleoperation systems as transparent if the mechanical impedance of the remote environment matches the impedance displayed to the user. In fact, matching the steady-state force at the client in the presence of communication delay to the zero-delay reference force corresponds to matching the steady-state mechanical impedances ($Z_{c,i} = \frac{F_{c,i}(s)}{\dot{X}_{d,i}(s)}$). If the external force $F_e(s)$ in the LTI model in Figure 4.3 is modeled as a viscous friction force, the input velocity $\dot{X}_{d,i}$ cancels out when the impedance values are equated. Figure 4.7 shows the bode magnitude plots for the cases without and with compensation enabled at the clients. For low-frequency inputs, the magnitude of the frequency response increases with increasing RTT. If the delay compensation is enabled, the steady-state magnitudes match to the zero-delay reference.



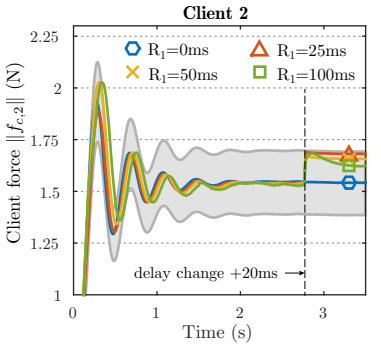
(a) Client 1: no compensation.



(b) Client 2: no compensation.

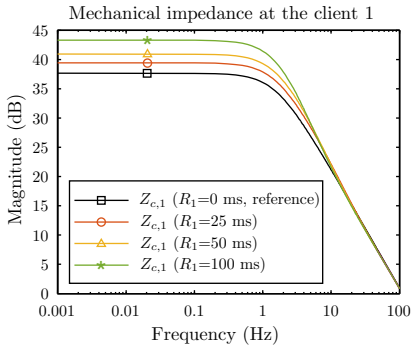


(c) Client 1: with compensation.

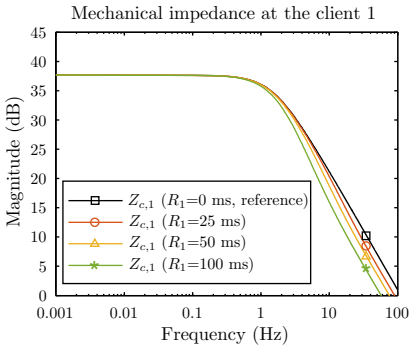


(d) Client 2: with compensation.

Figure 4.6: Rendered forces in the implemented SHVE setup for simulated user inputs with constant velocity. (a), (b): Haptic transparency is lost with increasing communication delay if no compensation scheme is used. The *area of perceived transparency* (grey area) denotes a force magnitude change of $c = \pm 10\,\%$ compared to the zero delay reference. (c), (d): Delay compensation is enabled. The steady state force is matched to the reference force. There is again a mismatch in the steady state force if a delay change occurs and the stiffness is not adapted accordingly (adopted from [2]).



(a) No compensation.



(b) With compensation.

Figure 4.7: Bode magnitude plot of the mechanical impedance $Z_{c,1}$ at the client for a single-user interaction and a viscous friction contact force model (adopted from [10], © 2014 IEEE).

4.2.4.2 Subjective evaluation with constant delay

The previous simulation results show that the proposed scheme perfectly compensates the effect of delay on the steady-state forces at the clients. As the delay compensation scheme is based on a steady-state analysis, it is not expected that the force rendered at the clients always perfectly matches the force without communication delay. Also, the human-in-the-loop has not been considered so far. The following experiment (denoted as the *constant delay experiment*) was performed to subjectively evaluate the performance of the proposed scheme to hide the effect of constant communication delay to the users.

Method The constant delay experiment was based on the following methodology.

Apparatus The subjective evaluation was performed with a single user only, because the cooperative interaction of two users would lead to uncontrollable conditions during the experiment. A subject cannot compare the stimuli reliably, because the difference between them could be a result of the second user's interaction, the delay simulation, or the compensation scheme being enabled or disabled.

Delay in the communication network was emulated with half of the round-trip time R_1 spent on the forward channel, and half on the backward channel: $T_{f,1} = T_{b,1} = \frac{R_1}{2}$. The server and the client run on the same machine to avoid any additional delay in the communication between them. The virtual environment contained a single cubic rigid object, which was situated on a rigid ground. Coulomb friction was simulated between the ground and the cube. The motion of the cube, which was free to move and rotate in 3-DOF each, was simulated on the server (mass m_o = 0.25 kg, damping $d = 0.999$). The original stiffness value was set to 700 N/m: $k^o_{c,1}$ = $k_{s,1}$ = 700 N/m. Together with a haptic rendering rate of 1 kHz, this stiffness setting resulted in a stable haptic interface. The client was equipped with a Geomagic Touch haptic device [79].

Task and procedure In a single trial, the subject had to interact with the cube and rate the perceived weight of the object on a scale from 1 (light) to 5 (heavy). Note that the mass of the object was actually not changed between trials. Only the communication delay changed, which led to increased force magnitudes at the client and the perception of increased weight of the object. At any time, the user could switch between the current stimuli, that is, the current evaluation cube, and two reference cubes, corresponding to ratings of 1 and 5, respectively, by pressing keys on the keyboard. This experimental procedure allowed the subjects to interact freely with the object and to compare the evaluation cube to the two references at one's convenience.

Stimuli The communication delay R_1 varied between $R_1 = \{0,\ 10,\ 25,\ 50,\ 75,\ 100,\ 125,\ 150\}$ ms. The different delay settings were evaluated with the proposed compensation scheme being turned off (denoted as stimuli $S^{NC}_{R_1}$, NC: no compensation) and turned on (denoted as stimuli $S^{WC}_{R_1}$, WC: with compensation). The settings S^{NC}_0 and S^{NC}_{150} corre-

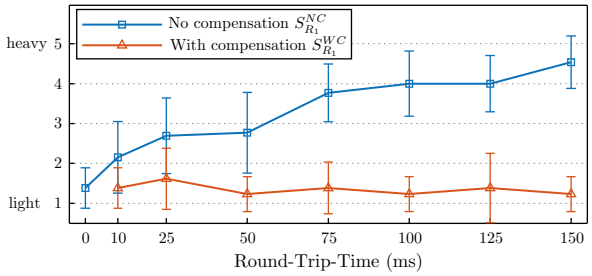


Figure 4.8: User ratings and their standard deviation for weight perception in the constant delay experiment (adopted from [10], © 2014 IEEE).

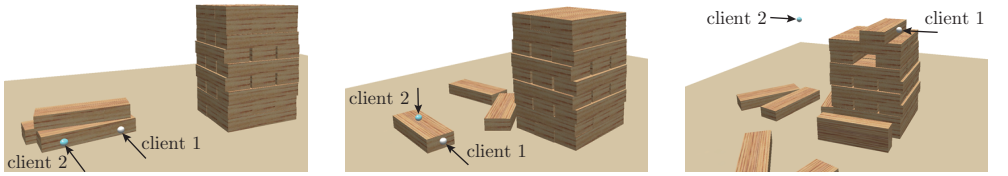
sponded to reference 1 (light) and 5 (heavy), respectively. In total, 15 trials[1] were evaluated by each subject. They were presented randomly to avoid any systematic bias in the results and prediction of trends.

Participants Fourteen subjects (12 male, 2 female) participated in the experiment. Four subjects were daily users of haptic devices, two were novice users, and the others already participated in past experiments in our lab. A training session was conducted with each participant to ensure proper knowledge of the experimental procedure and the references. They were instructed to interact freely with the objects, without specifying any task, but were advised to be consistent between different trials.

Results and discussion The average user ratings and their standard deviation are shown in Figure 4.8. The data is not normally distributed according to D'Agostino-Pearson's K2 test. Hence, statistical significance in the results was evaluated with the non-parametric Kruskal-Wallis test [220] (significance level of $5\,\%$). Without delay compensation, the perceived weight increased significantly with increasing delay ($\chi^2(7) = 65.68, p < 0.001$). Reference 1 was recognized on average with an error of around 0.38, while reference 5 was recognized with an error of 0.46. For a RTT of R_1 = 10 ms (the perceivable communication delay if $c = 10\,\%$ is assumed), the subjects already rated the weight on average with 2.15 and, thus, the rating is increased by 0.77. Post-hoc statistical analysis using Bonferroni correction showed, however, that this increase is not significant. Note that the definition of the perceivable communication delay is based on the Weber-fraction, which represents the JND (see Section 2.1.3) and, hence, denotes a force magnitude change which is as often perceived by the subjects as it is not. Thus, a statistically significant difference is not expected.

The results presented in Figure 4.8 for the trials with the proposed scheme being enabled show that the compensation scheme reduces the average user ratings close to reference 1 (R_1 = 0 ms). The difference between the $S_{R_1}^{NC}$ and the $S_{R_1}^{WC}$ trials is always found to be significant. Thus, the proposed scheme successfully compensates the effect of perceived increased weight due to communication delay in the CS-State architecture. The Kruskal-Wallis test indicated no significant increase/decrease in the user ratings depending on the delay, suggesting the usefulness over the complete tested delay range.

[1] The settings S_0^{NC} and S_0^{WC} are identical and, hence, there are only 15 trials.



(a) Cooperative pushing (b) Pushing and pressing (c) Pushing from opposite sides

Figure 4.9: The Jenga game used for evaluation of the proposed delay compensation scheme. (a), (b) and (c) illustrate the interactions in Figure 4.10, Figure 4.11 and Figure 4.12.

4.2.4.3 User interactions with time-varying communication delay

The *Jenga game* shown in Figure 4.9 was implemented to evaluate the proposed scheme with more complex interactions in the presence of time-varying delay. This application necessitates precise interaction with the objects, constitutes different contact scenarios between objects, and is suitable for single-point of contact haptic devices. The mass of each block was set to $m_o = 0.3\,\mathrm{kg}$. The coulomb friction coefficient in the physics simulation was set to $\mu = 0.45$, damping in the physics simulation was set to $d = 0.995$, and the original stiffness at the clients and the server was set to $k^o_{c,i} = k_{s,i} = 700\,\mathrm{N/m}$.

Time-varying communication delay was simulated as a piecewise constant function. Both the forward and the backward channel consisted of a deterministic delay (e.g., propagation delay). On top of this, additional stochastic delay (e.g., queuing delay) was simulated. The choice of a piecewise constant function was motivated by the fact that packet-reordering in Internet UDP traffic is rare [232]. Buffers and playout-scheduling can always be used to rearrange out-of-order packets to some extent [163], [233], leading to a piecewise constant playout delay. The stochastic delay had a gamma distribution, which describes common end-to-end delay in Internet communication [234].

This section presents force signals that have been recorded during three different real user interactions, which are illustrated Figure 4.9. With the human-in-the-loop it is not possible to repeat exactly the same interaction several times. Hence, a sample-by-sample comparison of the force signals is not feasible. The signals, however, still allow for a qualitative evaluation of the proposed scheme.

Cooperative pushing (see Figure 4.9a) The recorded signals in Figure 4.10 show that the proposed compensation scheme successfully compensates for the effect of constant communication delay on the force magnitude for the exemplary interaction already shown in Figure 4.1. The profile of the force magnitude during the interaction with zero communication delay (see (a) in Figure 4.10) is very similar to the one in the presence of delay and compensation enabled (see (c) in Figure 4.10). Without compensation, the magnitude of the rendered forces is clearly increased for both users (see (b) in Figure 4.10).

Pushing and pressing (see Figure 4.9b) The signals in Figure 4.11 are recorded while client 1 pushes a single block. During the movement, client 2 presses on the top surface of the block, which leads to an increased normal force and, hence, an increased friction force

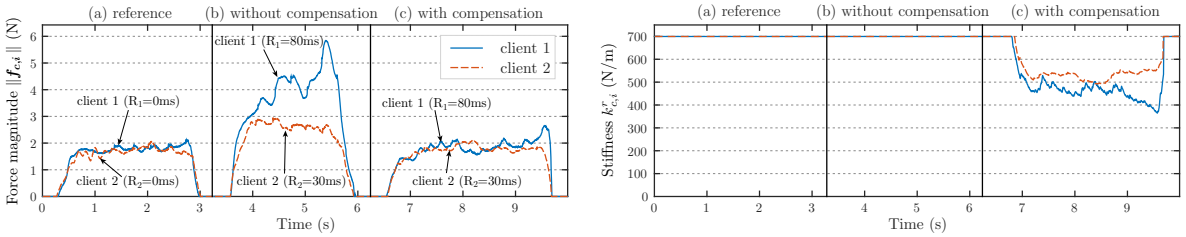


Figure 4.10: Cooperative pushing (see Figure 4.9a): magnitude of the rendered forces (left) and the stiffness (right) at client 1 and client 2 during the interaction in Figure 4.9a with zero communication delay (a), in the presence of constant communication delay without delay compensation (b) and with delay compensation enabled (c) (adopted from [2]).

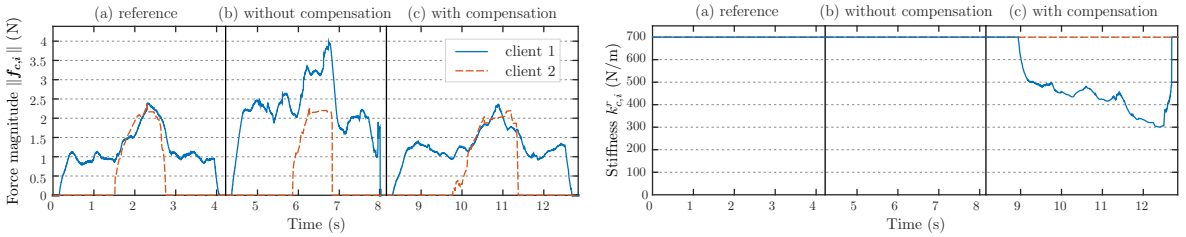


Figure 4.11: Pushing and pressing (see Figure 4.9b): magnitude of the rendered forces (left) and the stiffness (right) at client 1 and client 2 during the interaction in Figure 4.9b with zero communication delay (a), in the presence of time-varying communication delay without delay compensation (b) and delay compensation enabled (c) (adopted from [2]).

simulated in the physics simulation. The increased friction force is also perceived by client 1. As the contact forces are included in the object state updates transmitted to the clients (see Section 4.2.3.4), the client 1 can adapt accordingly. Again, the force signals recorded with simulated delay and compensation enabled (see (c) in Figure 4.11) are very similar to the reference (see (a) in Figure 4.11). The communication delay has no effect on client 2. He presses on the top surface and, hence, the stiffness is not reduced due to the motion constraint (see Section 4.2.3).

Pushing from opposite sides (see Figure 4.9c) The signals in Figure 4.12 are recorded while client 1 pushes a single block. After some time, client 2 presses from the opposite side. The object comes to rest as both forces are counteracting and have equal magnitudes. In this special case of the object not moving, the communication delay has no effect on the force magnitudes as one can observe from Figure 4.12. Note that one assumption of the derivations in Section 4.2.2 and Section 4.2.3 was that the object is moving in a steady state. The proposed scheme, however, also inherently captures such a scenario (see (c) in Figure 4.12).

4.2.4.4 Subjective evaluation with time-varying delay

The following experiment, denoted as the *time-varying delay experiment*, was used to test if the users can distinguish between a non-delayed interaction and an interaction in the presence of time-varying communication delay in the CS-State architecture.

Method The time-varying delay experiment adopted the *same-different procedure* [61], which is based on pairwise stimuli comparisons.

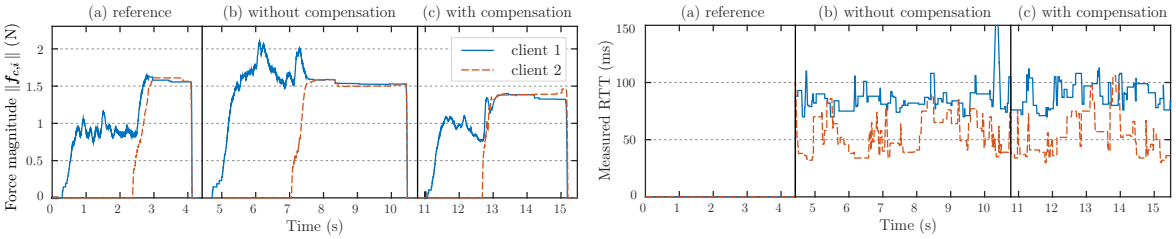
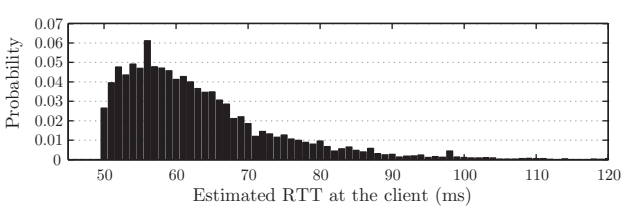
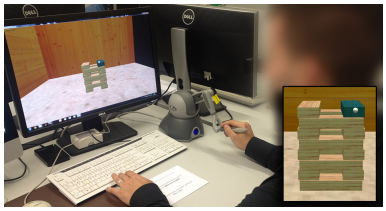


Figure 4.12: Pushing from opposite sides (see Figure 4.9c): Rendered forces at the clients (left) during the interaction in Figure 4.9c with simulated time-varying delay (right) at the clients. Zero communication delay (a), in the presence of time-varying communication delay without delay compensation (b) and with delay compensation enabled (c) (adopted from [2]).



(a) Histogram of the simulated delay during the experiment.



(b) Experimental setup.

Figure 4.13: (a) Histogram of the time-varying RTT at the client during the experiment. It can be seen that it follows approximately a gamma distribution. (b) The experimental setup, including a zoomed in version of the VE, used for the subjective evaluation (adopted from [2]).

Apparatus The SHVE with the Jenga game application, which already served as a basis for the evaluation of different human user interactions (Section 4.2.4.3), was adopted for the time-varying delay experiment. The experiment was performed with a single user for the reasons explained earlier in Section 4.2.4.2.

Time-varying delay was again simulated as piecewise constant delay as described in Section 4.2.4.3 with a deterministic round-trip delay of 50 ms. The delay histogram in Figure 4.13a illustrates the delay simulation during the subjective experiment.

Task and procedure In a single trial, the subject compared the two top blocks by pushing them one by one from the stack of blocks (see Figure 4.13b) and by answering the following two questions: 1) "Did you perceive the blocks to be of the *same* or *different* weight?" and 2) "Did you perceive any other artifacts? *Yes* or *No*?". All responses were written down by the experimenter. In case other artifacts were perceived, the subject was asked to explain them verbally. This additional question was posed in order to evaluate if there is any artifact, which was consistently perceived by the subjects.

This task necessitated the initial acceleration of the blocks, that is, it included a transient state. Hence, it is suitable to evaluate if the analysis based on the steady state limits the applicability of the proposed approach. The camera in the virtual environment and the workspace of the device had been rotated after each stimuli pair so that the user movement was always in the same direction. After nine comparisons were finished, the tower was rebuilt and the subject started again from the top.

The procedure was first verbally explained and demonstrated by the experimenter. Then, a training session with at least 18 comparisons was conducted with each subject. During the

Stimuli pair	Same	Different	Number of trials
S_0^{NC}–S_0^{NC}	301	59	360
S_0^{NC}–S_{TV}^{NC}	22	338	360
S_0^{NC}–S_{TV}^{WC}	265	95	360

Table 4.1: Answers summed over all subjects (adopted from [2]).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Overall
d'_{NC}	4.62	4.10	3.80	4.62	3.85	4.89	3.97	3.29	3.76	5.81	4.95	4.95	3.8	4.95	3.97	4.36±0.67
d'_{WC}	1.70	0.84	0.93	0.93	0.78	0.88	0.00	0.84	0.00	2.12	1.32	0.81	0.93	1.85	0.58	0.97±0.59

Table 4.2: Calculated d' values for the individual subjects, as well as the overall mean and standard deviation between them (adopted from [2]).

training, the subject was informed that the object weight varied, but also pairs with the same weight were included. Note that the weight was actually not changed between the stimuli. Only the delay simulation and the delay compensation was enabled or disabled (see below). As a consequence, the force feedback increased eventually, which was perceived as increased weight.

Stimuli The subjects had to discriminate between the interaction in the non-delayed reference case (stimulus S_0^{NC}, 0 ms delay, NC: no compensation) and in the presence of time-varying delay without delay compensation (stimulus S_{TV}^{NC}, TV: time-varying delay) or in the presence of time-varying delay with delay compensation enabled (stimulus S_{TV}^{WC}, WC: with compensation). The tested stimuli pairs were S_0^{NC}–S_0^{NC}, S_0^{NC}–S_{TV}^{NC} or S_0^{NC}–S_{TV}^{WC}. The order of the stimuli within a pair and the order of the pairs was randomly selected. Each subject compared 72 pairs in total (24 per stimuli pair). A break was enforced after half of the comparisons.

Participants Fifteen voluntary subjects participated in the experiment (13 male, 2 female, aged between 22 and 45). Four were frequent users of haptic devices, six had already participated in former experiments in our lab, and the others were novice users.

Results The same-different answers summed over all subjects are shown in Table 4.1. The answers of each subject were analyzed using signal detection theory (SDT) to achieve robustness towards uncertainties in human decision making and response bias [235]. The differencing strategy was assumed as the subjects' decision rule [235], because several pairs of different stimuli were compared [61].

The d' values given in Table 4.2, which are a measure of the subjects' sensitivity to the difference between two stimuli, were determined for the comparison pair S_0^{NC}–S_{TV}^{NC} (d'_{NC}) and for the pair S_0^{NC}–S_{TV}^{WC} (d'_{WC}). Note that both d' values were calculated with respect to the reference pair S_0^{NC}– S_0^{NC}. The d' values were determined using the table in [235]. Higher values of d' indicate higher discriminability. Two stimuli are often assumed to be indistinguishable for a threshold value of $d' \leq 1.0$ [61]. The averaged sensitivity indices are

calculated as $\bar{d}'_{NC} = 4.36 \pm 0.67$ and $\bar{d}'_{WC} = 0.97 \pm 0.59$. The data is normally distributed according to the chi-squared goodness of fit test. Hence, a two-sample t-test [219] (significance level of 5%) was performed and a statistically significant difference between the d'_{NC} and d'_{WC} group means can be reported ($t(28) = 14.66$, $p < 0.001$).

With respect to the second question, the subjects reported that they did not feel any other artifact for most of the trials. Subjects sometimes reported some kind of "discontinuities" or "jumps" in the force feedback for stimuli S_{TV}^{NC} or S_{TV}^{WC}, which could originate from large delay changes as discussed in the following. Three subjects reported that stimuli S_{NC} sometimes were of "varying weight', which is also attributed to the time-varying delay.

Discussion of the experimental results The subjective evaluation results show the effectiveness of the proposed scheme to compensate for the effects of communication delay. The subjects could barely distinguish the weight of the objects in the comparison of the stimuli without delay (the reference stimuli S_0^{NC}) and the stimuli with time-varying delay and delay compensation enabled (stimuli S_{TV}^{WC}).

But still, S_0^{NC}–S_{TV}^{WC} pairs were less often rated to be the same as S_0^{NC}–S_0^{NC} pairs. An analysis of the signals recorded during the experiment did not reveal an obvious reason for this. Besides general uncertainty in human decision making, large delay changes could be a main reason. The smooth stiffness adaptation (see Section 4.2.3.4) might lead to a discrepancy between desired and actually displayed stiffness.

The artifacts, which the users described as "discontinuities" or "jumps", probably stem from large delay changes. The transmission rate of state updates was kept at the rate of the haptic loop of 1 kHz. If the delay increases on the forward or the backward channel, the server or client won't receive new state updates for the amount of time the delay has increased. In this case, the client/server holds the last received state until the next update is received. If the delay decreases, the server or client will have several state updates in their receiving buffer. In this case, they do not display the latest update directly. Instead, they smoothly converge to the latest received state to avoid big jumps in the object/device state. Large delay changes might still lead to abrupt changes in the force feedback. Interestingly, 76.9 % of the "discontinuity"/"jump" answers were given for S_{TV}^{NC} trials (S_{TV}^{WC}: 23.1 %). This suggests that the reduced stiffness during the S_{TV}^{WC} trials helps to hide the delay changes. Buffering at the client [157] can be considered to avoid the aforementioned effects of delay changes. This comes, however, at the cost of an increased overall delay, but the proposed compensation scheme then does not need to adapt the stiffness to delay changes. On the other hand, a small playout delay should be always preferred as large delay leads to reduced operability [43].

No subject reported the impression of interacting with a softer object as an artifact during the interaction. At the beginning of the contact, the original stiffness is used. The stiffness is reduced only after the rendered force exceeds the static friction force (see Section 4.2.3.4). This is probably be the reason why the reduced stiffness is well hidden from the user.

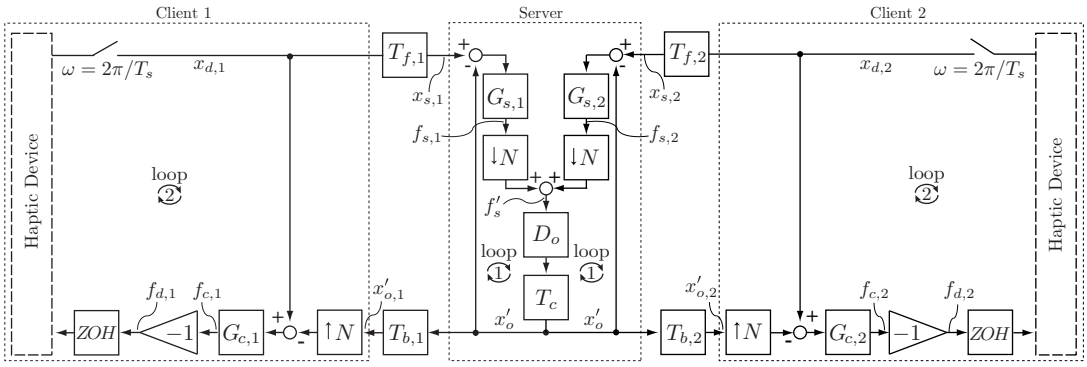


Figure 4.14: Two-user multi-rate model of the client/server architecture for the interaction with deformable objects. Signals marked with a prime are low-rate signals, sampled at a rate reduced by a factor N compared to the original sampling rate $1/T_s$. $G_{c,i}$ and $G_{s,i}$ denote the force rendering algorithms at the clients and the server for user i, respectively. On the server, D_o represents the computationally complex deformable object simulation and T_c denotes its computational delay (adapted from [1], © 2016 IEEE).

4.2.4.5 Discussion

In summary, the evaluation shows that the proposed scheme effectively hides the effect of communication delay from the user, although the derivation of the scheme is based on the steady state only.

With the proposed scheme, however, the object still does not move immediately at the initial contact with an object. For large delay values, this becomes noticeable haptically and visually, and the operability of the system reduces. For this reason, the proposed scheme is preferably applied only for delay that is not visually observable by the user.

Alternatively, a local physics simulation could be applied at the client to predict the object movement until the valid object state is received from the centralized server. This constitutes an alternative way to compensate for the delay in the CS-State architecture. This local-prediction approach is conceptually similar to the time warp algorithm [154], which is used in distributed simulations to ensure consistency. However, this approach suffers from possibly large inconsistencies that need to be resolved at the clients and, as a result, perceivable artifacts in the force feedback.

4.3 Interaction with deformable objects

4.3.1 Two-user client/server model

Figure 4.14 shows a model of the CS-State architecture, which represents the interaction of two clients with a one-dimensional deformable object, which is simulated at a lower rate compared to the force rendering algorithm (see Section 2.3.4.2). Analogous to the LTI model presented in Section 4.2.1, the intention is to establish a mathematical representation of multi-user interactions with deformable objects. Based on this model, Section 4.3.2 analyzes the transparency of the SHVE and Section 4.3.3 presents a scheme to compensate for the effect of communication delay on the forces rendered at the clients (see Section 4.1).

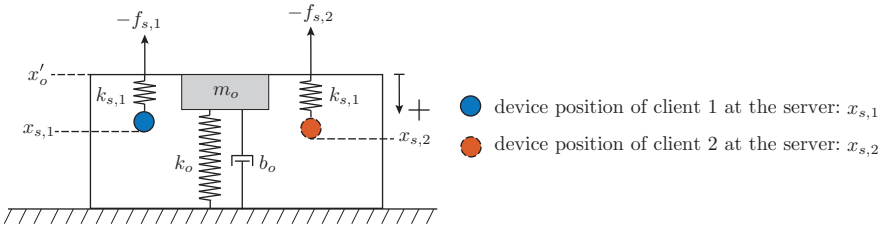


Figure 4.15: Mechanical model of the 1D mass-spring-damper deformable object (adopted from [1], © 2016 IEEE).

Client i transmits the haptic device position $x_{d,i}$, sampled at rate $1/T_s$, to the server over the communication network, denoted with the communication delays $T_{f,i}$ (forward channel for client i) and $T_{b,i}$ (backward channel for client i).

The block $G_{c,i}$ denotes the high-rate (rate $1/T_s$) constraint-based force rendering algorithm at client i, which uses an intermediate model of the deformable object (see Section 2.3.4.2). The clients receive the delayed object surface position, $x'_{o,i}$, from the server. Signals marked with a prime are low-rate signals, sampled at a rate reduced by factor N compared to the original sampling rate $1/T_s$. The local force rendering at the clients ensures that there is no delay in the control loop between the haptic device and the VE. The blocks $G_{s,i}$ denote the high-rate force rendering algorithms at the server. The transfer functions of the spring-based force rendering algorithms are given as:

$$G_{s,i}(z) = k_{s,i} \tag{4.33}$$

$$G_{c,i}(z) = k_{c,i} + b_{c,i}\frac{z-1}{zT_s} \tag{4.34}$$

The client-side force rendering algorithm, $G_{c,i}$, employs the virtual damping element $b_{c,i}$ to ensure a stable coupling between the haptic device and the VE (see Section 4.3.2.1).

The intermediate models at the clients and the server are updated at a rate of $1/(NT_s)$. The force rendering stiffness $k_{s,i}$ together with the current object surface position x'_o at the server and the force rendering parameters $k_{c,i}$ and $b_{c,i}$ together with the delayed surface position $x'_{o,i}$ at the clients fully describe the intermediate models for the 1D deformable object.

The block D_o denotes the deformable object simulation running at a rate reduced by factor N (rate $1/(NT_s)$). $T_c = z^{-1}$ denotes the one-step computational delay of the deformation simulation. Together, these blocks close the multi-rate deformation simulation on the server, which is discussed in [140] for a single-user interaction in a completely locally simulated haptic application. The mean of the force calculated at the server during one deformation simulation step can be used as input to the FEM simulation. If only the very last calculated force sample within one deformation simulation step is used instead, the force rendering on the server does not necessarily need to run at the high update rate.

The model of the deformable object is shown in Figure 4.15. The time-discrete transfer function for $D_o(z)$ is

$$D_o(z) = \frac{z^2}{(\frac{m_o}{T_n^2} + \frac{b_o}{T_n} + k_o)z^2 + (-\frac{2m_o}{T_n^2} - \frac{b_o}{T_n})z + \frac{m_o}{T_n^2}} \tag{4.35}$$

where m_o, k_o, b_o are the deformable object's mass, spring stiffness and damping coefficient, respectively, and $T_n = NT_s$ is the computation time for a single simulation step. All parameters of the 1D model are known, because the object parameters m_o, b_o and k_o and the force rendering parameters $k_{s,i}$, $k_{c,i}$ and $b_{c,i}$ are set in the application. The communication delays $T_{f,i}$ and $T_{b,i}$ can be measured. The rate $1/(NT_s)$ of the deformation simulation is also known.

4.3.2 System analysis

This section discusses the stability and analyzes the haptic transparency of the CS-State architecture based on the presented two-user model.

4.3.2.1 Stability of the CS-State architecture

The CS-State architecture is conceptually similar to so-called model-mediated teleoperation (MMT) [176]. In MMT, a model of the environment is estimated at the remote side (the server side) and transmitted to the operator side (the client side). Force feedback is then rendered locally without delay using the received model, making the system robust against communication delay [177]. In the CS-State architecture, the estimated environment model corresponds to the intermediate model of the deformable object.

Stability in a MMT system can be ensured if the stability of the coupling between the remote robot and the object in touch, as well as the stability of the closed-loop between the human operator, the local environment model, and the haptic device is guaranteed [177]. These conditions translate to the stability of the simulation loop at the server and the force rendering loop at the clients (denoted as loop 1 and loop 2 in Figure 4.14), which are briefly discussed in the following.

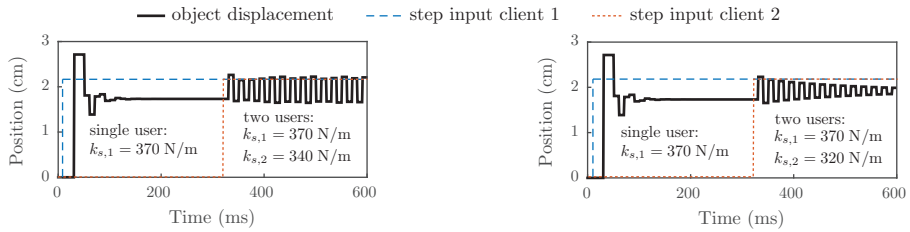
Stability of the multi-rate simulation on the server The stability of the multi-rate simulation employed on the server (see loop 1 in Figure 4.14) is discussed by Barbagli *et al.* [140] for a single-user interaction with a spring-damper object. They show, using lifting techniques, that the analysis of the closed loop 1 in Figure 4.14, considering only $G_{s,i}$, D_o and T_c (i.e., neglecting the decimator) is sufficient to analyze the stability of the multi-rate deformable object simulation. Hence, the closed-loop transfer function

$$G(z) = \frac{G_{s,i}(z)D_o(z)T_c(z)}{1 + G_{s,i}(z)D_o(z)T_c(z)} = \frac{k_{s,i}\ z}{(\frac{m_o}{T_n^2} + \frac{b_o}{T_n} + k_o)z^2 + (-\frac{2m_o}{T_n^2} - \frac{b_o}{T_n} + k_{s,i})z + \frac{m_o}{T_n^2}} \tag{4.36}$$

is investigated with Jury's stability criterion [236] to analyze the stability of the single-user multi-rate simulation on the server. This leads to the following criterion to achieve a stable simulation of the interaction with a mass-spring-damper object:

$$k_{s,i} < k_o + \frac{2b_o}{T_n} + \frac{4m_o}{T_n^2} =: k_{s,i}^{max} \tag{4.37}$$

The closed-loop transfer function in Equation (4.36) describes a single-user interaction, that is, only the left half of the model in Figure 4.14 has been considered.



(a) Step-response with unstable stiffness setting (b) Step-response with stable stiffness setting

Figure 4.16: Simulated step-response of the multi-rate simulation on the server. The simulation parameters are $T_s = 0.001$s, $N = 10$, $k_o = 100\,\mathrm{N/m}$, $b_o = 1.0\,\mathrm{Ns/m}$, $m_o = 0.01\,\mathrm{kg}$. The critical stiffness according to Equation (4.38) is $k_s^{max} = 700\,\mathrm{N/m}$. During the single user interaction between 0 ms and 320 ms, the simulation is stable for both cases. As soon as the second user applies a step-force, the simulation is stable only if the condition in Equation (4.38) is fulfilled (adopted from [1], © 2016 IEEE).

With M users interacting with the same object, the stiffness values of the parallel force rendering algorithms at the server (see Figure 4.15) add up and have to be considered in the stability condition:

$$\sum_{j=1}^{M} k_{s,j} =: k_s < k_o + \frac{2b_o}{T_n} + \frac{4m_o}{T_n^2} =: k_s^{max} \tag{4.38}$$

A stable simulation on the server requires the careful selection of the stiffness values $k_{s,i}$. The object characteristics k_o, b_o, m_o and the simulation rate $1/T_n$ influence the maximum possible stiffness to be used in the force rendering at the server. The object mass and damping contribute to a higher value of k_s^{max}.

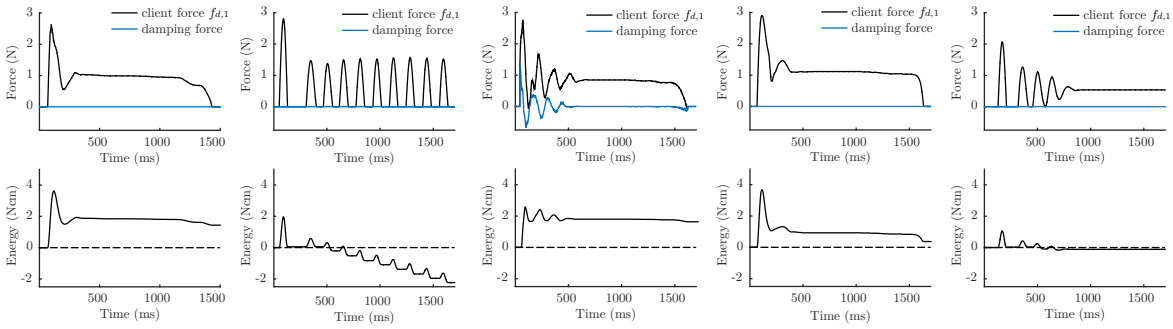
Figure 4.16 shows the simulated step-response of the multi-rate simulation on the server with $k_{s,i}$ chosen to be slightly above and below the critical stiffness k_s^{max}.

Stability of the local force rendering at the clients The force rendering loop at the clients (see loop 2 in Figure 4.14) closes the loop between the haptic device and the copy of the VE locally. Hence, the client resembles a standalone haptic VE, for which stability is often investigated based on the concept of passivity (see Section 2.2.2). Passivity requires that the energy injected into the system is greater than the energy extracted from it at all times [89].

The force rendering algorithms at the clients use the intermediate model to calculate force feedback at the required high rate to avoid stability issues due to sampling [82]. The local rendering also ensures that there is no delay in the haptic rendering loop.

However, delayed object deformation updates at the client, that is, delayed position updates of the intermediate models might lead to an active VE and artificially inject energy in the local rendering loops [237].

The virtual damping elements $b_{c,i}$ can be applied in the local force rendering at the clients to dissipate energy and achieve stability. The damper can be fixed or adapted to the current scenario to dissipate only the required amount of energy [89]. In the following transparency analysis, the damping elements $b_{c,i}$ are assumed to be employed to ensure a stable force rendering at the clients.



(a) $R_1 = 0\,\text{ms}$ (b) $R_1 = 100\,\text{ms}$ (c) $R_1 = 100\,\text{ms}$ (d) $R_1 = 100\,\text{ms}$ (e) $R_1 = 100\,\text{ms}$

Figure 4.17: Force signals commanded to the haptic device and observed energy at a client for client/server-based interactions with a mass-spring-damper object. (a) Stable interaction for a loose grasp of the haptic device for zero communication delay. (b) Delayed intermediate model updates lead to instability for a loose grasp of the haptic device. (c) A virtual damping element stabilizes the interaction for a loose grasp in the presence of delay. (d) For a normal grasp, the interaction is stable in the presence of 100 ms communication delay even without additional virtual damping, because the human arm impedance also contributes to system stability. (e) Stable interaction for a loose grasp and in the presence of communication delay. Compared to (b), the object stiffness k_o is higher (partially adopted from [1], © 2016 IEEE).

Passivity is a conservative design requirement, because the human user might not be able to destabilize the system, even if the coupling between the haptic device and the local VE is active [82]. This is because the human arm impedance also dissipates energy and contributes to system stability [94].

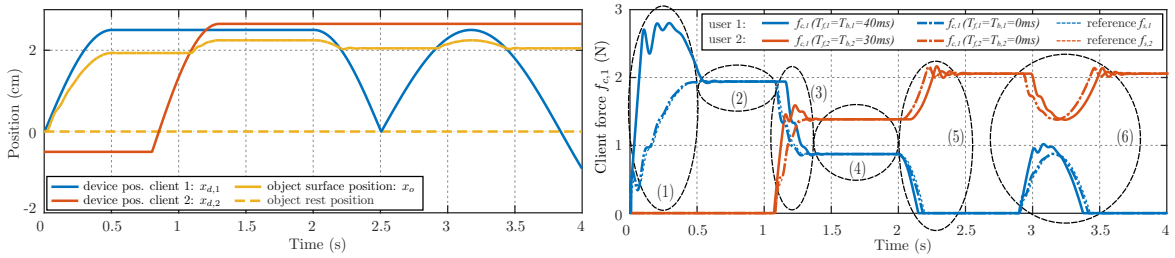
Figure 4.17 exemplarily illustrates the effects of communication delay, hand grip, damping, and object stiffness on the stability of the client/server architecture. The stiffness values for the shown interactions are selected rather extreme to enforce destabilizing effects for illustration purposes, but still satisfying Equation (4.38), with higher values for $k_{c,1}$ (700 N/m) than for $k_{s,1}$ (250 N/m) and k_o (350 N/m). Note that transparency, as discussed in the following section, requires $k_{c,1}$ to equal $k_{s,1}$.

The client/server based interaction is stable for a loose grasp of the haptic device for zero communication delay (Figure 4.17a). The observed energy, calculated with a Passivity Observer [89] (see Section 2.2.2), stays above zero, that is, the haptic rendering at the client is passive.

For a loose grasp in the presence of communication delay, the haptic device starts to oscillate and the system becomes unstable, because the delay leads to out-of-phase intermediate model updates and the human impedance is nearly zero (Figure 4.17b). The observed energy becomes negative, that is, the local rendering is active. The virtual damping element $b_{c,i}$ stabilizes the CS-State architecture in this case (Figure 4.17c).

If the user applies a normal grasp in the presence of 100 ms communication delay (Figure 4.17d), the system is stable even without additional virtual damping ($b_{c,1} = 0$), because the impedance of the human arm contributes to system stability.

If the object stiffness k_o is increased to 1000 N/m, the system is also stable for a loose grasp in the presence of 100 ms communication delay and without additional virtual damping (Figure 4.17e), although the local rendering is slightly active. The position updates of the



(a) Device and object surface positions

(b) Force signals at the clients

Figure 4.18: Illustration of the effects of delay during a two-user interaction. The simulated client device trajectories in (a) lead to the rendered force signals in (b) and, as a result, to the displacement of the object surface position. In the presence of communication delay, the force magnitude increases for the client that actively presses on the object. The simulation parameters are $k_o = 100\,\mathrm{N/m}$, $b_o = 2\,\mathrm{Ns/m}$, $m_o = 0.1\,\mathrm{kg}$, $k_{c,1} = k_{s,1} = 340\,\mathrm{N/m}$, $b_{c,1} = 5\,\mathrm{Ns/m}$ (adapted from [1], © 2016 IEEE).

local intermediate model at the client become smaller for a higher object stiffness rendered at the server. Hence, they can contribute less to system instability. In the theoretical case of an infinitely stiff object on the server, no updates of the local model at the client are necessary and the client resembles a standalone HVE.

Since passivity-ensuring control and the selection of proper damping values has been investigated in detail, for example, in [82], [89], [97], [237], the following analysis does not focus on the selection of suitable values for $b_{c,i}$ to ensure stability at the clients. Instead, stability-ensuring damping elements are assumed and considered in the transparency analysis accordingly.

4.3.2.2 Realism of the deformable object simulation

The selection of $k_{s,i}$ influences the quality, that is, the physical realism of the multi-rate simulation at the server in the transient and in the steady state [140]. The simulated object position tends to oscillate and the settling time increases the closer the selected stiffness values are to k_s^{max}. Hence, small values of $k_{s,i}$ should be chosen. On the other hand, $k_{s,i}$ should be chosen as large as possible to limit the device penetration into the object and to display the intended object stiffness k_o to the user [140].

The algorithm presented by Barbagli *et al.* [140] gracefully increases the stiffness $k_{s,i}$ in the steady state to find a trade-off between these two contradicting implementation requirements. Fixed stiffness values for $k_{s,i}$ that ensure a stable multi-rate simulation on the server are assumed throughout this thesis.

4.3.2.3 Transparency of the client/server architecture

The effects of delay in a two-user interaction Figure 4.18 plots the force and position signals during a simulated interaction of two users with a 1D mass-spring-damper deformable object in the CS-State architecture. It illustrates the effect of communication delay on the forces rendered at the clients.

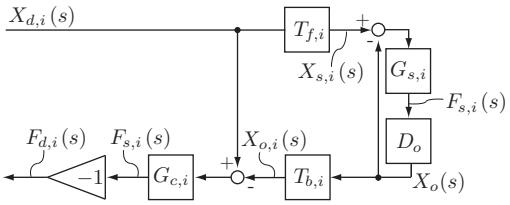


Figure 4.19: Time-continuous LTI system for a single client i. $D_o(s)$ describes the 1-DOF deformable object in Figure 4.15 (adapted from [9], © 2015 IEEE).

At first, only client 1 pushes on the object. The communication delay leads to a peak in the rendered force magnitude (compare the solid and dashed line in the section marked with (1) in Figure 4.18b). In the steady state, while the client 1 holds still, there is no effect of the communication delay (2).

Then, client 2 additionally presses on the object (3). There is a peak in the rendered force magnitude for client 2. There is no effect on the force magnitude of client 1, except some delay, which stems from the communication between the clients and the server. There is no force error in the steady state for both clients (4).

The force increases for client 2 with some delay as client 1 stops pressing on the object (5). For client 1, the change in object deformation is of course also delayed while he/she removes the haptic device from the virtual object. This leads to a slightly steeper force reduction during the removal compared to the non-delayed reference. Later, client 1 presses again on the object (6). There are again the previously mentioned artifacts for the *acting* client 1, but no artifacts for the *observing* client 2 (except the inevitable delay, of course).

In summary, the communication delay in the CS-State architecture leads to an increased force magnitude during the transient state for the client that actively presses on the deformable object. There is no effect in the steady state. The passively observing client renders the same force magnitude compared to the non-delayed reference, but with some delay. The force magnitude decreases faster in the presence of delay while removing the haptic device from the object.

These observations motivate to analyze the transparency of the CS-State architecture for a single user only.

Ideal transparency The time-continuous LTI system in Figure 4.19 is used to analyze the effect of constant communication delay on the haptic transparency for client i. In the following, the capital letters denote signals in the Laplace domain. This LTI system allows for a convenient analysis of the effect of delay on the transparency of the CS-State architecture. Sampling blocks like the decimator and expander in Figure 4.14 are inherently not included in the time-continuous system. Similar to other transparency analyses (e.g., [153]), the haptic device is not included in the model. This allows us to investigate the effect of communication delay on the rendered forces, which is independent of the device dynamics.

The communication delay until the local model is updated at the client i is encapsulated in $R_i = T_{f,i} + T_{b,i}$ in the following. The delay elements $T_{f,i}$ and $T_{b,i}$ can be combined because of linearity. The computational delay is neglected in the LTI model and in the following anal-

ysis, because it is present in a fully local simulation and in the distributed client/server-based simulation.

The force $F_{c,i}(s)$ rendered at the client can be derived from Figure 4.19 with the transfer functions $G_{c,i} = k_{c,i} + b_{c,i}s$, $G_{s,i} = k_{s,i}$ and $D_o(s) = 1/(m_o s^2 + b_o s + k_o)$ as

$$\begin{aligned} F_{c,i}(s) &= (k_{c,i} + b_{c,i}s)\left[1 - e^{-sR_i}\frac{k_{s,i}}{m_o s^2 + b_o s + k_o + k_{s,i}}\right] X_{d,i}(s) \\ &= (k_{c,i} + b_{c,i}s)\left[1 - e^{-sR_i}K_i(s)\right] X_{d,i}(s) \end{aligned} \tag{4.39}$$

where $K_i(s)$ represents the fraction in Equation (4.39) to simplify the equations. The server force $F_{s,i}(s)$ is:

$$F_{s,i}(s) = k_{s,i}\left[1 - K_i(s)\right] X_{s,i}(s) \tag{4.40}$$

The client force $F_{c,i}(s)$ for zero communication delay is:

$$F_{c,i}(s)\big|_{R_i=0} = (k_{c,i} + b_{c,i}s)\left[1 - K_i(s)\right] X_{d,i}(s) \tag{4.41}$$

Ideal transparency as defined in [151] requires identical position and force signals on the server and the client. The client force (4.39) equals the server force (4.40) only if $k_{c,i}$ is set to the value of $k_{s,i}$, $b_{c,i}$ is zero, and if no communication delay exists ($R_i = 0\,\mathrm{ms}$). The stiffness setting with $k_{c,i}$ set to the value of $k_{s,i}$ is denoted as the *original stiffness setting* $k^o_{c,i}$.

In the presence of communication delay, the force and position signals differ, that is, ideal transparency as defined in [151] is not achievable. The damping $b_{c,i}$ is employed to ensure stability at the clients. The loss of transparency caused by $b_{c,i} \neq 0$ is a trade-off for stability.

Lawrence [149] has defined ideal transparency in the context of bilateral teleoperation systems as a match between the mechanical impedance of the remote environment and the locally displayed impedance. Similarly, this definition has been used in the context of SHVEs, for example, in [152]. Ideal transparency then requires a match between the impedance rendered at the server and the clients (see Section 2.4.1).

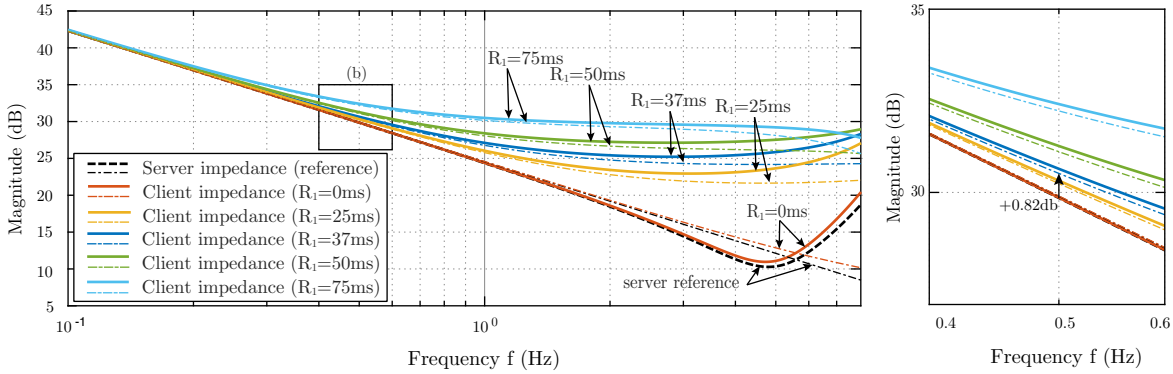
The mechanical impedances at the client i and the server are calculated as:

$$Z_{c,i}(s) = \frac{F_{c,i}(s)}{\dot{X}_{d,i}(s)}, \qquad Z_{s,i}(s) = \frac{F_{s,i}(s)}{\dot{X}_{s,i}(s)} \tag{4.42}$$

Figure 4.20 plots the magnitude of the impedance as a function of the input frequency f ($s = j\omega = j2\pi f$). The magnitude of the impedance rendered at the client ($|Z_{c,i}|$) increases with the delay and the input frequency compared to the server-side reference magnitude ($|Z_{s,i}|$). Ideal transparency as defined in [149] is lost.

Note that the client requires the original stiffness setting with $k_{c,i}$ set to $k^o_{c,i} = k_{s,i}$, because unequal stiffness settings would introduce a impedance mismatch even for zero delay and input frequencies close to zero.

Perceived transparency The concept of *perceived transparency* alleviates the aforementioned definition of transparency. If the displayed impedance in a bilateral teleoperation system is not distinguishable from the impedance of the remote environment, the system is said to be perceived transparent [153]. Similarly, if there is no perceivable difference between



(a) Impedance magnitude as a function of the input frequency (b) Zoomed version

Figure 4.20: (a) Magnitude of the impedance displayed to the user in the presence of communication delays (solid lines) and the server-side reference impedance (dashed line). The error between the impedance at the client and the server increases with the communication delay and the input frequency. The dashed-dotted lines show the low-frequency approximations using Equation (4.44). The simulation parameters are $k_o = 100\,\mathrm{N/m}$, $b_o = 2\,\mathrm{Ns/m}$, $m_o = 0.1\,\mathrm{kg}$, $k_{c,1} = k_{s,1} = 340\,\mathrm{N/m}$, $b_{c,1} = 5\,\mathrm{Ns/m}$. (b) The approximated impedance at the client increases by 0.82 dB for a delay of 37 ms ((a) adapted from [1], © 2016 IEEE).

a fully local (non-delayed) interaction with the deformable object and the interaction through the distributed system, the CS-State architecture is assumed to be perceived transparent.

Human perception of kinesthetic stimuli is known to be limited [61]. Empirically derived perception thresholds for the impedance of mass-spring-damper systems do not exist and are difficult to obtain [153].

Kinesthetic human movements have peak frequencies in the range of 1–8 Hz, where the upper limit of 4–8 Hz corresponds, for example, to playing a piano [50]. The input frequency for applications that require precise interaction, for example, surgical simulation, are rather in the area of 0.5–2 Hz [238]. The considered system shows a spring-like behavior for low input frequencies (see Figure 4.20). For this reason, the thresholds determined for human kinesthetic stiffness perception are assumed to describe the perception thresholds during the considered client/server-based interaction.

The perceivable difference compared to an initial stimulus is often described by the Weber-fraction c. For stiffness values, it is known to range between 13 and 28 % (see Section 2.1.3). With the definition of $a := 1 + c$, where c is the Weber-fraction, the maximum allowable impedance magnitude at the client is written as:

$$\left|Z_{c,i}(s)\right|^{max} = a\left|Z_{s,i}(s)\right| \tag{4.43}$$

To analytically analyze the effect of communication delay, a user input with a frequency ω, that is, the user poking on the object surface, is considered. For the aforementioned low-frequency input, the effect of m_o and b_o is assumed to be small [238] and $K_i(s)$ is approximated as

$$K_i(s) \approx \frac{k_{s,i}}{k_o + k_{s,i}} := K_i \tag{4.44}$$

in the following.

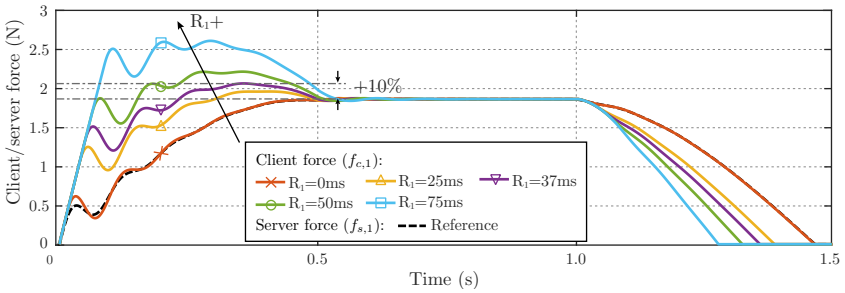


Figure 4.21: Force signals at the client 1 and the server reference force for a simulated device movement in the presence of communication delay. The predicted R_1^{max} using Equation (4.45) for the sine-like input is 37.7 ms ($\omega = 0.5\,\text{Hz}$, $c = 10\,\%$). The peak force increases by 10 % for this delay. The simulation parameters are $k_o = 100\,\text{N/m}$, $b_o = 2\,\text{Ns/m}$, $m_o = 0.1\,\text{kg}$, $k_{c,1} = k_{s,1} = 340\,\text{N/m}$, $b_{c,1} = 5\,\text{Ns/m}$ (adopted from [1], © 2016 IEEE).

Inserting Equation (4.39) and Equation (4.40) into Equation (4.42), equating the magnitudes of the mechanical impedances as in Equation (4.43) with the original stiffness setting ($k_{c,i}$ set to $k_{s,i} = k_{c,i}^o$) and $e^{-j\omega R_i} = \cos(\omega R_i) - j\sin(\omega R_i)$, and rearranging to R_i leads to ($\omega \neq 0$):

$$R_i^{max}(\omega) = \frac{\arccos\left(\frac{(1+K_i^2)(k_{c,i}^2+b_{c,i}^2\omega^2)-a^2k_{c,i}^2(1-K_i)^2}{2k_{c,i}^2K_i+2b_{c,i}^2\omega^2K_i}\right)}{\omega} \tag{4.45}$$

This predicts the communication delay for which the magnitude of the impedance rendered at the client increases by c percent. This result is confirmed in Figure 4.20b, where the magnitude of the approximated impedance at the client for $R_1 = 37\,\text{ms}$ increases by 0.82 db (i.e., 10 %) compared to the server reference. This delay corresponds to R_1^{max} for the used simulation parameters and $c = 10\,\%$. Hence, if c correctly describes the human perceptual limitation, R_i^{max} defines the tolerable communication delay for a sine-like input with known frequency ω. The tolerable communication delay reduces proportional to the input frequency, that is, less delay is acceptable for faster interactions. The Weber-fraction of $c = 10\%$ was selected for illustration purposes, which lies at the very low end of the experimentally found difference thresholds for stiffness [50]. Note that the approximation of $K_i(s)$ as K_i leads to an prediction error. For the used simulation parameters, the error is 0.14 db, that is, 1.6 %.

Figure 4.21 shows the rendered force signals at the client and the server for a simulated 1-DOF interaction in the CS-State architecture in the presence of different communication delays. The client first presses, then holds and finally releases the object. The movement frequency during pressing and releasing the object is $f = 0.5\,\text{Hz}$. The communication delay for which the impedance magnitude increases by c percent, calculated with Equation (4.45), is 37.5 ms. The peak force rendered at the client, that is, the force amplitude, increases by 10 % compared to the server reference force for the predicted delay value.

Humans perceive the stiffness of a deformable object from different cutaneous and kinesthetic cues (see Section 2.1.2). During the interaction, humans combine, for example, the sensed force with the displacement caused by their action, which is sensed by the visual and proprioceptive sensory modalities [57]. There is, however, no consensus about the most

important piece of sensory information used to estimate stiffness [57]. Pressmann et al. [60] tested different perception models and found that the model which divides the maximum force by the amount of penetration best explains the perceived stiffness.

The maximum force increases in the presence of communication delay compared to the zero-delay reference while pressing on the object (see Figure 4.18 and Figure 4.21). Hence, if communication delay exists and the user recognizes the increased force, she/he also presumably perceives an increased stiffness (i.e., impedance) during the client/server-based interaction with the deformable object.

The force magnitude also decreases slightly faster in the presence of delay while removing the haptic device from the object (see Fig. 4.21). Di Luca *et al.* [57] have found that the human sensitivity to stiffness is much higher during loading movements compared to unloading movements. Similarly, the artifacts while pushing on the object are perceived to be stronger. Hence, the next section focuses on the mitigation of the force magnitude peaks while pressing on the object to hide the communication delay from the users.

4.3.3 Compensating the effect of delay

This section presents a scheme to compensate for the effect of communication delay during the CS-based interaction with deformable objects. Similarly to the scheme for the interaction with rigid objects presented in Section 4.2, it adaptively adjusts the stiffness $k_{c,i}$ at the client. At first, a 1D mass-spring-damper deformable object is considered and it is subsequently shown how the results can be transferred to the interaction with 3D FEM-simulated deformable objects.

4.3.3.1 Stiffness adaptation for 1-DOF interactions

To compensate for the effect of communication delay on the forces rendered at the client, the magnitude of the impedance at the client is matched to the reference magnitude at the server by using a reduced stiffness in the force rendering at the clients.

The reduced stiffness $k^r_{c,i}$ is derived by approximating $K_i(s)$ as K_i as defined in Equation (4.44) and equating $|Z_{c,i}(s)|$ (with $k_{c,i}$ set zo $k^r_{c,i}$) and $|Z_{s,i}(s)|$. This leads to

$$k^r_{c,i}(\omega) = k_{s,i}\sqrt{\frac{k_o^2}{(k_o+k_{s,i})^2 - 2k_{s,i}(k_o+k_{s,i})\cos(\omega R_i) + k_{s,i}^2} - \frac{(b_{c,i}\omega)^2}{k_{s,i}^2}} \tag{4.46}$$

with $R_i < \pi/\omega$. If $k_o \gg b_{c,i}\omega$, the second fraction under the root in Equation (4.46) can be neglected. This is typically true for low-frequency inputs and limited damping applied at the clients. This implies that the characteristics of the touched object rather than the damping $b_{c,i}$, which is applied to ensure stability, dominates the impedance rendered at the client.

The impedance magnitudes plotted in Figure 4.20 support this assumption. For the used simulation parameters ($k_o = 100\,\mathrm{N/m}$, $b_{c,i} = 5\,\mathrm{Ns/m}$), the impedance magnitude at the client increases only slightly for $R_1 = 0\,\mathrm{ms}$ when compared to the server reference. This increase is caused solely by the damping at the client. The effect of communication delay on the impedance magnitude at the client for $R_1 > 0\,\mathrm{ms}$ is much larger.

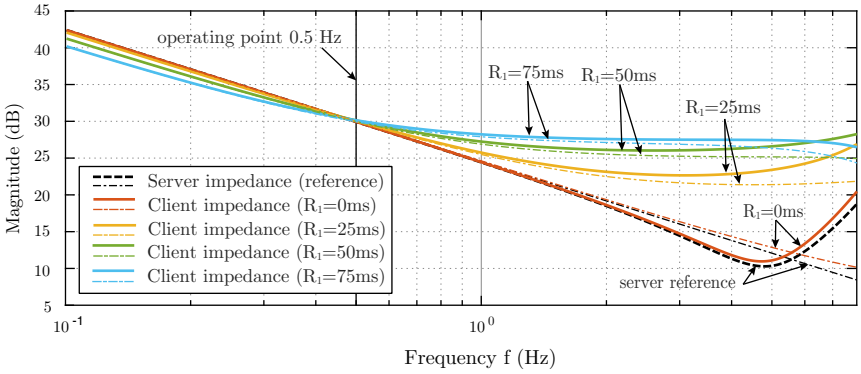


Figure 4.22: Magnitude of the client/server impedance in the presence of communication delay (solid lines) with the reduced stiffness $k^r_{c,1}$ applied at the clients. The impedance magnitude closely matches the server impedance magnitude (dashed line) for the considered input frequency for all simulated delay values. The dashed-dotted lines show the low-frequency approximations with $K_i(s)$ approximated with K_i as defined in Equation (4.44). The simulation parameters are as given in Figure 4.20, except $k_{c,1}$, which was set to the value calculated with Equation (4.47) (adopted from [1], © 2016 IEEE).

With the original stiffness setting, that is, $k_{c,i}$ is set to $k_{s,i} = k^o_{c,i}$, the reduced stiffness is calculated for $k_o \gg b_{c,i}\omega$ as:

$$k^r_{c,i}(\omega) = k^o_{c,i} \frac{k_o}{\sqrt{(k_o + k^o_{c,i})^2 - 2k_{c,i}(k_o + k^o_{c,i})\cos(\omega R_i) + (k^o_{c,i})^2}} \tag{4.47}$$

Equation (4.46) and Equation (4.47) allow us to calculate a reduced stiffness to be applied in the constraint-based force rendering at the clients for a user input with frequency $\omega = 2\pi f$.

The fraction in Equation (4.47) becomes one for zero communication delay, which results in the original stiffness value $k^o_{c,i}$ to be used at the client. It decreases monotonically for $R_i < \pi/\omega$.

Figure 4.22 shows the client-side impedance magnitudes for different communication delays if the reduced stiffness calculated with Equation (4.47) is applied. They closely match to the reference, the server-side impedance, at the considered input frequency of 0.5 Hz.

If no compensation is applied and the original stiffness $k^o_{c,i}$ is used at the client, the impedance magnitude error is 39.6 % (2.9 dB) for the used simulation parameters and $R_1 = 75\,\text{ms}$ (Figure 4.20). With $k^r_{c,i}$ from Equation (4.47), the error reduces to 3.2 % (0.27 dB). The matching error stems from the approximation of $K_i(s)$ as K_i and the neglected damping term in Equation (4.47). If Equation (4.46) is used to calculate $k^r_{c,i}$, the error is 3.0 % (0.26 dB) and differs only slightly from the approximative solution calculated with Equation (4.47).

The reduced stiffness introduces a mismatch between the reference impedance and the displayed impedance during the steady state (compare Figure 4.20 and Figure 4.22 for frequencies below 0.2 Hz). Furthermore, calculating the reduced stiffness with Equation (4.46) or Equation (4.47) is difficult in practice, because the human input frequency ω is not instantaneously known. If there is a mismatch between the true human input frequency and the estimated input frequency used to calculate the reduced stiffness, the displayed impedance will not match the desired impedance.

In the following, a formula to calculate the stiffness $k^r_{c,i}$ with the time-domain signals available at the clients is derived. This simplifies the practical implementation of the compensation scheme.

The server force from Equation (4.40) is calculated in the time-domain with the approximation of $K_i(s) \approx K_i$ from Equation (4.44) as:

$$f_{s,i}(t) = k_{s,i}\left[1 - K_i\right] x_{s,i}(t) \tag{4.48}$$

The client force from Equation (4.39) is given in the time-domain as:

$$f_{c,i}(t) = k_{c,i}\left[x_{d,i}(t) - K_i x_{d,i}(t - R_i)\right] + b_{c,i}\left[\dot{x}_{d,i}(t) - K_i \dot{x}_{d,i}(t - R_i)\right] \tag{4.49}$$

This can be rewritten as:

$$\begin{aligned} f_{c,i}(t) = \quad & k_{c,i}\left[1 - K_i\right] x_{d,i}(t) + k_{c,i} K_i \int_{t-R_i}^{t} \dot{x}_{d,i}(\tau) d\tau \\ & + b_{c,i}\left[1 - K_i\right] \dot{x}_{d,i}(t) + b_{c,i} K_i \int_{t-R_i}^{t} \ddot{x}_{d,i}(\tau) d\tau \end{aligned} \tag{4.50}$$

If $k_{c,i}$ is set to the original stiffness setting $k^o_{c,i} = k_{s,i}$, the first summand in Equation (4.50), which is denoted as $f^{ref}_{c,i}(t)$, corresponds to the server-side reference force $f_{s,i}(t + T_{f,i})$ since $x_{s,i}(t) = x_{d,i}(t - T_{f,i})$. The second summand describes the force error caused by the spring-based rendering and the communication delay, which is denoted as $\Delta f^k_{c,i}(t)$. The third and fourth component (summarized as $\Delta f^b_{c,i}(t)$) stem from the stability-ensuring damping at the client and the communication delay. Hence, the client force is given for the original stiffness setting as:

$$f_{c,i}(t) = f^{ref}_{c,i}(t) + \Delta f^k_{c,i}(t) + \Delta f^b_{c,i}(t) \tag{4.51}$$

In the following, the case wherein the impedance rendered at the client is dominated by the characteristic object stiffness rather than the damping ($k_o \gg b_{c,i}\omega$) is considered. Hence, the damping term $\Delta f^b_{c,i}(t)$ can be neglected in the derivations, because it has only a limited effect on the sought stiffness (see Equation (4.46)).

The implementation at the client is inherently time-discrete. Hence, the following formulas use the time-discrete notation of the signals, wherein t_n denotes the current time step. To derive the reduced stiffness, the client force $f_{c,i}[t_n]$ from Equation (4.50) with $k_{c,i}$ set to $k^r_{c,i} = k^o_{c,i} - k^k_{c,i}$ is equated with the desired reference force $f^{ref}_{c,i}[t_n]$. This leads to

$$k^r_{c,i}[t_n] = k^o_{c,i} - \underbrace{k^o_{c,i} \frac{\Delta f^k_{c,i}[t_n]}{f^{ref}_{c,i}[t_n] + \Delta f^k_{c,i}[t_n]}}_{k^k_{c,i}} = k^o_{c,i} - k^o_{c,i} \frac{\Delta f^k_{c,i}[t_n]}{f_{c,i}[t_n]} \tag{4.52}$$

where the subtrahend $k^k_{c,i}$ is the amount by which the stiffness is reduced.

The stiffness value $k^r_{c,i}[t_n]$ can be calculated at the client at runtime. The original stiffness setting $k^o_{c,i}$ is known. The force $f_{c,i}[t_n]$ corresponds to the currently rendered force at the client in the presence of communication delay if the original stiffness value $k^o_{c,i}$ is used. The force error $\Delta f^k_{c,i}[t_n]$ can be calculated from Equation (4.50), by replacing the integral, as:

$$\Delta f^k_{c,i}[t_n] = k^o_{c,i} K_i \left(x_{d,i}[t_n] - x_{d,i}[t_{n-R_i}]\right) \tag{4.53}$$

The device position $x_{d,i}[t_n]$ is read from the haptic device and can be stored for the last R_i milliseconds. The RTT R_i is assumed to be known, because it can be measured, for example, using time stamps.

The force $f_{c,i}[t_n]$ and the force error $\Delta f^k_{c,i}[t_n]$ have the same sign while pushing onto the object, that is, the fraction in Equation (4.52) is positive. It is smaller than one and the stiffness $k^r_{c,i}$ reduces compared to $k^o_{c,i}$ for $|\Delta f^k_{c,i}[t_n]| < |f_{c,i}[t_n]|$.

In the steady state, when the device velocity is zero, $\Delta f^k_{c,i}[t_n]$ becomes zero and the original stiffness is used ($k^r_{c,i} = k^o_{c,i}$).

While removing the haptic device from the object, $f_{c,i}[t_n]$ and $\Delta f^k_{c,i}[t_n]$ have different signs. The fraction in Equation (4.52) becomes negative and the stiffness theoretically increases compared to $k^o_{c,i}$. The value of $k^r_{c,i}$ is limited to a maximum of $k^{max}_{c,i} = k^o_{c,i}$ due to the fact that the proposed compensation scheme focuses only on the loading phase as discussed in Section 4.3.2.3. Hence, the compensation scheme does not affect the unloading of the deformable object.

At the initial contact with the object at the client, before the object deformation has arrived (i.e., the object is not yet deformed) or $|f_{c,i}[t_n]| < |\Delta f^k_{c,i}[t_n]|$, $k^r_{c,i}$ is set to a value of $k^o_{c,i}(1 - K_i) = k^o_{c,i}\frac{k_o}{k_o+k_{s,i}}$. This corresponds to the approximated stiffness the user perceives for zero-delay (see, for example, $f^{ref}_{c,i}$ in Equation (4.51)). It also constitutes the lower limit $k^{min}_{c,i}$ for $k^r_{c,i}$.

In other words, the result of Equation (4.52) is limited to the interval $[k^{min}_{c,i}, k^{max}_{c,i}]$. This corresponds to the desired behavior of the stiffness value to be used at the client to compensate for the force magnitude artifacts, which is observable in Figure 4.18 and Figure 4.21.

An exponential moving average filter on the calculated stiffness values is applied to smoothly update the stiffness for two implementation reasons. Firstly, the device position/velocity signal is noisy. This leads to noisy stiffness values calculated with Equation (4.52) and, hence, a noisy force feedback displayed to the user. Secondly, the device velocity changes over time and the stiffness is modulated accordingly while the user is in contact with the object. If updated abruptly, there are jumps and possibly instabilities in the rendered force feedback.

Increasing the stiffness while the user is in contact artificially creates energy, that is, the local rendering at the client is not passive if the applied damping $b_{c,i}$ does not dissipate enough energy. It is possible to ensure passivity in the local force rendering during the stiffness modulation by adopting proper control schemes (e.g., as proposed in [7]). Note that an active force rendering does not necessarily imply instability. This is due to the fact that the device damping and the human impedance also contribute to system stability [94] and passivity is a conservative stability criterion [82]. In fact, no instabilities due to such stiffness updates were experienced in the experimental setup used throughout this thesis.

4.3.3.2 Extension from 1-DOF to 3-DOF

So far, only a one-dimensional interaction has been considered, where the user movement, the object deformation, and the considered forces are all aligned. This is not true for the interaction with a 3D FEM-simulated deformable object.

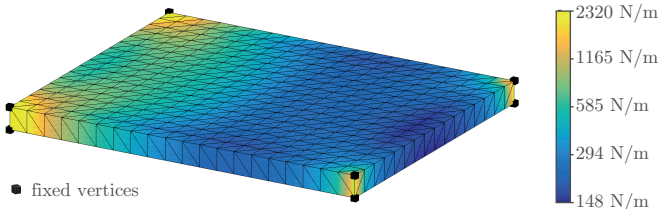


Figure 4.23: Estimated stiffness of the deformable object with inhomogeneous material properties (mass-density 10 000 kg/m^3, Young's modulus 10 kN/m^2 (left half) and 2.5 kN/m^2 (right half), Poisson's ratio 0.45) (adopted from [1], © 2016 IEEE).

The device position changes along the direction of the device velocity until the object deformation arrives at the client. This possibly leads to a force error, denoted by the vector

$$\Delta \boldsymbol{f_{c,i}^k}[t_n] = k_{c,i}^o K_i(\boldsymbol{x_{d,i}}[t_n] - \boldsymbol{x_{d,i}}[t_{n-R_i}]) \tag{4.54}$$

The force at the client does not increase due to the communication delay if the device movement is parallel to the surface. The force vector $\Delta \boldsymbol{f_{c,i}^k}$ is projected on the normal $\boldsymbol{n}$ of the currently touched surface triangle using the dot product $(\cdot)$ to account for this:

$$\Delta f_{c,i}^k[t_n] = |\Delta \boldsymbol{f_{c,i}^k}[t_n] \cdot \boldsymbol{n}| \tag{4.55}$$

The following equation considers again the case wherein the impedance rendered at the client is dominated by the object stiffness ($k_o \gg b_{c,i}\omega$). Rewriting Equation (4.52) leads to the reduced stiffness for the 3D interaction:

$$k_{c,i}^r[t_n] = k_{c,i}^o - k_{c,i}^o \frac{\Delta f_{c,i}^k[t_n]}{\|\boldsymbol{f_{c,i}}[t_n]\|} \tag{4.56}$$

Again, the result is limited to the interval $[k_{c,i}^{min}, k_{c,i}^{max}]$. If the user moves tangentially to the surface, the dot product in the calculation of $\Delta f_{c,i}^k[t_n]$ becomes zero and the original stiffness $k_{c,i}^o$ is used.

If surface friction is rendered, the client force $\boldsymbol{f_{c,i}}$ is not parallel to $\boldsymbol{n}$. In this case, the component of $\boldsymbol{f_{c,i}}$ that is normal to the surface should be considered in Equation (4.56), because the friction component is rendered directly at the client and not influenced by the delay.

The proposed scheme approximates the complex FEM simulation with a linear spring with stiffness k_o, which is incorporated in K_i. This stiffness is known in the 1D example, because it is an explicit parameter of the mass-spring-damper system. However, it is not known for a general 3D deformable object. The material model properties, the object geometry, and vertices with a fixed position in the simulation influence the local stiffness of the object.

To account for this, the local stiffness of a general deformable object is estimated at each surface vertex position by simulating a constant force input in the direction of the vertex normal and dividing the force magnitude with the resulting vertex displacement. This precomputation provides a stiffness map, which the client uses to approximate the stiffness at the current contact point using barycentric interpolation. Figure 4.23 shows the stiffness map of the FEM-simulated membrane-liked object used throughout this thesis.

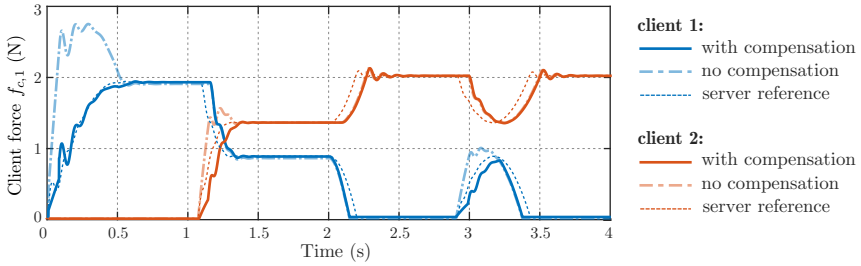


Figure 4.24: Two-user interaction with a 1-DOF mass-spring-damper deformable object. The simulated input and object are identical to Figure 4.18. The RTT is 80ms for client 1 and 60 ms for client 2 (adopted from [1], © 2016 IEEE).

4.3.4 Evaluation

This section presents the evaluation of the proposed scheme to compensate for the effect of communication delay during the remote interaction with deformable objects in the CS-State architecture.

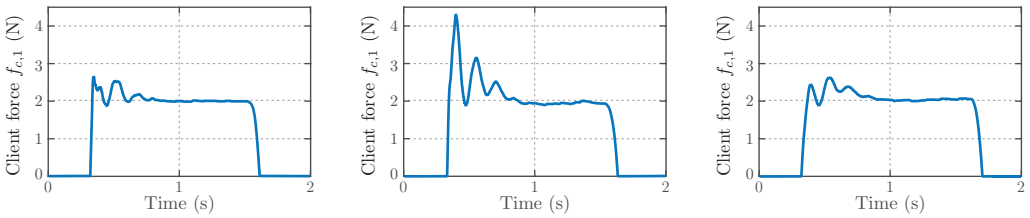
4.3.4.1 Objective evaluation with a 1D object

Figure 4.24 shows the force signals recorded at the clients for the same simulated two-user interaction with the 1D object as in Figure 4.18. The simulation parameters were $T_s = 0.001\text{s}$, $k_o = 100\,\text{N/m}$, $b_o = 2.0\,\text{Ns/m}$, $m_o = 0.1\,\text{kg}$, $k^o_{c,i} = k_{s,i} = 340\,\text{N/m}$, and $b_{c,1} = 5\,\text{Ns/m}$. The force profiles in the presence of communication delay are very similar to the server reference if the proposed compensation scheme is enabled. It can be observed that the peak in the force magnitude while pressing on the object is successfully removed for the active client. There is no effect of the proposed scheme on the forces rendered for the passive client.

The force signals commanded to the haptic device during a human user interaction with the 1D object are shown in Figure 4.25. The same parameters as in the previous simulations were used. Only the damping $b_{c,i}$ was set to zero, because no instabilities were experienced at the client with the applied parameters. The zero-delay client force served as the reference, because it equals the server force in this case (see Equation (4.48) and Equation (4.50)). A sample-by-sample comparison of the force signals is not feasible, because it is not possible to repeat exactly the same interaction several times with the human-in-the-loop. The qualitative comparison of the force signals in Figure 4.25 shows that the force profiles are very similar and the peak while pressing on the object is successfully removed if the proposed scheme to compensate the effect of communication delay is enabled.

4.3.4.2 Subjective evaluation

A subjective test has been performed to asses if the proposed scheme successfully hides the communication delay from the users. The goal was to test if users are able to distinguish the haptic force feedback between the non-delayed interaction and the interaction in the presence of constant communication delay with a deformable object in the studied CS-State architecture.



(a) Zero-delay reference (b) Without compensation (c) With compensation

Figure 4.25: Force signals for a real user interaction with the 1D deformable object for (a) zero-delay, (b) in the presence of 50 ms communication delay without compensation and (c) in the presence of 50 ms communication delay with the proposed compensation scheme being activated (adapted from [1], © 2016 IEEE).

Method The experiment was based on the following methodology.

Apparatus The stiffness of a FEM-simulated object varies locally (see Section 4.3.3.2) and, hence, the perceived stiffness during the interaction with the object depends on the current contact point. This impairs the comparison of the stimuli under investigation and might influence the subjects' answers. For this reason, the experiment was based on a 1D mass-spring-damper object. A single user was considered in the experiment, because the effect of communication delay is apparent primarily for the active user (see Section 4.3.2.3).

The experiment was based on the CS-State architecture and the simulation parameters given in Section 4.3.4.1. Since no instabilities were experienced with this setup if the user holds the stylus in her/his hand, the damping $b_{c,i}$ was set to zero.

Procedure and task The same-different procedure [61] was adopted to asses if subjects can discriminate between the interaction in the non-delayed reference, the interaction in the presence of communication delay without delay compensation, and the interaction in the presence of communication delay with the proposed delay compensation enabled.

Communication delay in the CS-State architecture leads to increased force magnitudes and an increased impedance displayed to the user. In each trial, the subject had to poke two simulated objects and state if their stiffness (i.e., impedance) is perceived as same or different. Only the delay simulation and the compensation scheme was turned on/off between the two objects (see below). The subjects were asked to poke with a frequency of about 1 Hz and to keep consistent between trials. This task was selected, because communication delay affects the rendered force at the client only during active interactions with the object (see Section 4.3.2.3).

The two objects in each trial were presented sequentially. Once the first object was poked twice, the subjects proceeded to the next object by pressing a key on the keyboard. After both objects had been poked twice, the subject entered the answer on the keyboard and continued to the next trial. This setup allowed the subjects to hold the haptic device in one hand, while using the keyboard with the other hand. The two objects within one pair could be compared within 5 s to account for the short human haptic memory. A training phase was conducted with each subject to familiarize them with the task and the procedure.

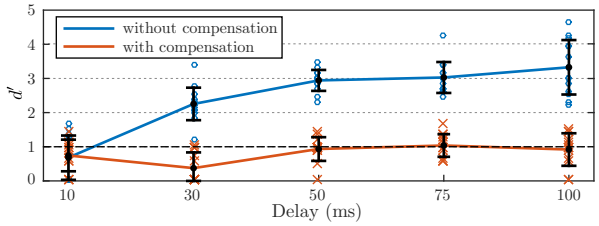


Figure 4.26: Scatter plot of the calculated d' values. The error bars denote the mean and standard deviation of the calculated d' values for all subjects. The solid line connects the means of the d' values (adopted from [1], © 2016 IEEE).

Stimuli The focus is on communication delays that are haptically, but not visually perceivable by the user. The results reported in [229] suggest that delayed deformation updates becomes visually perceivable at about 100 ms. For this reason, 100 ms denoted the maximum delay in the experiment. The tolerable communication delay for the used simulation parameters, calculated with Equation (4.45) for $\omega = 1\,\text{Hz}$ and $c = 10\,\%$, is 19 ms. The tested delay values were $R_1 = \{10, 30, 50, 75, 100\}$ ms. Nevertheless, the subject could not see the virtual object during the experiment to avoid visual cues influencing the subject's answers.

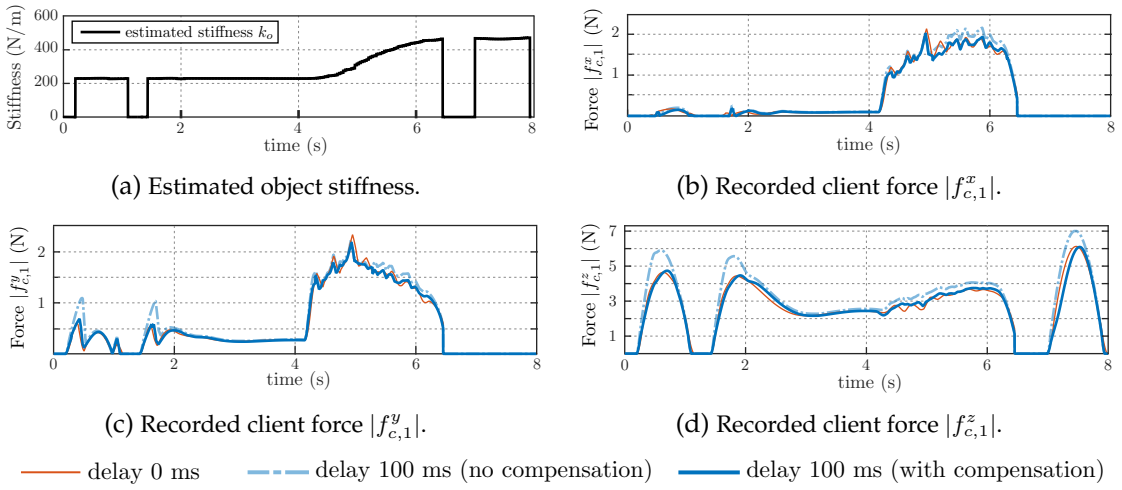
The stimulus S_0^{NC} (no compensation, 0 ms communication delay) denoted the reference against which the other stimuli $S_{R_1}^{NC}$ (no compensation, communication delay R_1) and $S_{R_1}^{WC}$ (with compensation, communication delay R_1) were compared. The 10 stimuli pairs ($S_{R_1}^{NC/WC}$–S_0^{NC}) plus the reference pair S_0^{NC}–S_0^{NC} had been presented 25 times each, which resulted in 275 trials per subject. The order of the trials and the two stimuli within a trial was randomized. A break was enforced after every 55 trials to avoid fatigue of the subjects.

Note that the object parameters (k_o, b_o, m_o) were not changed during the experiment. Only the delay was changed and the compensation scheme was enabled or disabled, which eventually resulted in a perceived difference between the two objects.

Participants 14 voluntary subjects participated in the experiment (12 male, 2 female, aged between 22 and 33). Three were frequent users of haptic devices, six had already participated in former experiments, and the others were novice users.

Results The subjects' answers were analyzed using Signal Detection Theory (SDT) [235]. Hit and false alarm rates and the corresponding d-prime values (d'), which are a measure of the subjects' sensitivity to differences between the stimuli, were calculated for each subject individually. The reference pair S_{NC}^0-S_{NC}^0 served as the baseline to calculate the false alarm rates and the d' values. The differencing strategy [235] was assumed as the human's decision rule as suggested in [61], because the experiment included several comparisons. The d' values, determined with the table in [235], are plotted in Figure 4.26. Higher values of d' indicate higher discriminability. A value of $d' = 1$ is often assumed as the threshold, for which the difference between the stimuli becomes perceivable [61].

A Lilliefors test suggested that the collected d' data is not normally distributed. Hence, non-parametric tests [220] were used to evaluate statistical significance in the results (significance level 5%).



(a) Estimated object stiffness.

(b) Recorded client force $|f_{c,1}^x|$.

(c) Recorded client force $|f_{c,1}^y|$.

(d) Recorded client force $|f_{c,1}^z|$.

Figure 4.27: The estimated object stiffness and the force signals recorded at the client for the single-user interaction with the 3D membrane-like FEM-simulated deformable object shown in Figure 4.2a (adopted from [1], © 2016 IEEE).

The d' values increase with the communication delay if no compensation is applied at the client. The average d' value between subjects is 0.68 for a delay of 10 ms, that is, the effect of 10 ms communication delay is assumed to be below the human perception threshold. The tolerable communication delay for the used setup is 19 ms. For 30 ms, d' increases to an average value of 2.25, which is above the threshold of $d' = 1$. A Kruskal-Wallis test indicates a statistically significant effect of the communication delay on the d' values ($\chi^2(4) = 46.13, p < 0.001$).

If the proposed compensation scheme was enabled, the d' values are generally smaller compared to the stimuli where no compensation was applied. They are below or close to the threshold of $d' = 1$. Only the average d' value for $R_1 = 75\,\text{ms}$ (1.04) is slightly above. Friedman's Test indicates a significant effect of the compensation method on the d' values ($\chi^2(1) = 66.47, p < 0.001$). The Kruskal-Wallis test found a significant effect of the delay on the d' values ($\chi^2(4) = 13.04, p = 0.011$) even if the proposed method was applied. This indicates that the compensation was not equally effective over the tested delay range. The post-hoc analysis using Bonferroni correction revealed, however, that only the d' values for $R_1 = 30\,\text{ms}$ and $R_1 = 75\,\text{ms}$ are statistically significantly different from each other. This can be attributed to the small sample size of the experiment.

In summary, the results of the subjective experiment confirm that the proposed scheme effectively hides the communication delay from the users for the tested delay range of up to 100 ms, since the d' values are below or close to the threshold of $d' = 1$ if the proposed scheme is enabled.

4.3.4.3 Objective evaluation with a FEM-simulated 3D object

The following signals were recorded during the interaction with the membrane-like FEM-simulated deformable object (see Figure 4.23).

Figure 4.27a shows the estimated object stiffness k_o during the simulated interaction il-

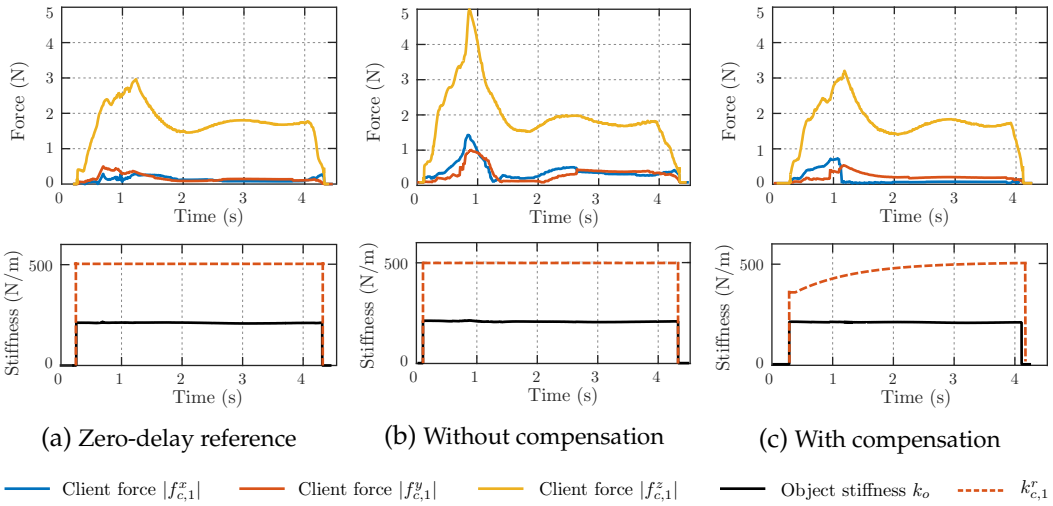


Figure 4.28: Force signals and stiffness values for a real user interaction with the FEM-simulated object for (a) zero-delay, (b) in the presence of 100 ms communication delay without compensation and (c) in the presence of 100 ms communication delay with the proposed compensation scheme being activated (adapted from [1], © 2016 IEEE).

lustrated in Figure 4.2a, by using the precomputed stiffness map shown in Figure 4.23. The client uses k_o to calculate the reduced local model stiffness $k^r_{c,1}$ as explained in Section 4.3.3.2. The object has inhomogeneous material properties. While moving tangential to the surface between second four and six (see Figure 4.2a) the estimated object stiffness increases.

The force signals in the x-/y-/z-direction in Figures 4.27b – 4.27d illustrate the effect of the proposed compensation scheme. The force profiles in all three directions are well matched to the zero-delay reference if the proposed compensation scheme is enabled. This validates the extension from 1D to 3D discussed in Section 4.3.3.2.

Figure 4.27 also shows that the effect of delay decreases if the object stiffness increases. This can be observed by comparing the force in the z-direction in Figure 4.27d for the first and the third interaction. The deformation is smaller for a stiffer object if the same force is applied on the object. A smaller deformation implies a smaller deformation error due to the delay at the client and, hence, a smaller effect on the rendered force.

Finally, Figure 4.28 shows the force signals commanded to the haptic device, the estimated object stiffness k_o, and the stiffness $k^r_{c,1}$, which used in the local force rendering at the client, during a human user interaction. The stiffness $k_{c,1}$ was originally set to 500 N/m. There is a clear mismatch between the force profile for the zero-delay reference in Figure 4.28a and the force profile without compensation at the client for a RTT of 100 ms in Figure 4.28b. In Figure 4.28c, the force profile is very similar to the zero-delay reference due to the proposed stiffness reduction.

4.3.4.4 Discussion

The proposed compensation scheme approximates the deformable object locally with a linear spring (k_o) and uses the presented model of the CS-State architecture to predict the effect of the delayed deformation updates on the rendered forces at the clients. The adaptive force

rendering algorithm exploits this knowledge to adjust the stiffness accordingly. The evaluation clearly shows the usefulness of the proposed scheme to compensate for the effect of communication delay, as the peaks in the force amplitudes are successfully removed.

The linear approximation of the deformable object with a spring has limitations. Firstly, it neglects dynamic properties as, for example, the inertia of the deformable object. The proposed scheme can be extended by employing a higher-order model to approximate the object response. This requires a more complex model estimation during the precomputation and is a possible extension of the presented work.

Secondly, the linear approximation is not directly applicable for non-linear FEM models. The local stiffness estimation can be improved to capture non-linearities during the contact. If, for example, a non-linear stiffness representation can be linearized locally for the current interaction, the presented approach can be adopted to calculate a stiffness to be used in the force rendering at the clients.

Furthermore, the precomputed local object stiffness becomes partially invalid if the deformable object topology changes at runtime, for example, in a cutting application. To cope with this, the server would need to update the stiffness map at runtime and transmit the results to the clients.

4.4 Chapter summary

Communication delay in the CS-State architecture leads to delayed object state updates and, as a result, changes in the magnitudes of the rendered forces at the clients. In particular, communication delay leads to

- increased force magnitudes while moving a rigid object and
- increased force magnitudes while actively pressing on a deformable object.

The analyses of the effect of communication delay on the rendered forces at the clients in this chapter revealed guidelines on how much delay is tolerable before its effect becomes haptically perceivable. These guidelines incorporate the simulation parameters, the user interactions, and the limitations of human perception.

This chapter has also presented schemes to compensate for these effects by adaptively adjusting the spring coefficient used in the local force rendering algorithms at the clients. These schemes can be seen as methods to achieve haptic transparency if the tolerable communication delay is exceeded.

Compensating the effect of communication delay in principle always requires prediction. The proposed schemes predict the effect of communication delay on the force magnitudes at the clients using the presented mathematical models of the CS-State architecture. This knowledge is incorporated into algorithms to calculate a reduced stiffness to be used in the constraint-based force rendering at the clients. The objective and subjective evaluation has shown that this approach successfully hides the communication delay from the users.

It is important to note that the proposed schemes deal only with the effect of communication delay on the haptic feedback. Due to the sensibility of the haptic perception system, users perceive the effects of communication delay haptically, but not visually, for lower delays. For higher delay values, however, the delayed object state updates become perceivable haptically and visually. Approximative local physics simulations could be applied at the clients as an alternative way to compensate for the effect of delay as discussed in the outlook section of the following chapter.

Chapter 5

Conclusion and outlook

Shared Haptic Virtual Environments enable distant users, possibly distributed around the globe, to remotely collaborate in a virtual reality application and to physically interact and feel the objects in the environment. Promising applications are found in the areas of education, teaching, training, industry, and entertainment. Different architectures, which vary in the level of distribution of different parts of the application, can be conceived for the implementation of SHVEs.

This dissertation focuses on a client/server architecture, wherein the physics simulation runs solely on the centralized server. Copies of the VE are used locally at the clients to render the scene visually and haptically without delay in the control loop between the haptic device and the VE. This ensures consistency and makes the SHVE robust against instabilities caused by communication delay. This architecture suffers, however, from two main challenges as discussed in Section 2.4.

Firstly, the object state updates, which are calculated by the centralized physics simulation on the server, have to be transmitted to the clients to update the local copies of the VE. For movable rigid objects, this transmission is characterized by high packet rates and low payload. For deformable objects, on the other hand, lower packet rates and high payload characterize the communication. In both cases, stringent delay constraints complicate the efficient transmission of object state updates as communication and coding delay lead to a loss of fidelity. This constitutes the second challenge in the considered architecture and prohibits the use of block-based coding schemes to reduce the packet and bit rates required for the transmission of object state updates.

5.1 Summary of the results

Chapter 3 introduced data reduction schemes that are designed to add only negligible delay in the communication from the server to the clients. The data reduction scheme for the interaction with movable rigid objects (Section 3.2) reduces the packet rate by predicting object states (positions and rotations) using the last received updates. Prediction error thresholds are used to determine if a new update packet should be transmitted. It uses separate thresholds for the visual and haptic modality to achieve a good trade-off between packet rate reduction and fidelity. The haptic threshold is continuously adapted to the currently rendered

force to account for limitations of human kinesthetic force perception. The proposed scheme achieves a packet rate reduction from 1000 to 28.6 update packets per second in the tested scenarios without adding a significant visual-haptic distortion. In comparison to the state-of-the-art, it requires fewer update packets to achieve a better haptic feedback quality.

The data reduction scheme for the interaction with deformable objects compresses the payload of individual update packets by applying prediction of vertex displacements, prediction error thresholds, quantization and entropy coding. Vertices close to the current interaction are given higher priority in the encoding process as they are deemed to be important for the haptic feedback quality. This allows us to introduce coding noise in areas of the polygon mesh where the distortion is barely perceivable or even below visual or haptic detection thresholds. In the subjective evaluation, the proposed scheme reduced the bit rate from initially 4.1 Mbit/s to 372 kbit/s (compression ratio 11:1) in the considered task without adding a significant visual-haptic distortion.

The two data reduction schemes consider the visual and haptic modalities separately to keep the introduced distortion below or close to the respective detection thresholds. The evaluation has shown in both cases that this approach outperforms state-of-the-art data reduction schemes. The reason is the different sensitivity of the human visual and haptic perception system to changes in the virtual environment. The haptic modality is more sensitive and, thus, requires more updates.

The increased sensitivity of the haptic perception system also causes the effect that communication delay in the considered client/server architecture becomes perceivable haptically, but not visually for small communication delay values. As the object position, rotation, or deformation is simulated solely on the server, the local copies of the VE at the clients are updated only after a RTT. This leads to increased force magnitudes rendered at the clients and a loss of fidelity. The transparency analyses in Chapter 4 reveal guidelines on the tolerable communication delay, before the increased force magnitudes become haptically perceivable. These analyses are based on mathematical models of two-user interactions with rigid or deformable objects.

Chapter 4 also proposed schemes to achieve haptic transparency in the presence of communication delay. Using the aforementioned models, the effect of delay on the force magnitude is predicted and the local force rendering algorithms are adapted accordingly to compensate for it. The experimental results, including subjective user studies, have shown that the proposed compensation schemes successfully hide the effect of constant and time-varying communication delay from the users.

5.2 Limitations and outlook

With the proposed compensation schemes, the object still does not move or deform immediately at the initial contact at the client if communication delay exists. This is due to the fact that the physics simulation runs on the server and the object state/deformation has to be transmitted to the client. For large delay values, the delay becomes noticeable not only haptically, but also visually. The proposed schemes only compensate for the effects on the

rendered forces. This constitutes the main limitation of the presented compensation schemes. Although the evaluation has shown that delay of up to 100 ms can be compensated successfully, the delay becomes visible to the user, especially for values above 100 ms. This reduces the fidelity, for example, for European users connecting to a simulation server in Asia.

As a consequence, local physics simulations become necessary. This constitutes a fundamental change in the considered architecture, but is an interesting direction for future research. These local simulations at the clients could be as accurate as at the server to handle the interaction with rigid objects, because the computational complexity of rigid body physics simulation is low (see Section 2.3.2). A centralized simulation at the server could calculate a globally valid state, which is then used to correct the local simulations. Consistency problems and simulation errors will arise especially for collaborative interactions as studied, for example, in Section 4.2.4. The challenge in this context is the correction of simulation errors. If these errors, which will grow with the amount of communication delay, are not corrected conveniently, the fidelity of the SHVE reduces and collaboration might become unfeasible.

Similarly, local deformation simulations could be applied at the clients. Due to the complexity of physics-based deformation simulations (see Section 2.3.2), these could be only physically plausible, but not physically correct. The approximative simulations could use, for example, radial functions, which were proposed in [239] to approximate deformations of point-cloud data.

The aforementioned limitations stem from the communication delay and the highlighted directions for future work consider only the application, which has to adapt to given network conditions. At the same time, research on next-generation communication networks focuses on providing low-delay, high-reliability transmission of haptic data to enable various novel applications in the areas of remote monitoring, remote surgery, eduction and training, etc. [240], [241]. Clearly, this development towards the so-called *Tactile Internet* will support the implementation of SHVEs. Data reduction schemes as investigated in this dissertation will be necessary to realize a scalable network architecture [241].

The Tactile Internet is envisioned to adopt a flexible and programmable network architecture, for example, using the technologies of software defined networking [242], network virtualization [243], and cloud-computing [244]. This allows one to jointly consider application requirements and network resources to optimize the resource allocation and user experience. Clearly, providing the lowest possible delay for all users would be desirably, but is also not practical in reality due to resource constraints. The analyses in Chapter 4 have shown that some communication delay is tolerable and, if this delay is exceeded, the effects of delay can be compensated to some extent by the application. Also, the packet and bit rates can be adjusted as shown in Chapter 3 to introduce distortions that are not perceivable or, for example, perceivable but not disturbing. This flexibility constitutes the basis for a joint optimization of the shared network and the applications. The applications' requirements might vary over time, since the required delay, packet and bit rate typically depend on the current user interaction and the simulation parameters. An *Application-Network-Interface* that allows the application and the network to exchange information about the desired QoS and

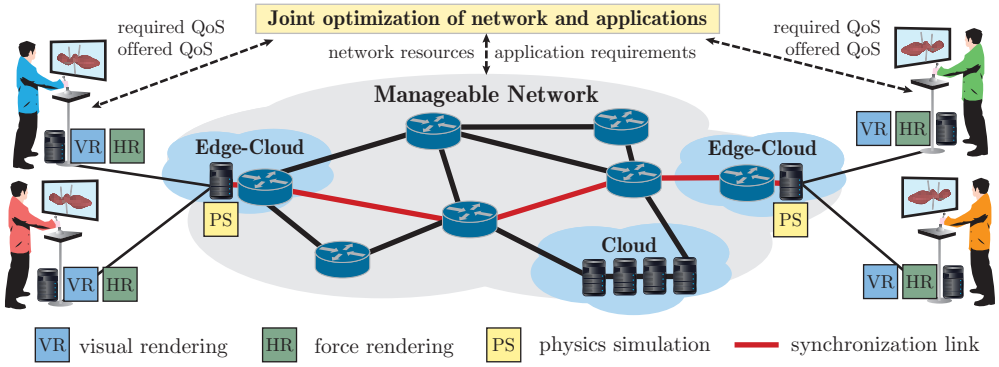


Figure 5.1: Vision of a flexible and manageable network infrastructure supporting the implementation of distributed SHVEs. The applications and the network exchange information about their current state and desired or offered QoS, respectively. This allows for the optimization of the resource allocation between users. Edge-cloud services offer computational power to drive physics simulations for the collaborative virtual applications.

the possible QoS would enable the joint orchestration of the application and the network.

But still, the speed of light will constitute a lower bound on the achievable communication delay. For long-distance collaboration in a SHVE, edge-cloud simulation servers could bring the physics engine close to the users to allow for low-delay communication with a simulation server as illustrated in Figure 5.1. The core network then provides a high-reliability synchronization link between the edge-clouds. The challenge in this context is the synchronization of the edge-cloud servers and the correction of simulation errors.

Bibliography

Publications by the author

Journal publications

[1] C. Schuwerk, X. Xu, and E. Steinbach, "On the transparency of client/server-based haptic interaction with deformable objects," *IEEE Transactions on Haptics*, Sep. 2016. DOI: 10.1109/TOH.2016.2612635.

[2] C. Schuwerk, X. Xu, R. Chaudhari, and E. Steinbach, "Compensating the effect of communication delay in client-server-based shared haptic virtual environments," *ACM Transactions on Applied Perception*, vol. 13, no. 1, Dec. 2015. DOI: 10.1145/2835176.

[3] C. Schuwerk, G. Paggetti, R. Chaudhari, and E. Steinbach, "Perception-based traffic control for shared haptic virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 23, no. 3, Oct. 2014. DOI: 10.1162/PRES_a_00196.

[4] X. Xu, C. Schuwerk, B. Cizmeci, and E. Steinbach, "Energy prediction for teleoperation systems that combine the time domain passivity approach with perceptual deadband-based haptic data reduction," *IEEE Transactions on Haptics*, vol. 9, no. 4, Oct. 2016. DOI: 10.1109/TOH.2016.2558157.

[5] M. Strese, C. Schuwerk, A. Iepure, and E. Steinbach, "Multimodal feature-based surface material classification," *IEEE Transactions on Haptics*, Nov. 2016. DOI: 10.1109/TOH.2016.2625787.

[6] R. Chaudhari, C. Schuwerk, M. Danaei, and E. Steinbach, "Perceptual and bitrate-scalable coding of haptic surface texture signals," *IEEE Journal of Selected Topics in Signal Processing (JSTSP)*, vol. 9, no. 3, pp. 462–473, Apr. 2015. DOI: 10.1109/JSTSP.2014.2374574.

[7] X. Xu, B. Cizmeci, C. Schuwerk, and E. Steinbach, "Model-mediated teleoperation: Toward stable and transparent teleoperation systems," *IEEE Access*, vol. 4, Jan. 2016. DOI: 10.1109/ACCESS.2016.2517926.

Conference publications

[8] C. Schuwerk, W. Freund, and E. Steinbach, "Low-delay compression of polygon mesh deformation data for remote haptic interaction with simulated deformable objects," in *Proceedings of the IEEE Haptics Symposium (HAPTICS)*, Philadelphia, USA, Apr. 2016.

[9] C. Schuwerk, X. Xu, W. Freund, and E. Steinbach, "Transparency analysis of client-server-based multi-rate haptic interaction with deformable objects," in *Proceedings of the IEEE World Haptics Conference 2015*, Chicago, USA, Jun. 2015, pp. 506–511. DOI: 10.1109/WHC.2015.7177761.

[10] C. Schuwerk, R. Chaudhari, and E. Steinbach, "Delay compensation in shared haptic virtual environments," in *Proceedings of the IEEE Haptics Symposium (HAPTICS)*, Houston, USA, Feb. 2014, pp. 371–377. DOI: 10.1109/HAPTICS.2014.6775484.

[11] ——, "Perceptually robust traffic control in distributed haptic virtual environments," in *Haptics: Perception, Devices, Mobility, and Communication (Eurohaptics Conference)*, vol. 7282, Tampere, Finland: Springer Berlin Heidelberg, Jun. 2012, pp. 469–480. DOI: 10.1007/978-3-642-31401-8_42.

[12] C. Schuwerk, N. Chaudhari, and E. Steinbach, "An area-of-interest based communication architecture for shared haptic virtual environments," in *Proceedings of the IEEE International Symposium on Haptic Audio-Visual Environments and Games (HAVE)*, Istanbul, Turkey, Oct. 2013, pp. 57–62. DOI: 10.1109/HAVE.2013.6679611.

[13] C. Schuwerk and E. Steinbach, "Smooth object state updates in distributed haptic virtual environments," in *Proceedings of the IEEE International Symposium on Haptic Audio-Visual Environments and Games (HAVE)*, Istanbul, Turkey, Oct. 2013, pp. 51–56. DOI: 10.1109/HAVE.2013.6679610.

[14] X. Xu, B. Cizmeci, C. Schuwerk, and E. Steinbach, "Haptic data reduction for time-delayed teleoperation using the time domain passivity approach," in *Proceedings of the IEEE World Haptics Conference*, Chicago, USA, Jun. 2015, pp. 512–518. DOI: 10.1109/WHC.2015.7177763.

[15] X. Xu, C. Schuwerk, and E. Steinbach, "Passivity-based model update for model-mediated teleoperation," in *Proceedings of IEEE International Conference on Multimedia & Expo, 6th IEEE International Workshop on Hot Topics in 3D - Hot3D*, Turin, Italy, Jun. 2015, pp. 1–6. DOI: 10.1109/ICMEW.2015.7169831.

[16] R. Chaudhari, Y. Yoo, C. Schuwerk, S. Choi, and E. Steinbach, "Objective quality prediction for haptic texture signal compression," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 2224–2228. DOI: 10.1109/ICASSP.2015.7178366.

[17] R. Chaudhari, C. Schuwerk, V. Nitsch, E. Steinbach, and B. Färber, "Opening the haptic loop: Network degradation limits for haptic task performance," in *Proceedings of the IEEE International Symposium on Haptic Audio-Visual Environments and Games (HAVE)*, Nanchang, Jiangxi, China, Oct. 2011, pp. 56–61. DOI: 10.1109/HAVE.2011.6088392.

[18] M. Strese, C. Schuwerk, and E. Steinbach, "Surface classification using acceleration signals recorded during human freehand movement," in *Proceedings of the IEEE World Haptics Conference*, Jun. 2015, pp. 214–219. DOI: 10.1109/WHC.2015.7177716.

[19] ——, "On the retrieval of perceptually similar haptic surfaces," in *Proceedings of the International Workshop on Quality of Multimedia Experience (QoMEX)*, Costa Navarino, Greece, May 2015, pp. 1–6. DOI: 10.1109/QoMEX.2015.7148141.

[20] M. Strese, J.-Y. Lee, C. Schuwerk, Q. Han, H.-G. Kim, and E. Steinbach, "A haptic texture database for tool-mediated texture recognition and classification," in *Proceedings of the IEEE International Symposium on Haptic Audio-Visual Environments and Games (HAVE)*, Dallas, Texas, USA, Oct. 2014, pp. 118–123. DOI: 10.1109/HAVE.2014.6954342.

General publications

[21] Oculus VR, LLC. (Feb. 2016). Oculus rift: Next-generation virtual reality, [Online]. Available: http://www.oculus.com/.

[22] M. Srinivasan and C Basdogan, "Haptics in virtual environments: taxonomy, research status, and challenges," *Computers & Graphics*, vol. 21, no. 4, pp. 393 –404, Apr. 1997.

[23] J. T. Dennerlein, D. B. Martin, and C. Hasser, "Force-feedback improves performance for steering and combined steering-targeting tasks," in *Proceedings of the SIGCHI conference on human factors in computing systems*, Apr. 2000, pp. 423–429.

[24] C. Basdogan, C. H. Ho, M. A. Srinivasan, and M. Slater, "An experimental study on the role of touch in shared virtual environments," *ACM Transactions on Computer-Human Interaction*, vol. 7, no. 4, pp. 443–460, Dec. 2000.

[25] A. E. Saddik, "The potential of haptics technologies," *IEEE Instrumentation & Measurement Magazine*, vol. 10, no. 1, pp. 10–17, Apr. 2007.

[26] V. Hayward, O. R. Astley, M. Cruz-Hernandez, D. Grant, and G. Robles-De-La-Torre, "Haptic interfaces and devices," *Sensor Review*, vol. 24, no. 1, pp. 16–29, 2004.

[27] S. Benford, C. Greenhalgh, T. Rodden, and J. Pycock, "Collaborative virtual environments," *Communications of the ACM*, vol. 44, no. 7, pp. 79–85, Jul. 2001.

[28] E. F. Churchill and D. Snowdon, "Collaborative virtual environments: An introductory review of issues and systems," *Virtual Reality*, vol. 3, no. 1, pp. 3–15, 1998.

[29] P. Buttolo, R. Oboe, and B. Hannaford, "Architectures for shared haptic virtual environments," *Computers & Graphics*, vol. 21, no. 4, pp. 421–429, Jul. 1997.

[30] K. Montgomery, C. Bruyns, J. Brown, S. Sorkin, F. Mazzella, G. Thonier, A. Tellier, B. Lerman, and A. Menon, "Spring: A general framework for collaborative, real-time surgical simulation," *Medicine meets Virtual Reality*, pp. 296–303, 2002.

[31] C. Gunn, M. Hutchins, D. Stevenson, M. Adcock, and P. Youngblood, "Using collaborative haptics in remote surgical training," in *Proceedings of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2005, pp. 481–482.

[32] T. R. Coles, D. Meglan, and N. W. John, "The role of haptics in medical training simulators: a survey of the state of the art," *IEEE Transactions on Haptics*, vol. 4, no. 1, pp. 51–66, Jan. 2011.

[33] R. Iglesias, S. Casado, T. Gutiérrez, A. García-Alonso, W. Yu, and A. Marshall, "Simultaneous remote haptic collaboration for assembling tasks," *Multimedia Systems*, vol. 13, no. 4, pp. 263–274, Dec. 2007.

[34] J. Marsh, M. Glencross, and S. Pettifer, "Minimising latency and maintaining consistency in distributed virtual prototyping," in *Proceedings of the ACM Siggraph International Conference on Virtual Reality and its Applications in Industry*, 2004, pp. 386–389.

[35] J. Marsh, M. Glencross, S. Pettifer, and R. Hubbold, "A network architecture supporting consistent rich behavior in collaborative interactive applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 3, pp. 405–416, Mar. 2006.

[36] Y. Ishibashi and H. Kaneoka, "Fairness among game players in networked haptic environments: influence of network latency," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2005, pp. 1–4.

[37] W. Ferrell and T. Sheridan, "Supervisory control of remote manipulation," *IEEE Spectrum*, vol. 4, no. 10, pp. 81–88, Oct. 1967.

[38] R. T. Souayed, D. Gaiti, W. Yu, G. Dodds, and A. Marshall, "Experimental study of haptic interaction in distributed virtual environments," in *Proceedings of the Eurohaptics Conference*, 2004, pp. 260–266.

[39] M. Fotoohi, S. Sirouspour, and D. Capson, "Stability and performance analysis of centralized and distributed multi-rate control architectures for multi-user haptic interaction," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 977–994, Sep. 2007.

[40] P. Hinterseer, S. Hirche, S. Chaudhuri, E. Steinbach, and M. Buss, "Perception-based data reduction and transmission of haptic data in telepresence and teleaction systems," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 588–597, Feb. 2008.

[41] E. Steinbach, S. Hirche, J. Kammerl, I. Vittorias, and R. Chaudhari, "Haptic data compression and communication for telepresence and teleaction," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 87–96, Jan. 2011.

[42] E. Steinbach, S. Hirche, M. Ernst, F. Brandi, R. Chaudhari, J. Kammerl, and I. Vittorias, "Haptic communications," *Proceedings of the IEEE*, vol. 100, no. 4, pp. 937–956, Apr. 2012.

[43] S. Matsumotoy, I. Fukuday, and H. Morinoy, "The influences of network issues on haptic collaboration in shared virtual environments," in *Proceedings of the Fifth PHANToM Users Group Workshop*, 2000.

[44] S. Lee and J. Kim, "Transparency analysis and delay compensation scheme for haptic-based networked virtual environments," *Computer Communications*, vol. 32, no. 5, pp. 992–999, May 2009.

[45] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[46] M. Grunwald, *Human haptic perception: Basics and applications*. Birkhäuser, 2008.

[47] M. O. Ernst and H. H. Bülthoff, "Merging the senses into a robust percept.," *Trends in cognitive sciences*, vol. 8, no. 4, pp. 162–169, Apr. 2004.

[48] G. Robles-De-La-Torre, "The importance of the sense of touch in virtual and real environments," *IEEE Multimedia*, vol. 13, no. 3, pp. 24–30, Jul. 2006.

[49] S. Lederman and R. Klatzky, "Haptic perception: a tutorial," *Attention, Perception, & Psychophysics*, vol. 71, no. 7, pp. 1439–1459, Oct. 2009.

[50] L. Jones, "Kinesthetic sensing," *Human and Machine Haptics*, 2000.

[51] J. M. Loomis and S. J. Lederman, "Tactual perception," in *Handbook of perception and human performance*, K. R. Boff, L. Kaufman, and J. P. Thomas, Eds., vol. 2, Wiley New York, 1986.

[52] V. Hayward and K. Maclean, "Do it yourself haptics: part I," *IEEE Robotics & Automation Magazine*, vol. 14, no. 4, pp. 88–104, Jan. 2007.

[53] J. Taylor, "Proprioception," in *Encyclopedia of Neuroscience*, L. R. Squire, Ed., Oxford: Academic Press, 2009, pp. 1143 –1149.

[54] W. Bergmann Tiest and A. Kappers, "Cues for haptic perception of compliance," *IEEE Transactions on Haptics*, vol. 2, no. 4, pp. 189–199, Oct. 2009.

[55] M. A. Srinivasan and R. H. LaMotte, "Tactual discrimination of softness," *Journal of Neurophysiology*, vol. 73, no. 1, pp. 88–101, Jan. 1995.

[56] L. A. Jones and I. W. Hunter, "A perceptual analysis of stiffness," *Experimental Brain Research*, vol. 79, no. 1, pp. 150–156, Jan. 1990.

[57] M. Di Luca, B. Knörlein, M. O. Ernst, and M. Harders, "Effects of visual-haptic asynchronies and loading-unloading movements on compliance perception," *Brain Research Bulletin*, vol. 85, no. 5, pp. 245–259, Jun. 2011.

[58] H. Z. Tan, N. I. Durlach, Y. Shao, and M. Wei, "Manual resolution of compliance when work and force cues are minimized," *Advances in Robotics, Mechatronics and Haptic Interfaces*, vol. 49, pp. 99–104, 1993.

[59] H. Z. Tan, N. I. Durlach, G. Beauregard, and M. Srinivasan, "Manual discrimination of compliance using active pinch grasp: the role of force and work cues," *Perception & psychophysics*, vol. 57, no. 4, pp. 495–510, Nov. 1995.

[60] A. Pressman, L. J. Welty, A. Karniel, and F. A. Mussa-Ivaldi, "Perception of delayed stiffness," *The International Journal of Robotics Research*, vol. 26, pp. 1191–1203, Nov. 2007.

[61] G. A. Gescheider, *Psychophysics: The Fundamentals*, 3rd ed. Mahwah, NJ, USA: Lawrence Erlbaum, May 1997.

[62] E. H. Weber, *Die Lehre vom Tastsinn und Gemeingefühl auf Versuche gegründet*. Braunschweig, Germany, 1851.

[63] H. E. Ross and E. E. Brodie, "Weber fractions for weight and mass as a function of stimulus intensity.," *The Quarterly Journal of Experimental Psychology*, vol. 39, no. 1, pp. 77–88, May 1987.

[64] R. Höver, M. Di Luca, and M. Harders, "User-based evaluation of data-driven haptic rendering," *ACM Transactions on Applied Perception*, vol. 8, no. 1, pp. 1–23, Oct. 2010.

[65] E. Dorjgotov, G. R. Bertoline, L. Arns, Z. Pizlo, and S. R. Dunlop, "Force amplitude perception in six orthogonal directions," in *Proceedings of the IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2008, pp. 121–127.

[66] L. A. Jones, "Matching forces: constant errors and differential thresholds," *Perception*, vol. 18, no. 5, pp. 681–687, Oct. 1989.

[67] X. D. Pang, H. Z. Tan, and N. I. Durlach, "Manual discrimination of force using active finger motion.," *Perception & psychophysics*, vol. 49, no. 6, pp. 531–540, Jun. 1991.

[68] S. Allin, Y. Matsuoka, and R. Klatzky, "Measuring just noticeable differences for haptic force feedback: implications for rehabilitation," in *Proceedings of the IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2002, pp. 8–11.

[69] J. Kammerl, R. Chaudhari, and E. Steinbach, "Exploiting directional dependencies of force perception for lossy haptic data reduction," in *Proceedings of the IEEE International Symposium on Haptic Audio Visual Environments and Games*, Oct. 2010, pp. 1–6.

[70] F. E. van Beek, W. M. B. Tiest, and A. M. L. Kappers, "Anisotropy in the haptic perception of force direction and magnitude," *IEEE Transactions on Haptics*, vol. 6, no. 4, pp. 399–407, Dec. 2013.

[71] H. Z. Tan, F. Barbagli, K. Salisbury, C. Ho, and C. Spence, "Force-direction discrimination is not influenced by reference force direction," *Haptics-e*, pp. 1–6, 2006.

[72] F. Barbagli, K. Salisbury, C. Ho, C. Spence, and H. Z. Tan, "Haptic discrimination of force direction and the influence of visual information," *ACM Transactions on Applied Perception*, vol. 3, no. 2, pp. 125–135, Apr. 2006.

[73] V. Varadharajan, R. Klatzky, B. Unger, R. Swendsen, and R. Hollis, "Haptic rendering and psychophysical evaluation of a virtual three-dimensional helical spring," *IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 57–64, Mar. 2008.

[74] U. Koçak, K. L. Palmerius, C. Forsell, A. Ynnerman, and M. Cooper, "Analysis of the JND of stiffness in three modes of comparison," in *Haptic and Audio Interaction Design, Lecture Notes in Computer Science*, 2011, pp. 22–31.

[75] G. Paggetti, B. Cizmeci, C. Dillioglugil, and E. Steinbach, "On the discrimination of stiffness during pressing and pinching of virtual springs," in *Proceedings of the IEEE International Symposium on Haptic, Audio and Visual Environments and Games (HAVE)*, Nov. 2014, pp. 94–99.

[76] S. Choi and K. J. Kuchenbecker, "Vibrotactile display: perception, technology, and applications," *Proceedings of the IEEE*, vol. 101, no. 9, pp. 2093–2104, Sep. 2013.

[77] K. J. Kuchenbecker, J. Romano, and W. McMahan, "Haptography : capturing and recreating the rich feel of real surfaces," in *Proceedings of the 14th International Symposium on Robotics Research (ISRR)*, 2009, pp. 245–260.

[78] T. H. Massie and J. K. Salisbury, "The phantom haptic interface: A device for probing virtual objects," in *Proceedings of the ASME winter annual meeting, symposium on haptic interfaces for virtual environment and teleoperator systems*, vol. 55, 1994, pp. 295–300.

[79] Geomagic (3D Systems Inc.) (Jan. 2016). Geomagic touch haptic device, [Online]. Available: http://www.geomagic.com/en/products/phantom-omni/overview.

[80] J. E. Colgate and J. Brown, "Factors affecting the z-width of a haptic display," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1994, pp. 3205–3210.

[81] J. E. Colgate and G. G. Schenkel, "Passivity of a class of sampled-data systems: application to haptic interfaces," in *Proceedings of the American Control Conference*, vol. 3, Jun. 1994, pp. 3236–3240.

[82] ——, "Passivity of a class of sampled-data systems: application to haptic interfaces," *Journal of Robotic Systems*, vol. 14, no. 1, pp. 37–47, Jan. 1997.

[83] R. Gillespie and M. R. Cutkosky, "Stable user-specific haptic rendering of the virtual wall," *American Society of Mechanical Engineers, Dynamic Systems and Control Division*, vol. 58, no. 11, pp. 397–406, 1996.

[84] J. J. Abbott and A. M. Okamura, "Effects of position quantization and sampling rate on virtual-wall passivity," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 952–964, Sep. 2005.

[85] N. Diolaiti and G. Niemeyer, "Stability of haptic rendering: discretization, quantization, time-delay and coulomb effects," *IEEE Transactions on Robotics*, vol. 22, no. 2, pp. 1–13, Apr. 2006.

[86] J. Bao and P. L. Lee, *Process control: The passive systems approach*, 1st ed. London: Springer Science & Business Media, 2007.

[87] C. A. Desoer and M. Vidyasagar, *Feedback systems: Input-output properties*. SIAM, 1975.

[88] A. Jafari and J.-H. Ryu, "Input-to-state stable approach to release the conservatism of passivity-based stable haptic interaction," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 285–290.

[89] B. Hannaford and J.-H. Ryu, "Time-domain passivity control of haptic interfaces," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 1, pp. 1–10, Feb. 2002.

[90] D. Ruspini, K. Kolarov, and O. Khatib, "The haptic display of complex graphical environments," in *Proceedings of the 24th annual conference on computer graphics and interactive techniques*, Aug. 1997, pp. 345–352.

[91] C.-H. Ho, C. Basdogan, and M. A. Srinivasan, "Efficient point-based rendering techniques for haptic display of virtual objects," *Presence: Teleoperators and Virtual Environments*, vol. 8, no. 5, pp. 477–491, Oct. 1999.

[92] C. Basdogan and M. A. Srinivasan, "Haptic rendering in virtual environments," in *Handbook of Virtual Environments: Design, implementation, and applications*, 1, vol. 1996, 2002, pp. 117–134.

[93] K. Salisbury, F. Conti, and F. Barbagli, "Haptic rendering: introductory concepts," *IEEE Computer Graphics and Applications*, vol. 24, no. 2, pp. 24–32, Apr. 2004.

[94] J. J. Gil, A. Avello, Á. Rubio, and J. Flórez, "Stability analysis of a 1 DOF haptic interface using the routh-hurwitz criterion," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 4, pp. 583–588, Jul. 2004.

[95] J.-H. Ryu, Y. S. Kim, and B. Hannaford, "Sampled-and continuous-time passivity and stability of virtual environments," *IEEE Transactions on Robotics*, vol. 20, no. 4, pp. 772–776, Aug. 2004.

[96] J.-H. Ryu, C. Preusche, B. Hannaford, and G. Hirzinger, "Time domain passivity control with reference energy following," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 5, pp. 737–742, Sep. 2005.

[97] R. Adams and B. Hannaford, "Stable haptic interaction with virtual environments," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 465–474, Jun. 1999.

[98] A. Jafari, M. Nabeel, and J.-H. Ryu, "Multi degree-of-freedom input-to-state stable approach for stable haptic interaction," in *Proceedings of the IEEE World Haptics Conference (WHC)*, Jun. 2015, pp. 293–298.

[99] D. Kubus, I. Weidauer, and F. M. Wahl, "1khz is not enough - how to achieve higher update rates with a bilateral teleoperation system based on commercial hardware," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 5107–5114.

[100] S. Choi and H. Z. Tan, "Perceived instability of virtual haptic texture. I. Experimental studies," *Presence: Teleoperators and Virtual Environments*, vol. 13, no. 4, pp. 395–415, Mar. 2004.

[101] A. El Saddik, M. Orozco, M. Eid, and J. Cha, *Haptics technologies: Bringing touch to multimedia*, 1st ed. Berlin: Springer Science, 2011.

[102] U. Meier, O. López, C. Monserrat, M. C. Juan, and M. Alcañiz, "Real-time deformable models for surgery simulation: a survey.," *Computer methods and programs in biomedicine*, vol. 77, no. 3, pp. 183–97, Mar. 2005.

[103] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson, "Physically based deformable models in computer graphics," *Computer Graphics Forum*, vol. 25, no. 4, pp. 809–836, Dec. 2006.

[104] I. Peterlik, M. Nouicer, C. Duriez, S. Cotin, and A. Kheddar, "Constraint-based haptic rendering of multirate compliant mechanisms," *IEEE Transactions on Haptics*, vol. 4, no. 3, pp. 175–187, May 2011.

[105] D. Baraff, "An introduction to physically based modeling: Rigid body simulation i – unconstrained rigid body dynamics," *SIGGRAPH Course Notes*, 1997.

[106] ——, "An introduction to physically based modeling: Rigid body simulation ii – nonpenetration constraints," *SIGGRAPH Course Notes*, 1997.

[107] D. M. Kaufman, S. Sueda, D. L. James, and D. K. Pai, "Staggered projections for frictional contact in multibody systems," *ACM Transactions on Graphics*, vol. 27, no. 5, 164:1–164:11, Dec. 2008.

[108] J. Bender, K. Erleben, and J. Trinkle, "Interactive simulation of rigid body dynamics in computer graphics," in *Proceedings of the Computer Graphics Forum*, vol. 33, 2014, pp. 246–270.

[109] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics : Comparison of Bullet , Havok , MuJoCo , ODE and PhysX," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 4397 –4404.

[110] A. Boeing and T. Bräunl, "Evaluation of real-time physics simulation systems," in *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, vol. 1, 2007, pp. 281–288.

[111] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot, "Realistic haptic rendering of interacting deformable objects in virtual environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 1, pp. 36–47, Jan. 2006.

[112] J. Barbic, "Real-time reduced large-deformation models and distributed contact for computer graphics and haptics," PhD thesis, 2007.

[113] M. Müller and M. Gross, "Interactive virtual materials," *Proceedings of Graphics Interface (GI)*, pp. 239–246, 2004.

[114] A. Pentland and J. Williams, "Good vibrations: Modal dynamics for graphics and animation," *ACM SIGGRAPH Computer Graphics*, vol. 23, no. 3, pp. 207–214, Jul. 1989.

[115] C. Basdogan, "Real-time simulation of dynamically deformable finite element models using modal analysis and spectral lanczos decomposition methods," in *Proceedings of the Medicine Meets Virtual Reality (MMVR) Conference*, Jan. 2001.

[116] J. Barbic and D. L. James, "Six-dof haptic rendering of contact between geometrically complex reduced deformable models," *IEEE Transactions on Haptics*, vol. 1, no. 1, pp. 39–52, Jun. 2008.

[117] Z. Tang, Y. Yang, X. Guo, and B. Prabhakaran, "Distributed haptic interactions with physically based 3d deformable models over lossy networks," *IEEE Transactions on Haptics*, vol. 6, no. 4, pp. 417–428, Oct. 2013.

[118] M. G. Choi and H.-S. Ko, "Modal warping: real-time simulation of large rotational deformation and manipulation," *IEEE Transactions on Visualizatiousn and Computer Graphics*, vol. 11, no. 1, pp. 91–101, Feb. 2005.

[119] G. Bianchi, B. Solenthaler, G. Székely, and M. Harders, "Simultaneous topology and stiffness identification for mass-spring models based on fem reference deformations," in *Medical Image Computing and Computer-Assisted Intervention*, vol. 3217, Sep. 2004, pp. 293–301.

[120] A. Duysak, J. J. Zhang, and V. Ilankovan, "Efficient modelling and simulation of soft tissue deformation using mass-spring systems," in *Proceedings of the 17th International Congress and Exhibition on Computer Assisted Radiology and Surgery*, 2003, pp. 337 –342.

[121] M. Sedef, E. Samur, and C. Basdogan, "Real-time finite element simulation of linear viscoelastic tissue behavior based on experimental data," *IEEE Computer Graphics and Applications*, vol. 26, no. 6, pp. 28–38, Nov. 2006.

[122] I. Peterlík, M. Sedef, C. Basdogan, and L. Matyska, "Real-time visio-haptic interaction with static soft tissue models having geometric and material nonlinearity," *Computers & Graphics*, vol. 34, no. 1, pp. 43–54, Feb. 2010.

[123] M. Lin and S. Gottschalk, "Collision detection between geometric models: a survey," in *Proceedings of the IMA Conference on Mathematics of Surfaces*, May 1998, pp. 602–608.

[124] P. Jiménez, F. Thomas, and C. Torras, "3D collision detection: A survey," *Computers and Graphics*, vol. 25, no. 2, pp. 269–285, Apr. 2001.

[125] T. Larsson and T. Akenine-Möller, "Collision detection for continuously deforming bodies," *Eurographics*, vol. 19, pp. 325–333, 2001.

[126] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino, "Collision detection for deformable objects," *Computer Graphics Forum*, vol. 24, no. 1, pp. 61–81, Mar. 2005.

[127] G. van den Bergen, "Efficient collision detection of complex deformable models using AABB trees," *Journal of Graphics Tools*, vol. 2, no. 4, pp. 1–13, Jan. 1997.

[128] S. Gottschalk, M. C. Lin, and D. Manocha, "Obbtree: A hierarchical structure for rapid interference detection," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, Aug. 1996, pp. 171–180.

[129] C. Zilles and J. Salisbury, "A constraint-based god-object method for haptic display," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, Aug. 1995, pp. 146–151.

[130] K. Salisbury and C. Tarr, "Haptic rendering of surfaces defined by implicit functions," in *Proceedings of the IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator System*, Nov. 1997, pp. 61–68.

[131] S. Choi and H. Z. Tan, "Haptic rendering of surface textures," *IEEE Computer Graphics and Applications*, vol. 24, no. 2, pp. 40–47, Apr. 2004.

[132] H. Culbertson, J. J. L. Delgado, and K. J. Kuchenbecker, "One hundred data-driven haptic texture models and open-source methods for rendering on 3d objects," in *Proceedings of the IEEE Haptics Symposium (HAPTICS)*, Feb. 2014, pp. 319–325.

[133] F. Rydén and H. J. Chizeck, "A proxy method for real-time 3-dof haptic rendering of streaming point cloud data," *IEEE Transactions on Haptics*, vol. 6, no. 3, pp. 257–267, Jul. 2013.

[134] F. Rydén, S. N. Kosari, and H. J. Chizeck, "Proxy method for fast haptic rendering from time varying point clouds," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 2614–2619.

[135] M. Ortega, S. Redon, and S. Coquillart, "A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 3, pp. 458–469, May 2007.

[136] W. S. Harwin and N. Melder, "Improved haptic rendering for multi-finger manipulation using friction cone based god-objects," in *Proceedings of the Eurohaptics Conference*, 2002.

[137] M. Altomonte, D. Zerbato, D. Botturi, and P. Fiorini, "Simulation of deformable environment with haptic feedback on gpu," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2008, pp. 3959–3964.

[138] A. Frank, I. Twombly, T. Barth, and J. Smith, "Finite element methods for real-time haptic feedback of soft-tissue models in virtual reality simulators," in *Proceedings of the IEEE Virtual Reality Conference*, 2001, pp. 257–263.

[139] Y. Adachi, T. Kumano, and K. Ogino, "Intermediate representation for stiff virtual objects," in *Proceedings of the Virtual Reality Annual International Symposium*, 1995, pp. 203–210.

[140] F. Barbagli, P. Domenico, and K. Salisbury, "A multirate approach to haptic interaction with deformable objects single and multipoint contacts," *The International Journal of Robotics Research*, vol. 24, no. 9, pp. 703–715, Sep. 2005.

[141] G. Saupin, C. Duriez, and S. Cotin, "Contact model for haptic medical simulations," in *Proceedings of the International Symposium on Biomedical Simulation*, Springer, Jul. 2008, pp. 157–165.

[142] S. Ullrich and T. Kuhlen, "Haptic palpation for medical simulation in virtual environments.," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 4, pp. 617–25, Apr. 2012.

[143] S. Chan, N. H. Blevins, and K. Salisbury, "Deformable haptic rendering for volumetric medical image data," in *Proceedings of the IEEE World Haptics Conference (WHC)*, Apr. 2013, pp. 73–78.

[144] T. C. Knott and T. W. Kuhlen, "Accurate contact modeling for multi-rate single-point haptic rendering of static and deformable environments," in *Proceedings of the Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS)*, 2015, pp. 71–80.

[145] M. Cavusoglu and F. Tendick, "Multirate simulation for high fidelity haptic interaction with deformable objects in virtual environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, 2000, pp. 2458–2465.

[146] P. Jacobs, M. J. Fu, and M. C. Cavusoglu, "High fidelity haptic rendering of frictional contact with deformable objects in virtual environments using multi-rate simulation," *The International Journal of Robotics Research*, vol. 29, no. 14, pp. 1778–1792, Sep. 2010.

[147] C. Basdogan, C.-H. Ho, M. Slater, and M. Srinivasan, "The role of haptic communication in shared virtual environments," in *Proceedings of the Third PHANToM Users Group Workshop*, 1998, pp. 1–5.

[148] IEEE, "IEEE standard for distributed interactive simulation–application protocols," *IEEE Std 1278.1-2012 (Revision of IEEE Std 1278.1-1995)*, pp. 1 –747, 2012.

[149] D. Lawrence, "Stability and transparency in bilateral teleoperation," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 624–637, Oct. 1993.

[150] G. Raju, G. Verghese, and T. Sheridan, "Design issues in 2-port network models of bilateral remote teleoperation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 1989, pp. 1316–1321.

[151] Y. Yokokohji and T. Yoshikawa, "Bilateral control of master-slave manipulators for ideal kinesthetic coupling-formulation and experiment," *IEEE Robotics and Automation*, vol. 10, no. 5, Oct. 1994.

[152] S. Niakosari and S. Sirouspour, "Improving transparency in network-based haptics," in *Proceedings of the IEEE World Haptics Conference*, 2009, pp. 547–552.

[153] S. Hirche and M. Buss, "Human-oriented control for haptic teleoperation," *Proceedings of the IEEE*, vol. 100, no. 3, pp. 623–647, Mar. 2012.

[154] D. Delaney, T. Ward, and S. McLoone, "On consistency and network latency in distributed interactive applications: a survey – part I," *Presence: Teleoperators and Virtual Environments*, vol. 15, no. 2, pp. 218–234, Apr. 2006.

[155] ——, "On consistency and network latency in distributed interactive applications: a survey – part II," *Presence: Teleoperators and Virtual Environments*, vol. 15, no. 4, pp. 465–482, Aug. 2006.

[156] K. Hikichi, H. Morino, I. Fukuda, S. Matsumoto, Y. Yasuda, I. Arimoto, M. Iijima, and K. Sezaki, "Architecture of haptics communication system for adaptation to network environments," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, Aug. 2001, pp. 744–747.

[157] Y. Ishibashi, T. Kanbara, T. Hasegawa, and S. Tasaka, "Traffic control of haptic media in networked virtual environments," in *Proceedings of the IEEE Workshop on Knowledge Media Networking*, Jul. 2002, pp. 11–16.

[158] M. Fujimoto and Y. Ishibashi, "Packetization interval of haptic media in networked virtual environments," in *Proceedings of 4th ACM SIGCOMM Workshop on Network and System Support for Games (NetGames)*, ACM Press, 2005, pp. 1–6.

[159] S. Lee and J. Kim, "Dynamic network adaptation scheme employing haptic event priority for collaborative virtual environments," in *Proceedings of the International Conference on Immersive Telecommunications (ImmersCom)*, 2007, 12:1–12:6.

[160] R. Rakhsha and D. Constantinescu, "Passive shared virtual environment for distributed haptic cooperation," in *Proceedings of the IEEE Haptics Symposium (HAPTICS)*, Feb. 2014, pp. 221–226.

[161] ——, "Distributed haptic cooperation with passive multirate wave communications," in *Proceedings of the IEEE Haptics Symposium (HAPTICS)*, Mar. 2012, pp. 117–123.

[162] G. Sankaranarayanan and B. Hannaford, "Experimental comparison of internet haptic collaboration with time-delay compensation techniques," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 206–211.

[163] K. Hikichi, H. Morino, and I. Arimoto, "The evaluation of delay jitter for haptics collaboration over the internet," in *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, Nov. 2002, pp. 1492–1496.

[164] M. Fujimoto and Y. Ishibashi, "A compensation scheme for network delay jitter of haptic media in networked virtual environments," in *Proceedings of the 8th World Multi-Conference on Systemics, Cybernetics, and Informatics (SCI)*, 2004, pp. 30 –34.

[165] M. Glencross, C. Jay, J. Feasel, L. Kohli, and M. Whitton, "Effective cooperative haptic interaction over the internet," in *Proceedings of the IEEE Virtual Reality Conference*, 2007, pp. 115–122.

[166] G. Sankaranarayanan and B. Hannaford, "Experimental internet haptic collaboration using virtual coupling schemes," in *Proceedings of the IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2008, pp. 259–266.

[167] J. Cheong, S.-I. Niculescu, and C. Kim, "Motion synchronization control of distributed multisubsystems with invariant local natural dynamics," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 382–398, Apr. 2009.

[168] K. Huang and D. Lee, "Consensus-based peer-to-peer control architecture for multiuser haptic interaction over the internet," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 417–431, Apr. 2013.

[169] J. Kammerl, R. Chaudhari, and E. Steinbach, "Combining contact models with perceptual data reduction for efficient haptic data communication in networked VEs," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 1, pp. 57–68, Jan. 2011.

[170] R. Shea, J. Liu, E. C.-H. Ngai, and Y. Cui, "Cloud gaming: architecture and performance," *IEEE Network*, vol. 27, no. 4, pp. 16–21, Jul. 2013.

[171] R. Anderson and M. Spong, "Bilateral control of teleoperators with time delay," *IEEE Transactions on Automatic Control*, vol. 34, no. 5, pp. 494–501, May 1989.

[172] G. Niemeyer and J.-J. Slotine, "Stable adaptive teleoperation," *IEEE Journal of Oceanic Engineering*, vol. 16, no. 1, pp. 152–162, Jan. 1991.

[173] J.-H. Ryu, J. Artigas, and C. Preusche, "A passive bilateral control scheme for a teleoperator with time-varying communication delay," *Mechatronics*, vol. 20, no. 7, pp. 812–823, Oct. 2010.

[174] S. Lee and J. Kim, "Priority-based haptic event filtering for transmission and error control in networked virtual environments," *Multimedia Systems*, vol. 15, no. 6, pp. 355–367, Dec. 2009.

[175] M. Claypool and K. Claypool, "Latency and player actions in online games," *Communications of the ACM*, vol. 49, no. 11, pp. 40–45, Nov. 2006.

[176] P. Mitra and G. Niemeyer, "Model-mediated telemanipulation," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 253–262, Feb. 2008.

[177] C. Passenberg, A. Peer, and M. Buss, "A survey of environment-, operator-, and task-adapted controllers for teleoperation systems," *Mechatronics*, vol. 20, no. 7, pp. 787–801, Oct. 2010.

[178] J. Kim, H. Kim, B. K. Tay, M. Muniyandi, M. A. Srinivasan, J. Jordan, J. Mortensen, M. Oliveira, and M. Slater, "Transatlantic touch: a study of haptic collaboration over long distance," *Presence: Teleoperators and Virtual Environments*, vol. 13, no. 3, pp. 328–337, Jun. 2004.

[179] J. Qin, K.-S. Choi, R. Xu, W.-M. Pang, and P.-A. Heng, "Effect of packet loss on collaborative haptic interactions in networked virtual environments: an experimental study," *Presence: Teleoperators and Virtual Environments*, vol. 22, no. 1, pp. 36–53, Feb. 2013.

[180] Y. Ishibashi, T. Hasegawa, and S. Tasaka, "Group synchronization control for haptic media in networked virtual environments," in *Proceedings of the IEEE International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2004, pp. 106–113.

[181] J. E. Lengyel, "Compression of time-dependent geometry," in *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, 1999, pp. 89–96.

[182] A. Marshall, K. Yap, and W. Yu, "Quality of service issues for distributed virtual environments with haptic interfaces," in *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, Oct. 2008, pp. 40–45.

[183] K. S. Perumalla, "Parallel and distributed simulation: Traditional techniques and recent advances," in *Proceedings of the 38th Conference on Winter Simulation*, 2006, pp. 84–95.

[184] S. Jafer, Q. Liu, and G. Wainer, "Synchronization methods in parallel and distributed discrete-event simulation," *Simulation Modelling Practice and Theory*, vol. 30, pp. 54–73, Jan. 2013.

[185] C. Shahabi, A. Ortega, and M. R. Kolahdouzan, "A comparison of different haptic compression techniques," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2002, pp. 657 –660.

[186] P. Otanez, J. Moyne, and D. Tilbury, "Using deadbands to reduce communication in networked control systems," in *Proceedings of the IEEE American Control Conference*, vol. 4, 2002, pp. 3015–3020.

[187] A. Bhardwaj, S. Chaudhuri, and O. Dabeer, "Design and analysis of predictive sampling of haptic signals," *ACM Transactions on Applied Perception*, vol. 11, no. 4, pp. 1–20, Jan. 2015.

[188] P. Hinterseer, E. Steinbach, S. Hirche, and M. Buss, "A novel, psychophysically motivated transmission approach for haptic data streams in telepresence and teleaction systems," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Mar. 2005, pp. 1097–1100.

[189] P. Hinterseer, E. Steinbach, and S. Chaudhuri, "Perception-based compression of haptic data streams using kalman filters," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2006, pp. 473–476.

[190] ——, "Model based data compression for 3d virtual haptic teleinteraction," in *Proceedings of the International Conference on Consumer Electronics (ICCE)*, 2006, pp. 23–24.

[191] J. Kammerl, I. Vittorias, V. Nitsch, B. Faerber, E. Steinbach, and S. Hirche, "Perception-based data reduction for haptic force-feedback signals using velocity-adaptive deadbands," *Presence: Teleoperators and Virtual Environments*, vol. 19, no. 5, pp. 450–462, Oct. 2010.

[192] N. Sakr, J. Zhou, N. D. Georganas, and J. Zhao, "Prediction-based haptic data reduction and transmission in telementoring systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 5, pp. 1727–1736, Apr. 2009.

[193] N. Sakr, N. D. Georganas, and J. Zhao, "Human perception-based data reduction for haptic communication in six-dof telepresence systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 11, pp. 3534–3546, Oct. 2011.

[194] O. Dabeer and S. Chaudhuri, "Analysis of an adaptive sampler based on weber's law," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1868–1878, Apr. 2011.

[195] F. Brandi, J. Kammerl, and E. Steinbach, "Error-resilient perceptual coding for networked haptic interaction," in *Proceedings of the ACM Multimedia Conference*, ACM Press, 2010, pp. 351 –360.

[196] J. Kammerl, F. Brandi, F. Schweiger, and E. Steinbach, "Error-resilient perceptual haptic data communication based on probabilistic receiver state estimation," in *Proceedings of the Eurohaptics Conference*, Tampere, Finland, Jun. 2012.

[197] V. Gokhale, J. Nair, and S. Chaudhuri, "Opportunistic adaptive haptic sampling on forward channel in telehaptic communication," in *Proceedings of the IEEE Haptics Symposium (HAPTICS)*, Apr. 2016, pp. 217–222.

[198] C. Murphy, "Believable dead reckoning for networked games," in *Game Engine Gems 2*, E. Lengyel, Ed., Natick, MA, USA: A K Peters/CRC Press, 2011, pp. 307–328.

[199] A. Maglo, G. Lavoué, F. Dupont, and C. Hudelot, "3D mesh compression: Survey, comparisons, and emerging trends," *ACM Computing Surveys*, vol. 47, no. 3, pp. 1–41, Feb. 2015.

[200] L. Váša, S. Marras, K. Hormann, and G. Brunnett, "Compressing dynamic meshes with geometric laplacians," *Computer Graphics Forum*, vol. 33, no. 2, pp. 145–154, Jun. 2014.

[201] L. Váša and V. Skala, "Coddyac: connectivity driven dynamic mesh compression," in *Proceedings of the 3DTV Conference*, 2007, pp. 3–6.

[202] K. Mamou, T. Zaharia, and F. Prêteux, "Famc: The MPEG-4 standard for animated mesh compression," *IEEE International Conference on Image Processing (ICIP)*, pp. 2676–2679, Oct. 2008.

[203] N. Stefanoski and J. Ostermann, "SPC: fast and efficient scalable predictive coding of animated meshes," *Computer Graphics Forum*, vol. 29, no. 1, pp. 101–116, Mar. 2010.

[204] K. Müller, A. Smolic, M. Kautzner, P. Eisert, and T. Wiegand, "Predictive compression of dynamic 3D meshes," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Sep. 2005, pp. 264–267.

[205] M. Alexa and W. Müller, "Representing animations by principal components," *Eurographics*, vol. 19, no. 3, pp. 411–418, Sep. 2000.

[206] L. Ibarria and J. Rossignac, "Dynapack: Space-time compression of the 3D animations of triangle meshes with fixed connectivity," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Jul. 2003, pp. 126–135.

[207] S. Varakliotis, J. Ostermann, and V. Hardman, "Coding of animated 3-D wireframe models for internet streaming applications," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2001, pp. 353–356.

[208] E. S. Jang, J. D. K. Kim, S. Y. Jung, M. J. Han, S. O. Woo, and S. J. Lee, "Interpolator data compression for MPEG-4 animation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 7, pp. 989–1008, Jul. 2004.

[209] M. Corsini, M. C. Larabi, G. Lavoué, O. Petrík, L. Váša, and K. Wang, "Perceptual metrics for static and dynamic triangle meshes," *Computer Graphics Forum*, vol. 32, no. 1, pp. 101–125, Jan. 2013.

[210] W. Freund, "Client-server based haptic interaction with deformable objects," Master Thesis, Chair of Media Technology, Technical University of Munich, 2015.

[211] R. Smith. (Jan. 2016). Open dynamics engine (ODE), [Online]. Available: http://www.ode.org.

[212] R. Chaudhari, E. Steinbach, and S. Hirche, "Towards an objective quality evaluation framework for haptic data reduction," in *Proceedings of the IEEE World Haptics Conference*, Istanbul, Turkey, Jun. 2011, pp. 539–544.

[213] K. J. Kuchenbecker, J. Fiene, and G. Niemeyer, "Improving contact realism through event-based haptic feedback," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 2, pp. 219–230, Mar. 2006.

[214] D. Salomon, *Curves and surfaces for computer graphics*. New York, NY, USA: Springer, 2006.

[215] X. Zhang, T. E. Ward, and S. Mcloone, "An information-based dynamic extrapolation model for networked virtual environments," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 8, no. 3, pp. 1–19, Jul. 2012.

[216] D. M. Ennis and J. M. Ennis, "Accounting for no difference/preference responses or ties in choice experiments," *Food Quality and Preference*, vol. 23, no. 1, pp. 13–17, Jan. 2012.

[217] K. Shoemake, "Animating rotation with quaternion curves," *ACM SIGGRAPH Computer Graphics*, vol. 19, no. 3, pp. 245–254, Jul. 1985.

[218] A. Bhardwaj and S. Chaudhuri, "Does just noticeable difference depend on the rate of change of kinesthetic force stimulus?" In *Proceedings of the Eurohaptics Conference*, Jun. 2014, pp. 40–47.

[219] R. Ott and M. Longnecker, *An introduction to statistical methods and data analysis*. Cengage Learning, 2008.

[220] G. W. Corder and D. I. Foreman, *Nonparametric statistics for non-statisticians: A step-by-step approach*. John Wiley & Sons, 2009.

[221] G. N. N. Martin, "Range encoder range encoding: an algorithm for removing redundancy from a digitized message," in *Proceedings of the Video & Data Recording Conference*, 1979, p. 2427.

[222] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2012, pp. 778–785.

[223] I. H. Witten, A. Moffat, and T. C. Bell, *Managing gigabytes: Compressing and indexing documents and images*. Morgan Kaufmann, 1999.

[224] F. Conti, F. Barbagli, D. Morris, and C. Sewell. (Jan. 2016). Chai 3d: An open-source library for the rapid development of haptic scenes, [Online]. Available: http://www.chai3d.org.

[225] J. Barbič, F. S. Sin, and D. Schroeder. (Jan. 2016). Vega FEM library, [Online]. Available: http://run.usc.edu/vega/.

[226] G. Niemeyer and J.-J. E. Slotine, "Telemanipulation with time delays," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 873–890, Sep. 2004.

[227] R. Dorf and R. Bishop, *Modern Control Systems*. Pearson Prentice Hall, 2008.

[228] S. Hirche and M. Buss, "Advances in telerobotics," in, M. Ferre, M. Buss, R. Aracil, C. Melchiorri, and C. Balaguer, Eds. Springer Berlin Heidelberg, 2007, ch. Human Perceived Transparency with Time Delay, pp. 191–209.

[229] B. Knörlein, M. Di Luca, and M. Harders, "Influence of visual and haptic delays on stiffness perception in augmented reality," in *Proceedings of the IEEE 2009 International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009, pp. 49–52.

[230] M. H. Zadeh, D. Wang, and E. Kubica, "Perception-based lossy haptic compression considerations for velocity-based interactions," *Multimedia Systems*, vol. 13, no. 4, pp. 275–282, Nov. 2007.

[231] S. Suzuki, K. Matsunaga, H. Ohnishi, and Y. Ishibashi, "Influences of network delay variation on haptic perception under adaptive reaction force control," in *Proceedings of the IEEE Global Conference on Consumer Electronics (GCCE)*, 2014, pp. 669–673.

[232] S. P. Tinta, A. E. Mohr, J. L. Wong, and S. Brook, "Characterizing end-to-end packet reordering with udp traffic," in *Proceedings of the IEEE Symposium on Computers and Communications*, 2009, pp. 321–324.

[233] Y. J. Liang, N. Färber, and B. Girod, "Adaptive playout scheduling and loss concealment for voice communication over IP networks," *IEEE Transactions on Multimedia*, vol. 5, no. 4, pp. 532–543, Dec. 2003.

[234] C. Bovy, H. T. Mertodimedjo, G. Hooghiemstra, H. Uijterwaal, and P. V. Mieghem, "Analysis of end-to-end delay measurements in internet," in *Proceedings of the PAM Conference: Passive & Active Measurement*, 2002, pp. 1–8.

[235] N. A. Macmillan and D. Creelman, *Detection Theory: A User's Guide*, 2nd ed. Lawrence Erlbaum Associates, 1991.

[236] E. Jury, "A simplified stability criterion for linear discrete systems," *Proceedings of the IRE*, vol. 50, no. 6, pp. 1493–1500, Jun. 1962.

[237] M. Mahvash and V. Hayward, "High-fidelity passive force-reflecting virtual environments," *IEEE Transactions on Robotics*, vol. 21, pp. 38–46, Feb. 2005.

[238] G. De Gersem, H. V. Brussel, and F. Tendick, "Reliable and enhanced stiffness perception in soft-tissue telemanipulation," *The International Journal of Robotics Research*, vol. 24, no. 10, pp. 805–822, Oct. 2005.

[239] X. Xu and E. Steinbach, "Towards real-time modeling and haptic rendering of deformable objects for point cloud-based model-mediated teleoperation," in *5th IEEE International Workshop on Hot Topics in 3D (Hot3D), Proceedings of IEEE International Conference on Multimedia & Expo*, Chengdu, China, Jul. 2014.

[240] G. P. Fettweis, "The tactile internet: Applications and challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, Mar. 2014.

[241] A. Aijaz, M. Dohler, A. H. Aghvami, V. Friderikos, and M. Frodigh, "Realizing the tactile internet: haptic communications over next generation 5G cellular networks," *IEEE Wireless Communications*, pp. 2–9, Oct. 2016.

[242] B. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1617–1634, Mar. 2014.

[243] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 655–685, Jan. 2016.

[244] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, May 2010.

List of Figures

List of Tables