SHARED CONTROL FOR DEXTEROUS TELEMANIPULATION WITH HAPTIC FEEDBACK

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Weston Blaine Griffin

June 2003

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Mark R. Cutkosky, Principal Advisor

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Oussama Khatib

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Kenneth J. Waldron

Approved for the University Committee on Graduate Studies:

Abstract

Telemanipulation systems provide human operators with the ability to see, touch, and feel objects from a remote location. In telemanipulation, a remote robot is controlled by a human and interacts with an environment while relaying information back to the human, providing access to environments which may be hostile, hazardous, or difficult to access.

A new dexterous telemanipulation system has been developed, which enables an operator to use a glove-based interface to control a robotic hand for remote grasping and manipulation tasks. In addition to visual feedback, forces from the remote manipulator are fed back to the operator's fingertips to create a more immersive experience.

By combining teleoperation with techniques from autonomous dexterous manipulation, the remote robot hand has the ability to control low-level grasping forces and motions. By providing local control, the robot allows the operator to concentrate on the higher-level task requirements of a grasped object or tool. This thesis focuses on the development of a shared control system that combines high-level and low-level operator commands with those of a semi-autonomous robotic hand for remote object manipulation. Fingertip forces are relayed back to the operator from the remote manipulator. This feedback is augmented with audio, visual, and haptic cues to inform the operator when the robot intervenes.

Shared control has the potential to overcome some of the limitations imposed by traditional telemanipulation architectures. Problems such as time delays and limitations in the fidelity of the master interface become less detrimental because commands from the operator are supplemented by local control. However, there is some concern that the operator's sense of presence will be reduced as the remote robot takes over more of the control. A set of experiments was therefore designed to evaluate the efficacy of shared control for dexterous telemanipulation and to determine what combinations of force, visual and audio feedback provide the best performance and operator sense of presence. The results demonstrate the benefits of shared control and the need to choose carefully the types and methods of direct and indirect feedback.

For my parents.

Acknowledgments

While my doctoral work is supposed to represent an individual contribution, I would not have been able to complete any of it without the support of my colleagues, friends, and family. I owe a large debt of gratitude to all those that have helped me along the way.

First, I would like to thank my advisor, Mark Cutkosky. While at times his brilliance can be somewhat humbling, it is only out matched by his patience in teaching and passion for all things mechanical. His easygoing and gentle nature has an amazing ability to inspire his students to work harder than they ever have before and seek their own path. Yet, I could always count on him to provide direction and support whenever I felt lost, stressed, or lacked the confidence to continue.

I would also like to acknowledge the members of my reading and defense committees: Oussama Khatib, Ken Waldron, Kenneth Salisbury, and Günter Niemeyer, for their helpful suggestions for completing this work.

I owe a special thanks to Charles Pigott for his generosity. My research assistantship was funded through a fellowship created by Mr. Pigott. I am grateful for his continued support of Stanford's engineering program.

I would like to thank Prof. Keith Buffinton at Bucknell University for his guidance and support. Prof. Buffinton taught me a great deal of things about mechanical engineering, as well as life outside the classroom. I truly appreciated his support for my interests in engineering research and robotics.

At Stanford I was fortunate to be surrounded by an incredible number of supportive, friendly, and good natured people. Among the many people I met and worked with, I would like to thank my class professors (especially Ed Carryer and Bernie Roth), all my various project team members and friends (especially Adam and Erik), and the CDR staff members, Noelle, Judy, and Jeff, who helped with all things administrative, making my life a little bit easier.

I feel extremely fortunate to have joined such an excellent lab group as the Dexterous Manipulation Laboratory. My lab mates in the DML have always been willing to listen to my problems or lend me a hand when needed. However, the DML members are much more than colleagues, they have been my good friends through out the years. A special thanks goes out to previous lab members; Allison, Costa, Chris, Findley, Jorge, and of course Turner, with whom I worked closely with and learned so much from. I would also like to thank the current lab members (even though they all worked on the biomimetics "side":) for their support: Jonathan C., Jonathan K., Trey, and Moto. I wish them all the luck in their future endeavors.

A very important thanks goes to Will, who provided invaluable assistance in developing and completing the experiments needed for my thesis. I could not have asked for a better colleague and friend to work with.

I grateful to have made so many friends during my time at Stanford. The list is too long for me to cover but I would like say a very special thanks to Margy and Sean. Margy and I have shared many good times together. Her tenacious spirit and compassionate ideals will always be with me. And without Sean and his adventurous nature, I would probably still be stuck to my couch. From impromptu camping trips and mountain bike rides, to touring Europe with only a plan for the day, he has helped me to live life to the fullest and enjoy my time here at Stanford.

And last, but certainly not least, I would like to thank my family; my parents, David and Sheila, and my brothers, David and Marshall. Words simply can not express my gratitude for my family's love and support. They have been with me every step of the way and I could not have done it without them.

Contents

Tables

Figures

1 Introduction

Humans have a remarkable ability to grasp and manipulate objects with their hands. The high level of dexterity is achieved through complex sensorimotor mechanisms utilizing visual and tactile information and the physical structure of the hand. Utilizing these abilities, humans can modulate grasp forces, precisely position objects, and detect fine surface features. Consider how easily one can screw in a light bulb. This action requires a delicate grasp to prevent breakage, in addition to careful control over insertion force and position to engage the threads. Humans are also capable of readily dealing with environmental uncertainty or adapting to environmental changes.

In some situations robots are more aptly suited than their human counterparts. For example, robots used in automobile factories can assemble components with greater speed, accuracy, and endurance than a human worker. Robots can also be used in environments that are hazardous or dangerous for humans (e.g., radioactive sites) or difficult to access (e.g., deep underwater or in space) or not at human scale (e.g., microsurgery). Today, autonomous robot operation is practical only when the environment is highly structured (e.g., factory automation). Programming robotic systems to autonomously execute tasks in unstructured environments is extremely difficult. Even in structured environments, robots are relegated to relatively simple manipulations involving force and motion control of the robot arm and wrist. Dexterous manipulation, in which fine motions and forces are imparted with the fingertips, remains a topic of research.

A practical alternative to autonomous dexterous manipulation is dexterous telemanipulation. Teleoperation can provide human operators with the ability to see, touch, and feel objects from a remote location. In teleoperation, a remote robot is controlled by a human and interacts with an environment while relaying information back to the human. In other words, a teleoperated system extends a person's sensing and/or manipulation capability to a remote location [Sheridan 1992a]. By having the human "in the loop," the system

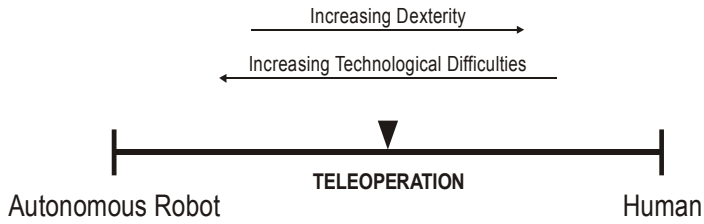

Figure 1-1. Teleoperation systems fall between fully autonomous robotic systems and humans.

benefits from natural human abilities in spatial reasoning, task planning, and adaptation, thus reducing the complexity as compared to a purely autonomous system. Additionally, teleoperation systems can provide access to environments that are hostile, hazardous, or difficult to access. In this way, teleoperation represents a middle ground between autonomous robots and humans (see Figure 1-1).

1.1 Motivation

Many current telemanipulation systems rely heavily on visual feedback and experienced operators. The sense of telepresence provided to the operator can be greatly enhanced with the addition of force (or haptic) feedback[1]. Telepresence means that the operator receives sufficient information from the remote environment, displayed in a sufficiently natural way, that the operator feels physically present at the remote site [Sherdian 1992b]. For the work reported in this thesis, to leverage humans' natural manipulation abilities, we utilize a hand-based interface that allows an operator to control a robotic hand for remote grasping and manipulation. Forces from the remote manipulator are fed back to the operator's fingertips using a device worn on the hand.

Although force feedback in a telemanipulation system enhances telepresence, it can impose severe restrictions on system stability and performance. Some of the problems inherent to the traditional telemanipulation framework can be addressed with alternate methods of control. In particular, a *shared control* approach combines both high-level and low-level commands from an operator with those of a semi-autonomous robotic hand controller for remote object manipulation. By providing local control, limitations such as master accuracy, bandwidth, and time delays become less detrimental because the opera-

1. The term *haptic* is defined as: relating to or based on the sense of touch. Haptic feedback typically refers to the application of force and/or tactile stimuli to an operator through some mechanical means.

tor's commands are no longer essential for stability at the slave end. However, as the robot assumes more control there is a potential problem with keeping the operator "in the loop." In other words, it may be difficult to maintain a high level of telepresence.

1.2 Shared Control Overview

Control methods for telemanipulation systems fall along a continuum with supervisory control at one end and bilateral (or direct) control at the other and shared control falling the in the middle. In *supervisory control*, the operator controls the remote system entirely by specifying high level commands that are interpreted by a computer. The corresponding low-level commands are then relayed to the remote robot and carried out autonomously. The operator may receive a variety of information about the state of the remote system and its interactions with the environment, but the information is typically in summary form [Sheridan 1992a, Sheridan 1992b].

In contrast, we use the term *bilateral* (or *direct*) telemanipulation to describe a system in which the force and motions of the remote system (the slave) are continuously controlled by the operator and the operator is continuously receiving feedback from the remote robot. The exchange of force and motion information is therefore bilateral. In general, bilateral systems offer a greater sense of telepresence than supervisory systems.

Shared control falls between supervisory control and bilateral control in that the human has the ability to control, and receive feedback from, the remote robot at a low level while maintaining the ability to supply high level commands (see Figure 1-2). Under this control scheme, the human operator can intervene in an autonomous task executed by the robot and the robot can augment the direct commands generated by the operator [Salisbury 1988].

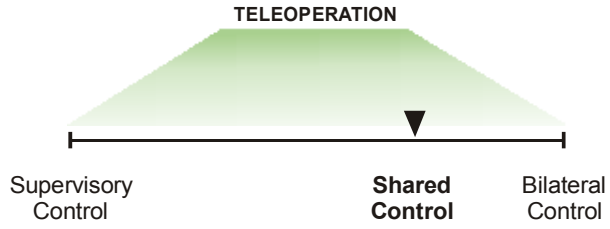

Figure 1-2. Teleoperation control continuum.

This thesis develops a shared control system for dexterous telemanipulation. The telemanipulation system centers around a unencumbering hand-based interface to enable the operator to intuitively command a robotic hand to perform remote grasping and manipulation tasks. The system is built upon a bilateral framework in which measured human hand motions are used to command robot hand motions and force information measured at the robot's fingertips is fed back to the operator's fingertips.

Utilizing force and tactile sensors, the robot hand is capable of autonomous dexterous manipulation. The robot hand can successfully control grasp forces and impart manipulation and rolling motions to grasped objects. In this context, control over grasp forces can be shared between the operator and the robot. There are advantages to having the robot hand take over force regulation if the task is sufficiently well defined. However, human capabilities in spatial reasoning, task planning, and adaptation remain necessary. The incorporation of low-level robot "intelligence" for securely manipulating objects allows the human operator to focus on the task itself, concentrating on the desired motions and behavior of the grasped object or tool.

1.3 Contributions

The major contributions of this thesis are:

- the development of a shared control framework for dexterous telemanipulation. While many parts of this system have been covered in other literature, shared control applied to a system with an unencumbering glove-based interface, fingertip force feedback, and a robot hand with force and tactile sensors has not.
- an investigation of an experimental shared control telemanipulation system. A set of human subject experiments was completed to determine if the addition of a dexterous shared controller to a traditional bilateral system could improve an operator's performance during task execution. The results demonstrate the benefits of shared control and the need to choose carefully the types and methods of direct and indirect feedback. Specifically, it is important to make the human operator aware of robot interventions and to inform the operator of impending state changes in the control.

- the development of a human-to-robot mapping method to support the use of a glove-based interface for intuitive object manipulation with a non-anthropomorphic robot hand. The method captures the intended motions of a virtual object grasped in the operator's hand and uses computed virtual object motions to create commanded motions for the robot. Kinematic and workspace dissimilarities between the human hand and our planar robot hand are compensated for by independently modifying and scaling relevant virtual object parameters.

1.4 Thesis Outline

This thesis is organized into seven chapters. This chapter provides an introduction to, and motivation for, research in shared control for dexterous telemanipulation. Chapter 1 also covers the major contributions of this work.

Chapter 2 discusses previous research in the area of teleoperation and telemanipulation and, in particular, systems with haptic feedback. A brief background of common teleoperation architectures is presented, as well as, a review of the transparency-stability trade-off that must be considered when designing a telemanipulation system. This trade-off provides part of the motivation for seeking other approaches, such as shared control. Relevant investigations in autonomous manipulation and supervisory and shared control schemes are discussed.

Chapter 3 presents our dexterous telemanipulation apparatus. The details of the master and slave components of the system are presented. Additionally, the software framework and overall system architecture are described. The advantages and limitations of the system are discussed and serve to motivate the work in the following chapters.

Chapter 4 develops a procedure for human-to-robot mapping. A review of relevant work is first presented. A solution to the mapping problem for a planar non-anthropomorphic robot hand is presented. The method is based on interpreting human finger motions to deduce motions of a virtual object held between the fingers. The key parameters describing the virtual object motions are then mapped to the robotic hand accounting for kinematic and workspace dissimilarities. Typical mapping results are shown and extensibility of the method is discussed.

Chapter 5 presents a shared control framework for dexterous telemanipulation. The approach is based on combining direct telemanipulation commands from a human operator with those of a semi-autonomous dexterous controller to guide the robot and its hand. The techniques used from autonomous dexterous manipulation theory (e.g., grasp force regulation and accounting for kinematics of contact during rolling motions) are discussed. A framework for cooperative robot fingertip control for object manipulation is presented. The controller utilizes fingertip force and tactile sensors in the robot hand for semi-autonomous control and for providing force information to relay back to the operator. Additional feedback channels, including audio tones and visual indicators, display state information computed by the controller and are designed to aid in keeping the operator "in the loop" as the robot assumes control over some aspects of the task. Methods for determining the intent of the operator during shared control and additional capabilities and benefits of the system are also discussed.

Chapter 6 describes a set of experiments designed to evaluate our implementation of a shared control system for dexterous telemanipulation. The experimental task is described and the various cases to determine the effects of shared control on that task are presented. The results from human subject testing are then presented. An analysis is performed on recorded test data to determine if shared control benefits the operators by improving performance. The performance of each case is evaluated based on each subject's average grasp force applied to an object and on the number of trial failures (object drops). The results from a post-experiment subject questionnaire concerning subjects' expressed preferences are also analyzed and discussed.

Chapter 7 summarizes the work presented in this thesis and the contributions made. Suggestions for future work are also discussed.

2 Background and Relevant Work

The fundamental goal of telemanipulation and teleoperation systems is provide human operators with the ability to interact with environments that would otherwise be difficult to access. The potential benefits of telemanipulation have been espoused for some time and the advantages of such systems are clear. Robotic interfaces can provide access to environments that are hazardous, remote, or require interactions at a smaller or larger scale. For example, teleoperated nuclear waste handling systems can keep operators at a safe distance from contaminated material. For space applications, telemanipulation systems allow for remote control of extra-vehicular activities (e.g., satellite capture and repair), substantially reducing the risk to humans and the costs associated with manned missions.

However, relatively few systems are have progressed beyond a laboratory setting. Many current telemanipulation systems rely heavily on visual feedback and experienced operators. For example, telemanipulators attached to remote-operated undersea vehicles are, at best, position controlled using a miniature replica of the slave arm, but typically rate controlled using joysticks [Bennet and Needles 1997]. Operators must completely rely on graphical and visual information to complete desired tasks.

The quality of a teleoperation experience is often referred to as “telepresence.” Ideally, the information from the remote environment (visual, aural, haptic, etc.) is displayed in such a way that the operator “feels” as if he/she is actually present at the remote environment [Sheridan 1992b]. In other words, the level of telepresence describes how well the master system can emulate the interactions between the slave and the remote environment. Presumably, with a greater level of telepresence, an untrained operator can perform tasks as easily as if he/she was at the remote location. The appropriate level of telepresence required for satisfactory performance is still an area of ongoing research.

Master-slave systems which provide force and tactile feedback have the potential to greatly enhance the level of telepresence. Bilateral force reflecting systems are nothing new. In fact, the first telemanipulation systems, developed around the mid 1940s, provided a direct physical connection with the "remote" environment though mechanical linkages. Shortly after, the linkage connections were replaced with electric servomotors allowing for a much greater distance between master and slave system [Goertz and Thompson 1954].

Research has shown that providing the operator with force feedback can improve task performance (e.g., [Howe and Kontarinis 1992], [Massimino and Sheridan 1994]). Hannaford et al. [1991] showed that during a peg-in-hole assembly task, task completion times, errors, and applied forces were reduced with the addition of force feedback. For these tasks, a force reflecting hand controller was used to control a six degree-of-freedom manipulator with a gripper end-effector.

As noted by Burdea and Zhuang [1991], providing fully immersive finger level force feedback for dexterous telemanipulation is a difficult task. To accurately display forces to a human hand simulating actual manipulation requires a complex master with a large number of actuators capable of applying forces at each finger joint. The device should not prevent normal manipulation motions, complicating the placement of numerous actuators. Several portable master devices capable of providing fingertip level force feedback have been developed. However, most system are limited to one degree-of-freedom force feedback per finger [Burdea 1996].

An evaluation of the CyberGrasp[1] force feedback system for use in dexterous telemanipulation was completed in earlier work [Turner et al. 2000]. The CyberGrasp is a lightweight exo-skeleton device worn on the back of the operators hand (see Chapter 3 for details). Forces are applied to the fingers through nylon cables in sheaths, which are tensioned by small motors worn in a backpack. Since the cable can only pull along a single axis, the forces applied to each individual finger are unidirectional. The experimental results indicated that the system did not necessarily increase speed of performance for some simple manipulation tasks. In fact, the force feedback had a negative effect of task completion time for a knob turning task. These results were most likely due to the single degree-

1. Immersion Corp., San Jose, CA. www.immersion.com

of-freedom in applied force provided by the CyberGrasp system. However, subjects were more likely to cause a trial failure without forces. Comments from the subjects and the experimenters' observations revealed that subjects were more careful if the forces were enabled, indicating that this type of force feedback may be more useful for delicate object manipulation. Additionally, most subjects preferred force feedback for an object pick-and-place task, even though the presence of force feedback did not affect task completion time [Turner 2001].

While the addition of force feedback can improve the level of telepresence in a telemanipulation system, several issues are introduced by the inclusion of the operator in the overall feedback loop. Suppose an operator is commanding the position of a slave arm or finger. If forces are feedback to the operator based on the slave motions and/or interactions between the slave and the environment, the applied forces can now affect the operator's position. Thus, the operator is becomes dynamically coupled to the master-slave system. This dynamic coupling means that the telemanipulation system is subject to the same constraints of typical feedback control systems. Mechanical compliance, time delays, model inaccuracies, and unaccounted for non-linearities can all limit the achievable performance and affect system stability.

The research presented in this thesis utilizes a force feedback telemanipulation testbed. While force feedback can enhance the telemanipulation experience, it is important to point out some of the fundamental issues encountered when developing a hapticly enabled telemanipulation system. The following sections provide a brief background of common teleoperation architectures and a review of the extensive previous work covering the stability-transparency trade-off inherent to bilateral systems. Additionally, practical methods for improving system telepresence are discussed. Next, alternate methods for improving telemanipulation performance are introduced. The methods of supervisory and shared control for dexterous telemanipulation are presented and relevant work is discussed.

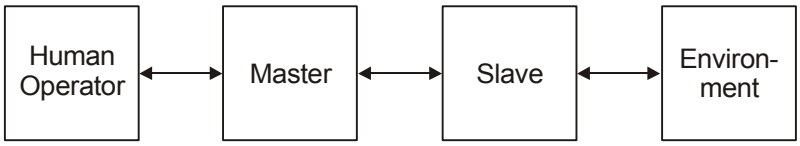

Figure 2-1. Basic blocks of the teleoperation architecture in which the human operator interacts with the environment through the master-slave system.

2.1 Bilateral Teleoperation

2.1.1 Teleoperation Architectures

The basic blocks of a teleoperation system consist of a human operator interacting with an environment through a teleoperated system (see Figure 2-1). The teleoperated system generally consists of a master and a slave with communication link between them. Master devices may range from a one degree-of-freedom joystick to immersive glove-based interfaces. Similarly, the slave device may range from a one degree-of-freedom "manipulator" to a complex system with a dexterous robot hand attached to a multi degree-of-freedom arm. Both sides of the telemanipulation system typical have some type of local control operating on position, velocity, and/or force. The master and slave systems may be controlled on the same computer or separated by hundreds of miles. By making some simplifying assumptions, a model describing the teleoperation system can be developed and analyzed for performance and stability using standard control techniques and electrical system equivalents.

The most common type of teleoperation architecture is one in which the master system sends position (or velocity) commands to a slave system and force information from environmental interaction is fed back to the master system from the slave system. This type of two-channel (one communication link in each direction) architecture is often referred to as a "position-force" architecture. However, in general both positions (or velocities) and forces can be transmitted between the master and slave [Lawrence 1993][2].

2. In general, teleoperation models are developed in terms of velocities and forces instead of positions and forces because of the natural force-velocity power relationship. Stability and transparency analyses are unaffected by this representation. However, implementation of a velocity based control can lead to offsets between the slave and master position, thus, in practice, position information is commonly communicated.

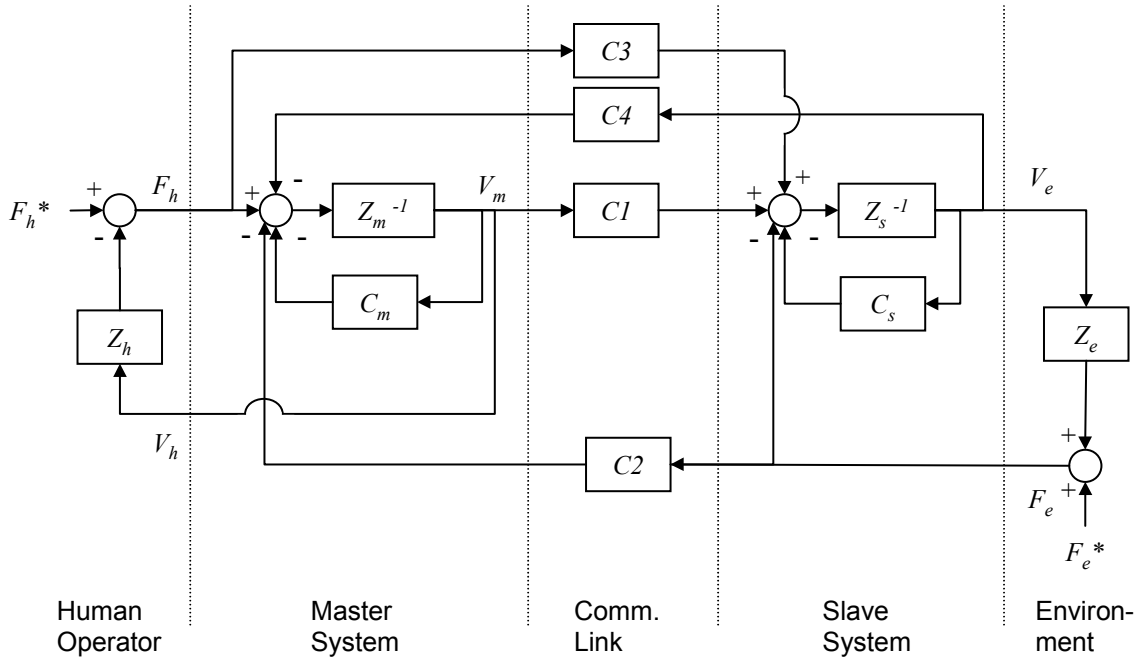

Figure 2-2. Block diagram representing the general four-channel one degree-of-freedom teleoperation architecture with velocity and force information shared between the master and slave system (adapted from [Lawrence 1993]).

Figure 2-2 shows a block diagram of a four-channel one degree-of-freedom teleoperation system with master, slave, and communication link models, as well as, operator and environment models. To simplify the stability and transparency analysis, the system behavior is assumed to be linear, e.g., contact with an environment is maintained. By linearizing the system, the system blocks can be represented in the frequency domain using the Laplace transform.

The "Z" terms represent the impedances of the various sub-systems shown. The impedance relates the velocity to force for a given system:

$$F_i(s) = Z_i(s) \cdot V_i(s) \tag{2.1}$$

The impedance of the operator's arm, hand, or finger is simplified with a linear model such as:

$$Z_h(s) = m_h \cdot s + b_h + \frac{k_h}{s} \tag{2.2}$$

The master and slave impedances are typically modeled as masses (e.g., $Z_m(s) = m_m \cdot s$). The environment may be modeled as a simple spring stiffness ($Z_e(s) = k_e / s$), but can be modeled more generally as spring-mass-damper system (as if the slave manipulator was grasping an object with mass, stiffness, and damping properties that is in constant contact with a infinitely stiff environment).

Additionally, compensators, $C_i(s)$, are typically included in a local feedback loop for the master and slave systems. The communication blocks, (e.g., $C1(s)$), depend by the type of architecture and can include filtering blocks, modeled delays, and compensators.

The four-channel architecture describes a general bilateral framework and the system blocks can be defined to easily represent simpler architectures. As an example, for a two-channel "position-force" architecture, we define the following:

$$C1(s) = C_s(s) = b_s + \frac{k_s}{s} \tag{2.3}$$

$$C2(s) = k_f \tag{2.4}$$

$$C3(s) = 0, C4(s) = 0 \tag{2.5}$$

where k_f represents a gain applied to the measured slave force, also commonly referred to as the *force-reflection ratio*. Additionally, if we assume that the master controller can only

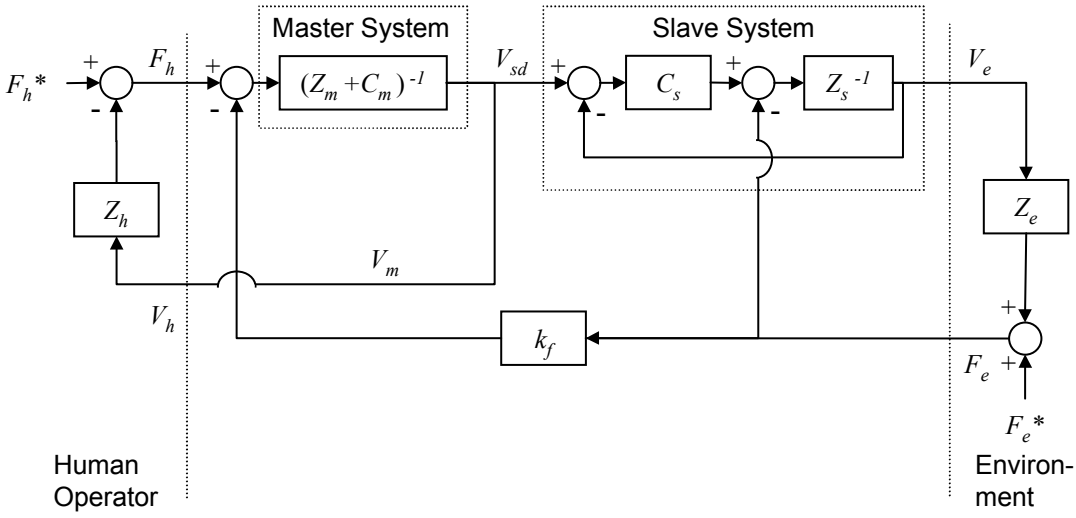

Figure 2-3. Traditional "position-force" two-channel architecture for bilateral teleoperation. The master system is defined by the master impedance and additional damping. The slave system is shown in the typical compensator-plant form with unity feedback and disturbance force input.

apply velocity damping (i.e., $C_m(s) = b_m$), we can reformulate the block diagram into a somewhat more tractable form, shown in Figure 2-3.

2.1.1.1 Evaluating Telepresence

Utilizing the generalized model for a one degree-of-freedom teleoperation system, one can begin to evaluate the "feel" of the teleoperation system and the stability of the system. Each of the sub-systems in Figure 2-2 can be represented as a two-port network, where velocity flows into and out of the ports and force is measured across the ports (Figure 2-4a). Combining the master, communication link, and slave system into one block results in a simplified two port network (Figure 2-4b).

A common method for quantitatively describing the "feel" of a teleoperation system is to evaluate the impedance felt by the operator at the master device interface. This impedance represents the impedance of the remote environment as "viewed" through the master-slave telemanipulation system and is referred to as the transmitted impedance, Z_t (see Figure 2-4c).

While the transmitted impedance, Z_t, provides an analytical description of the "feel" of a system, it is not always clear what this term should equal to create the ideal tele-

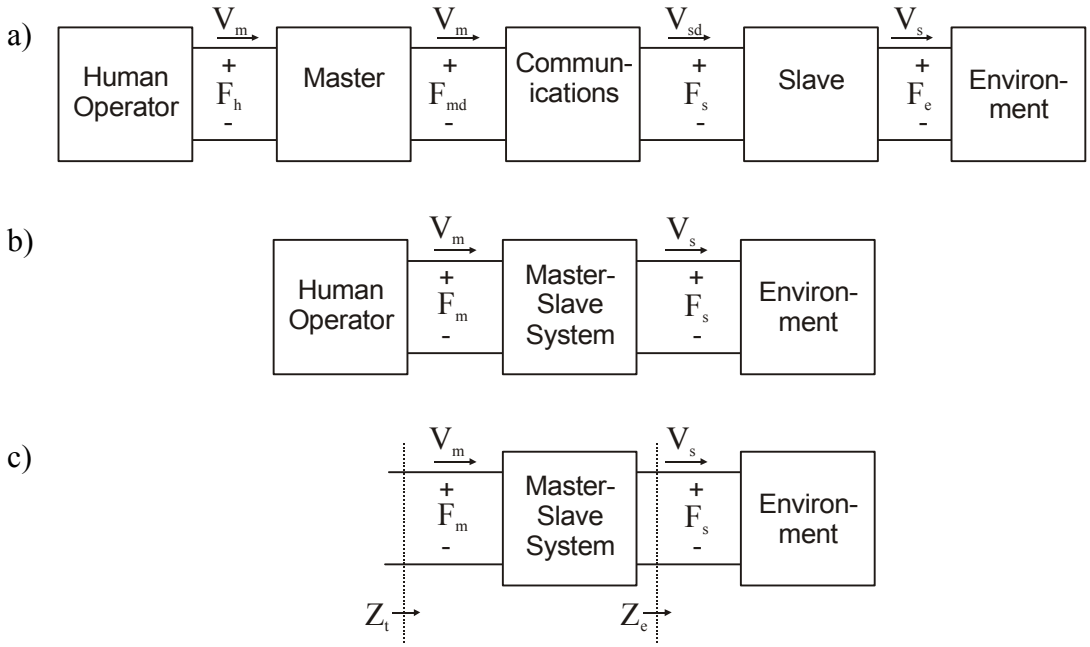

Figure 2-4. Two port network representation of a general bilateral teleoperation system. The blocks represent generalized impedances for each of the sub-systems. The impedance the operator feels is denoted by Z_t, the transmitted impedance of the environment through the master-slave system (adapted from [Anderson and Spong 1989]).

operation system. Some consider the ideal telemanipulation system to be one in which the transmitted impedance is equal to the environmental impedance, that is:

$$Z_t = Z_e \tag{2.6}$$

In other words, the operator will feel as if he or she is directly connected to the environment virtually though a massless rigid bar [Yokokohji and Yoshikawa 1994]. This is referred to as perfect transparency. However, it may not always be desirable to display exactly what the slave encounters during environmental interaction. In some cases it may make more sense to define the system performance based on the desired task and/or the feedback required by the operator for successful task completion [Raju et al. 1989]. For example, systems designed for teleoperation in environments with heavy loads, the interaction forces fed back to the operator are scaled down to an appropriate level.

Stability is also an important aspect in creating a teleoperation system with a high level of telepresence. Clearly, if a system exhibits unstable or nearly unstable behavior, the illusion for the operator of being virtually present at the remote sight can be destroyed, in addition to possibly making the task difficult or impossible to perform.

For teleoperation applications in which the remote site is truly remote, time delays are the primary cause of stability problems. At some point, time delay induced instability requires the system to be controlled “open-loop”, reducing the operator to a “wait and see” approach. For these situations, the general teleoperation architecture no longer applies. Supervisory control, originally developed by [Ferrell and Sheridan 1967] was introduced as a possible solution for teleoperation with large time delay. Supervisory control implies that an operator commands and monitors the actions of an autonomous remote slave. However, as discussed in the following sections, a purely supervisory system can limit the level of telepresence as the operator is no longer dynamically coupled to the system.

Moderate to small time delays can also cause a system to be unstable or nearly unstable. Nearly unstable systems can exhibit a “bounce” or “chatter” when the operator brings the slave manipulator in contact with a stiff environment. The “bounce” is at best irritating for the operator and, at worst, can possibly damage the slave or harm the operator. As Lawrence [1993] points out, this is partially due the non-linearities of contact, but one

of the main causes of instability is poorly damped dynamics while in contact (which can be modeled using system descriptions like Figure 2-2). Because the operator is not necessarily a passive impedance, he/she can help to stabilize the system by increasing arm or grasp stiffness by muscle co-contraction. However, requiring the operator to constantly make adjustments for system stability can cause operator fatigue and diminish the sense of telepresence.

In practice, creating a perfectly transparent and stable teleoperation system is virtually impossible. Real world artifacts such as time delays, sensor dynamics, and model approximations require a trade-off between stability and transparency. Over the past fifteen years, a considerable amount of research effort has focused on this fundamental trade-off, resulting in many different approaches attempting to define the optimal system. The literature covers a spectrum of approaches from guaranteed passive systems to four-channel transparency optimized systems, with traditional "position-force" and "position-position" architectures falling in the middle of the continuum [Cavusoglu et al. 2001]. The following section reviews some of different approaches to addressing the stability-transparency trade-off.

2.1.1.2 Stability and Transparency

At the far end of the spectrum, Anderson and Spong [1989] focused on the stability of teleoperation systems. Using concepts from networks and scattering theory, the authors show it is possible to design a stable bilateral teleoperator system in the presence of large time delays and demonstrated the method for time delays as large as two seconds. Working with the expanded two-port network representation (Figure 2-4a), the master and slave control laws are defined such that the communication block's behavior is identical to a two port lossless transmission line (which is inherently passive). If all other sub-systems are assumed to be passive or passively controlled, the telemanipulation system is guaranteed to be stable. However, the transparency of the system was not addressed. Lawrence [1993] suggested that even though the stability is guaranteed, this approach typically results in relatively poor transparency performance compared to more traditional "position-force" architectures. Lawn and Hannaford [1993] tested different control methods for a common set of telemanipulation tasks. Their results showed that passivity based algorithms had

approximately a 50% increase in total completion time, as compared to "position-position" and "position-force" algorithms.

In a similar approach to [Anderson and Spong 1989], Niemeyer and Slotine [1997] present a passivity based method for teleoperation and introduce the notion of wave variables. Wave variables represent a physical description of the passivity approach and allow the designer to easily address problems such as wave reflections, which can cause undesirable oscillatory behavior. Additionally, if both master and slave are similar and under velocity control, a "good sense" of telepresence can be achieved. However, as master and slave diverge in similarity, the level of telepresence can be more difficult to maintain.

Lawrence [1993] presented one of the first comprehensive papers to explore the trade-off between transparency and stability in bilateral systems. Starting with the network formulation, the necessary conditions for perfect transparency were derived. Using the two-port master-slave representation (Figure 2-4b), a hybrid matrix can be formed [Hannaford 1989] and relates the input velocity and force to the output velocity and force:

$$\begin{bmatrix} F_h(s) \\ V_h(s) \end{bmatrix} = \begin{bmatrix} h_{11}(s) & h_{12}(s) \\ h_{21}(s) & h_{22}(s) \end{bmatrix} \cdot \begin{bmatrix} V_e(s) \\ -F_e(s) \end{bmatrix} \tag{2.7}$$

Using the impedance relationships, $F_h(s) = Z_t(s) \cdot V_h(s)$ and $F_e(s) = Z_e(s) \cdot V_e(s)$, Lawrence specified the required hybrid parameter values for perfect transparency ($Z_t(s) = Z_e(s)$):

$$h_{11}(s) = h_{22}(s) = 0 \tag{2.8}$$

$$h_{12}(s) = -h_{21}(s) \tag{2.9}$$

Using the general four-channel block diagram, the hybrid parameters are expressed in terms of the transfer functions representing the impedances, compensators, and communication links. The requirement for perfect transparency results in the following expressions for the communication blocks:

$$C1(s) = Z_s(s) + C_s(s) \tag{2.10}$$

$$C2(s) = C3(s)^{-1} \tag{2.11}$$

$$C4(s) = -(Z_m(s) + C_m(s)) \tag{2.12}$$

This result is significant, in that, to create a perfectly transparent system all four channels are necessary. Furthermore, because of the impedance terms for the master and slave have inertial components, acceleration measurements are also required. However, with the introduction of time delays, prefect transparency is not possible and stability begins to degrade. Based on this, Lawrence then develops a "transparency optimized" four-channel system that is robust to time delays by adding communication link filters using passivity techniques and neglects the inertial terms to eliminate acceleration measurements. A performance comparison is made and shows that the "transparency optimized" system has greater transparency and stability than traditional two-channel architectures. However, the system is not guaranteed passive, reinforcing the notion that there is a trade-off between transparency and stability in practical implementations of teleoperation systems.

Yokokohji and Yoshikawa [1994] focused on developing a system with perfect transparency and showed that in the absence of time delay a system could also be passive. Similar to Lawrence [1993], the necessary conditions for perfect transparency and resulting control laws were presented. To implement such a system, the master and slave parameters must be known exactly and position, velocity, and acceleration measurements for both must be available without time delay. Using a well characterized one degree-of-freedom experimental set-up with the necessary sensors, a high degree of transparency was achieved. However, because the plant models were not known exactly, the dynamics could not be fully cancelled without causing instability, thus reducing the transparency.

Hashtrudi-Zaad and Salcudean [2002] investigated the stability and performance benefits with the addition of local force feedback at the master and slave. Using the formalism developed by Lawrence [1993], the authors showed that perfect transparency can be achieved with only three channels. Additionally, they show high levels of stability can be achieved without a decrease in transparency as compared to the conventional four-channel architecture. However, in the presence of time delays, transparency must be compromised to maintain stability.

Numerous other studies have investigated the stability-transparency trade-off though simulation and experimental comparison [Hannaford 1989, Lawn and Hannaford

1993, Eusebi and van der Ham 1994, Sherman et al. 2000, Cavusoglu et al. 2001, Arcara and Melchiorri 2002].

2.1.2 Two-Channel Approaches

Given the complexity and hardware requirements of four-channel bilateral systems, a significant amount of research has also focused on optimizing two-channel architectures. The two most common architectures are "position-force" and "position-position." As discussed previously, a "position-force" architecture is one in which a desired position (or velocity) command is sent to the slave based on the master position (or velocity) and the measured force on the slave side is fed back to the force controlled master (see Figure 2-3).

In a "position-position" architecture both master and slave are commanded based on the position error between the two. Using Lawrence's formalism and similar linear models, the communication blocks have the following transfer functions:

$$C1(s) = C_s(s) = b_s + \frac{k_s}{s} \tag{2.13}$$

$$C2(s) = 0, C3(s) = 0 \tag{2.14}$$

$$C4(s) = -C_m(s) = -\left(b_m + \frac{k_m}{s}\right) \tag{2.15}$$

To gain a better idea of the behavior of the "position-position" architecture we can slightly modify the architecture as follows. If the controllers for the slave and master are identical (i.e., $C_m(s) = C_s(s)$) and we compute the control force on the slave side, this "position-position" formulation can now be thought of as a quasi "position-force" arrangement in which the force fed back to the master is the impedance force generated by the slave controller [Niemeyer 1996]. However, by feeding back the impedance force, the movements at the master side may seem "sluggish," especially during free-space motions [Das et al. 1992].

For manipulation tasks, such as peg-in-hole and pick-and-place, the "position-force" architecture has been widely used (e.g., [Hannaford 1991, Lawn and Hannaford 1993, Massimino and Sheridan 1994, Daniel and McAree 1998]). Because the measured force is fed back to the master, the operator typically receives a "crisp" sense of contact

which can aid in task completion. However, the "position-force" architecture is also prone to stability problems with small to moderate time delays [Lawn and Hannaford 1993]. Even in cases without time delay, significant differences in master and slave masses can cause stability problems [Daniel and McAree 1998]. By feeding back only the measured force, the "position-force" architecture effectively "hides" the mass of the slave system from the operator (eliminating the "sluggish" behavior seen in the "position-position" architecture). However, if a one-to-one kinematic and force correspondence is desired, the dynamics of environmental impacts can not be easily cancelled out by the controller. Often, to address the stability problems, the overall loop gain is reduced (typically by reducing the magnitude of the force fed back). Of course, as the stability of the system increases, the transparency is also reduced. A common observation of force-reflecting systems is that the interactions with the environment tend to feel "mushy" or "spongy" to the operator [Lawrence 1993].

To improve performance of two-channel "position-force" bilateral systems, several methods have been proposed. For example, Colgate [1993] presents an impedance shaping framework in which the measured forces are combined with a shaping term derived using power scaling techniques and an *a priori* model of the task impedance. Consequently, transparency robustness (ability to provide good transparency even with uncertainty in environment and operator dynamics) and transparency were improved compared to traditional "position-force" implementations.

Fite et al. [2001] develop a two-channel design approach based on a "frequency-domain loop-shaping" perspective instead of the traditional hybrid network perspective. Casting the problem in the frequency domain enables the use of classical controls compensation techniques. The authors contest the idea the that stability and transparency are conflicting objectives. Applying a loop shaping compensator in simulation, the authors show it is possible to simultaneously improve transparency in terms of extending the bandwidth and increasing stability robustness by maintaining appropriate gain and phase margins. The system also incorporates local force feedback on both the master and slave to help account for uncertainties in the operator and environment dynamics. In an implementation of this method, the compensators were designed based on experimental determination of the transparency transfer functions [Speich and Goldfarb 2002]. The authors note that in order to

achieve an increase in stability without sacrificing transparency, the stability crossover frequencies must not be within or near the desired transparency bandwidth.

In cases where the master and slave have significantly different masses, the notion of perfect transparency may not be a useful framework for bilateral teleoperation system design. For systems with negligible time delay, Daniel and McAree [1998] suggest that the "physics of teleoperation" and the implications that this physics has on the fidelity of the forces fed back to the operator are perhaps more appropriate when designing a "position-force" system. Using a simple conservation-of-momentum argument for a perfect force reflecting system with one-to-one position correspondence, the authors show that the master-slave mass ratio (m_m / m_s) places a fundamental limitation on system performance. In other words, if a master and slave have equal velocities before an environmental impact (kinematic correspondence), the resulting rebound velocities will be much higher for the master if the slave is more massive (assuming prefect transmission of the impulse, i.e., perfect force reflection). If the operator can not absorb this energy, he/she may not be able to control the contact task. This type of "violent recoil" behavior of contact instability is commonly seen in "position-force" systems unless the force-reflection ratio is decreased. Again, we see the trade-off between stability and transparency. In developing a solution to this problem, Daniel and McAree focus on the needs of the operator in terms of the feedback required to provide a high level of telepresence, termed the "psychophysics" of teleoperation. The authors suggest that the force information can be broken up into two channels: an "energetic" or low frequency bilateral channel satisfying the proprioceptive needs of the operator, and an "information" unilateral channel feeding back high frequency force information satisfying the operator's tactile needs. The magnitude of the force-reflection ratio is limited by the master-slave mass ratio and desired slave stiffness. However, by breaking up the force signal, a notch filter can be applied to maximize the low frequency gain within stability limits but then still include the high frequency "information" force channel. It is important to note that the master system must have a high enough bandwidth to support the high frequency information channel.

2.1.3 Telemanipulation

While much work has been done in the area of improving stability and transparency in bilateral system, the research has mainly been confined to simulation or relatively simple teleoperation systems. Experimental validations of enhanced bilateral architectures or architecture comparisons have typically been implemented on one degree-of-freedom systems (e.g., [Anderson and Spong 1989, Yokokohji and Yoshikawa 1994, Lawn and Hannaford 1993]) or kinematically similar systems (e.g., [Speich and Goldfarb 2002, Sherman et al. 2000]). Additionally, most of these master systems have the operator grasping a tool or handle to complete teleoperation tasks (typically tapping on a wall) and therefore are operated with whole arm or hand motions.

The lessons learned from the investigations in to the stability and transparency are directly relevant to the development of a tele-*manipulation* system. Without a basic understanding the design trade-offs, achieving good performance is at best *ad hoc*. As we will see in the discussion of our telemanipulation set-up in Chapter 3, many of complex optimized architectures are not readily transferable to our system because of the unique aspects of our intuitive interface and simplified slave system. Our hand-based telemanipulation system with master-slave kinematic dissimilarities does not necessarily conform to the general four-channel architecture. The specification of desired levels of transparency and stability for these types of systems remains a difficult problem [Lawrence 1993]. The focus this work was not to create a guaranteed stable system or a perfectly transparent one, but rather to develop a telemanipulation system with good bilateral performance that serves as a test-bed for investigating alternate methods of control for telemanipulation.

2.2 Alternative Approaches

Other control frameworks have the potential to improve performance in telemanipulation systems. Incorporation of additional "intelligence" at the master and slave ends can help to overcome some of the limitations found in traditional bilateral systems. Problems such as time delay can be addressed by "breaking" the telemanipulation loop (reducing teleoperation to a "wait-and-see" approach) or modifying the information exchanged between the master and slave. Other issues, such as limited fidelity in the master interface can be compensated by having the slave robot feed back information utilizing other indirect methods.

The following sections discuss alternative telemanipulation control methods and the potential benefits.

2.2.1 Control Frameworks

There are two main approaches to developing an advanced control framework for telemanipulation: supervisory control and shared control. In this context, supervisory control implies that a human is supervising a teleoperated system. In the strictest definition, supervisory control is a hierarchical framework in which the human operator is limited to providing high level commands that are interpreted by a computer. Once enabled by the operator, the computer or tele-robot then executes the commands autonomously at the remote site utilizing sensors and feedback loops (see Figure 2-5 below). In less strict forms, supervisory control allows for a more continuous interaction between the human supervisor and the autonomous slave [Sheridan 1992b].

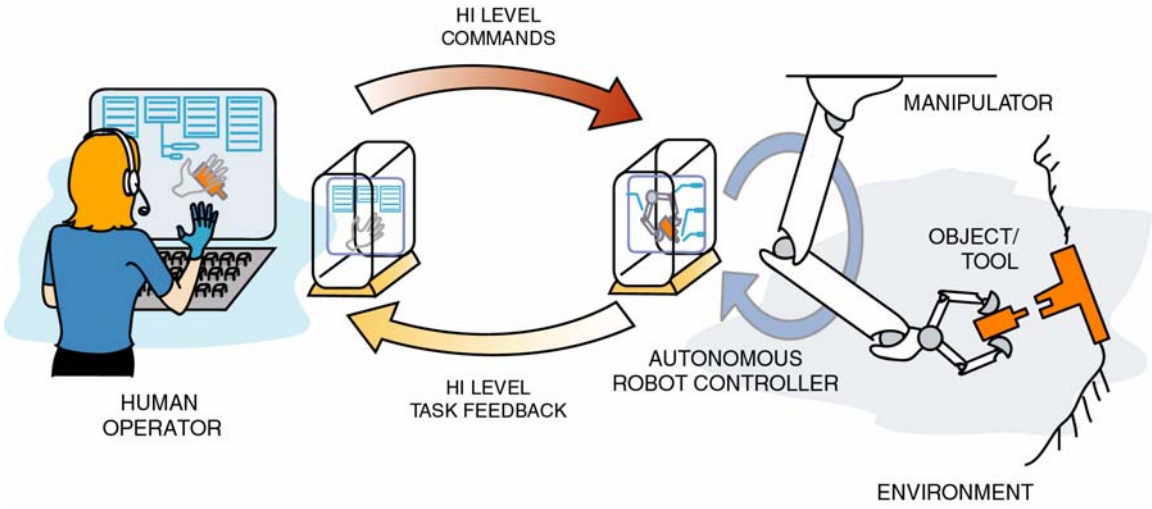

Figure 2-5. The supervisory control concept for dexterous telemanipulation in an assembly task. High level task commands are programmed by the human operator and sent to the remote robot controller for autonomous execution. The autonomous controller relies on local sensor information and feedback loops for task execution. The operator can monitor the task progress with feedback form the remote controller.

Utilizing the high level of interaction, operators can perform many functions such as planning, teaching, and monitoring of tasks to be completed in the remote environment. Being freed of low level control tasks, the operator's cognitive load can be reduced. However, factors such as task complexity and environmental uncertainty can heavily influence the required sophistication of autonomous algorithms and human command interpretation.

Furthermore, because the feedback loop limits the operator to monitoring the autonomous actions, a sense of telepresence can be lost in comparison to direct teleoperated systems.

Shared control represents a middle ground between supervisory control and traditional bilateral control, in that the human has the ability to control, and receive feedback from, the remote teleoperator at a low level (e.g., at the servo level) while maintaining the ability to supply high level commands (see Figure 2-6). In this way, the human can intervene in the autonomous task executed by the robot and, conversely, the robot can augment direct commands issued by the operator [Salisbury 1988]. Shared control seeks to capitalizes on the two level hierarchy by supplying the operator with a greater sense of telepresence over supervisory systems while overcoming limitations associated with direct telemanipulation systems. The shared control framework also reduces the complexity of the slave as compared to a purely supervised system, in terms of algorithms for autonomy and possible sensor requirements (e.g., machine vision).

Sharing of control can occur at the task level down to the servo level. In task level sharing, the operator and the robot's commands may be merged to complete a given task

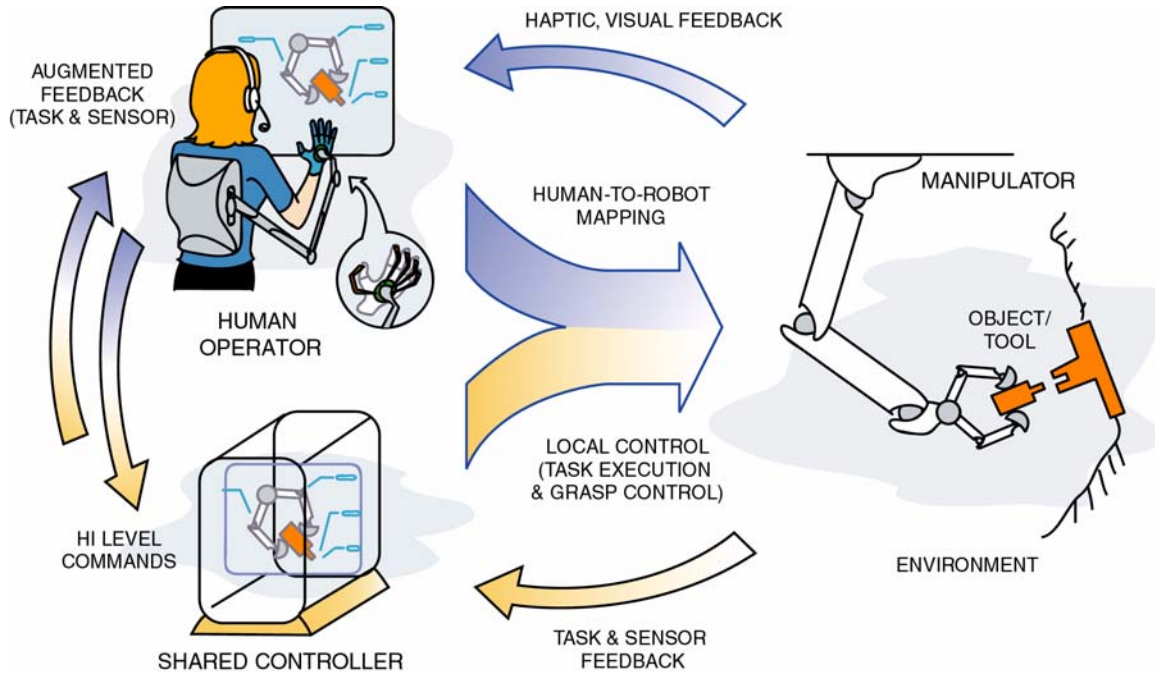

Figure 2-6. The shared control concept for dexterous telemanipulation in an assembly task. The essence of shared control is in the combination of operator commands, both high level and low level, with the commands from a semi-autonomous controller. Haptic, tactile, and visual information is fed back to the operator from the remote manipulator. Additionally, augmented feedback of direct quantities (e.g., force) and indirect quantities (e.g., grasp stability) can be supplied to the operator from the shared controller.

[Hayati and Venkataraman 1989]. For example, the operator may modify autonomously planned task trajectories to account for un-modeled environmental factors. Conversely, the slave system may need to modify the operator's commanded trajectory based on local sensor feedback. At a lower level, the operator may control specific motion directions while the robot controls the reciprocal force directions in a hybrid control scheme [Hannaford et al. 1991]. Additionally, the operator and robot may share control over a particular variable. Either the operator or the robot may have primary control but the other can intervene if necessary. For example, suppose an operator is grasping an object and regulating the applied internal force. If the operator reduces the applied force, the robot may take over to prevent dropping the object during a critical phase of the task.

The shared control framework can also provide benefits for teleoperation systems with small to moderate communication time delays. For example, in a ground-controlled satellite tele-maintenance system, an on-board robot controller can use local sensing to compensate for trajectory errors through compliance control and force limiting [Oda et al. 1999]. However, with larger time delays, a shared control system (especially one with haptic feedback) becomes a less practical solution due to stability concerns.

In many cases of traditional bilateral telemanipulation, the master display device may not be able to recreate fully the interactions with the remote environment. Factors such as fidelity and bandwidth limitations of a haptic device can reduce the immersive experience for the operator. Shared control can help to overcome some these issues by utilizing other forms of feedback. For example, suppose a force feedback system lacks the ability to display high frequency slip information for a delicate manipulation task. The shared controller could monitor the slip locally and display the information through a less direct method, such as audio tones or prerecorded vibration patterns sent to a vibrotactile display.

2.2.2 Previous Work

The following section briefly describes the origins of supervisory control and some current examples. A short review of advances in dexterous manipulation required for semi-autonomous slave manipulation is then presented. Next, many of the key works in shared control for telemanipulation and dexterous telemanipulation are discussed.

2.2.2.1 Supervisory Control

Although the concept of a supervisory controller for telemanipulation is a long established idea, examples of "pure" supervisory systems are rare. The concept of supervisory control was initially introduced by Ferrell and Sheridan [1967]. Motivated by the emergence of space flight, researchers were investigating the possibility of remote manipulation and control over vast distances, e.g., controlling a tele-robot on the surface of the moon from earth. The restrictions imposed by speed-of-light transmission time delays would limit the control of any typical teleoperation system to a "wait-and-see" approach or essentially an open-loop approach. Thus the authors advocated a human supervised remote manipulator with a local feedback loop that is not subject to the effects of time delay. Under supervisory control, the human operator can plan and can teach the remote system The operator can also monitor the automatic execution of the desired tasks and halt action if necessary. Additionally, the operator can give commands at the object or task level, removing the need for continuous human control thus reducing the operator's cognitive load [Sheridan 1992b].

One notable example of a strictly supervisory telemanipulation system is the approach taken by Cannon and Thomas [1997]. In their system, a human defines tasks for a remote manipulator by applying virtual tools that are overlaid on video of the remote environment. This system allowed operators to define high level directives at the object level, such as "put that there," using a instrumented glove to guide a virtual grasper. After a sequence of tasks were specified, the operator commanded the robot to plan the necessary task trajectories and end-effector motions and then commence execution.

Despite the obvious benefits of the supervisory control framework for applications with large time delay, there are several drawbacks. Because the operator is issuing commands in high level "natural" language, it is necessary at some point to translate the commands to specific robot operations. Unless the number of commands is significantly limited, a complex menu hierarchy built upon task primitives is usually necessary to ascertain the operator's intent [Backes and Tso 1990, Koide et al. 1993, Aigner and McCarragher 2000].

2.2.2.2 Dexterous Manipulation

For a telemanipulation system, the addition of a robotic hand end-effector with multiple fingers acting independently or cooperatively poses an additional challenge. If a desired task requires object manipulation (controlled autonomously or shared), the programming of the remote robot must draw upon dexterous robotic manipulation, an area which has received a considerable amount of research effort.

Many algorithms now exist for stably manipulating objects in a multi-fingered hand with point or soft rolling contacts. Over the last 20 years, considerable progress has been made and some important examples include [Kerr and Roth 1986, Buss et al. 1996, Han et al. 1999]. Algorithms for manipulation with sliding have also been explored, but the state-of-the-art is less well developed [Brock 1988, Kao and Cutkosky 1993, Howe and Cutkosky 1996]. The problems of grasp choice, grasp planning, and regrasping for autonomous manipulation have also been studied extensively (e.g., [Leveroni et al. 1998, Buss and Schlegl 2000]). For an overview of the fundamentals of dexterous manipulation and an extensive list of references see Okamura et al. [2000].

For finite motions, the manipulation algorithms require information about contact locations and contact conditions; otherwise, inaccurate motions and unstable grasps will result as slip and creep inevitably occur. In some cases, this information can be provided visually, but in the general case, robots, like humans, require tactile sensing. As a quick illustration, consider how clumsy we feel when doing tasks like fastening buttons or working with small tools in freezing weather. The cold quickly numbs our fingertips. Our muscles, being located primarily in our forearms and covered by coat sleeves, are much less affected; but the loss of haptic sensitivity reduces our dexterity.

In robots, contact type and location information can be provided via tactile sensors and by "intrinsic tactile sensing" [Mason and Salisbury 1985, Bicchi et al. 1993], and extrinsic sensing. A wide variety of tactile sensors have been developed, including arrays, analog position sensors, and dynamic tactile sensors (e.g., [Fearing 1990, Howe and Cutkosky 1993]). Still, the accuracies and update rates available from tactile sensors are typically poorer than those of conventional joint angle and torque sensors used in servo control [Son et al. 1995]. Perhaps as a consequence, the use of tactile information in dexterous

manipulation is still in an early stage [Howe 1992, Maekawa et al. 1992, Boshra and Zhang 2000].

Despite many advances in manipulation with rolling and sliding, grasp choice, and tactile sensing, the application of dexterous hands remains confined to simplified tasks in a laboratory setting. For short periods of time, robot hands can successfully control grasp forces and impart rolling motions to grasped objects. Over the longer term, human capabilities in spatial reasoning, tactile sensing, and the incorporation of tactile information in grasp and motion planning remain necessary.

There is, however, an opportunity to take advantage of short-term manipulation capabilities of a robot hand to enhance the overall dexterity and ease of use of a telemanipulation system. Supervisory control provides the framework for drawing upon short duration autonomy for task monitoring and execution. A shared control telemanipulation system incorporates the high level control and remote autonomy of supervised systems with the telepresence found in direct control and feedback of master-slave telemanipulation systems. The incorporation of low level "intelligence" for securely manipulating objects can reduce the demands on an immersive telemanipulation system. This is an important consideration because time delays, friction, limited servo bandwidth, and limited sensor resolution all conspire to make high-fidelity force and touch feedback difficult to achieve.

2.2.2.3 Shared Control for Dexterous Telemanipulation

In cases where time delay is not a significant problem, shared control offers advantages over strict, hierarchical, supervisory control frameworks; thus, many previous investigations have implemented a mix of hierarchical and shared control for telemanipulation. As Figure 2-7 illustrates, control methods for telemanipulation fall along a continuum from pure hierarchical supervisory control to direct bilateral control, with shared control in the middle. These approaches can also be evaluated in terms of the time delay the telemanipulation loop can tolerate. Systems with little or no time delay are typically dynamically coupled; i.e., strongly coupled systems have the operator controlling position and receiving force feedback based on the robot's interaction with the remote environment. Because of this tightly closed control loop involving the human operator, relatively small time delays can cause stability problems if the desired transparency of the system is high (in other

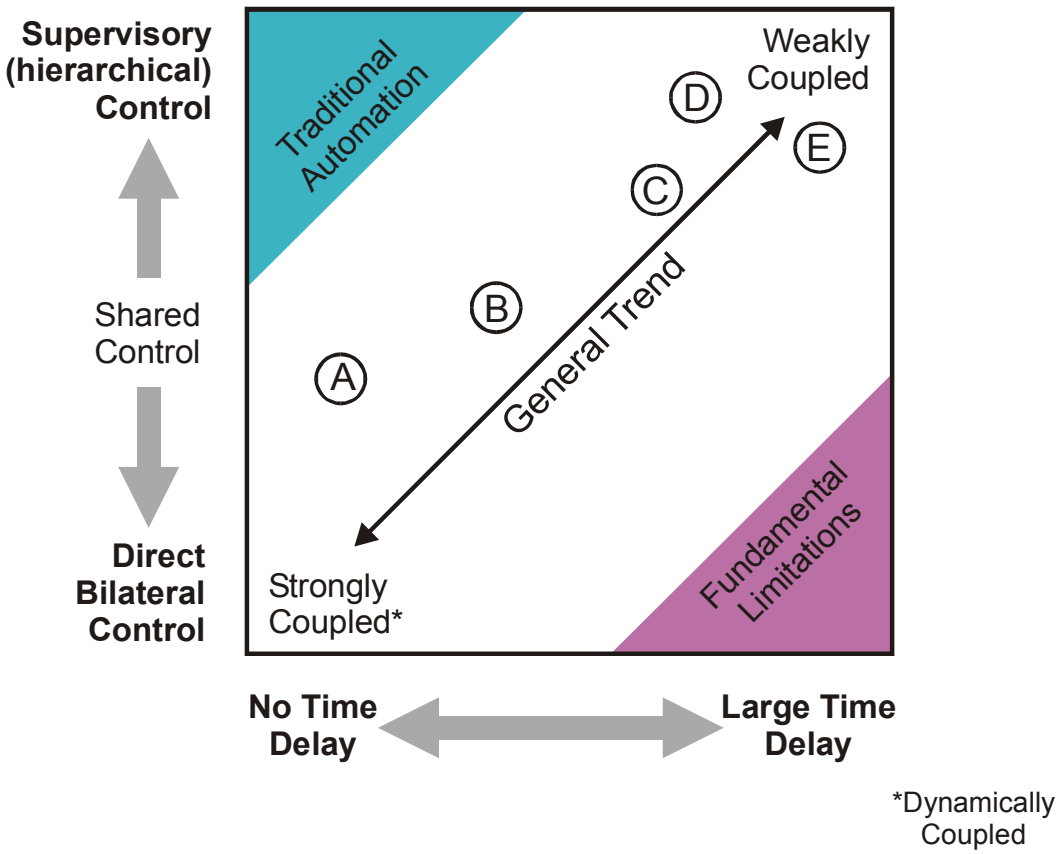

Figure 2-7. The continuum of shared control telemanipulation. The shared control framework represents a middle ground for telemanipulation system configurations, falling between bilateral systems (with a direct connection of force and motion between the operator and the remote robot) and supervisory control systems (wherein operators specify commands to be completed autonomously by the remote robot). Another important aspect of the continuum is the amount of time delay the system can tolerate. The operator and remote environment in traditional bilateral systems are strongly coupled (dynamically). However, as time delay increases the operator becomes more removed from the remote robot control. The general trend of telemanipulation research and development is shown. Example experimental systems reviewed in this section are indicated with letters A-E, corresponding to the following systems: (A) the system described in this thesis, (B) NASA's Robonaut system [Ambrose et al. 2000], (C) Michelman and Allen's [1994] system, (D) Cannon and Thomas's [1997] supervisory system, and (E) the DLR's robotic hand system [Brunner et al. 1994].

words, if it is able to faithfully reproduce the environmental stiffness to the operator). Fundamental limitations due to stability concerns constrain direct bilateral systems to applications with relatively small time delays. Systems designed for handling a larger amount of time delay typically have a weaker coupling between the operator and remote robot; thus, the actions of the robot become more autonomous. The general trend of telemanipulation systems spans from closely coupled bilateral systems to systems capable of handling large time delays but limits the operator to more supervisory roles.

Initially researchers of shared control focused on developing a framework for task-level sharing of motion trajectories for systems with time delay [Hayati and Venkataraman

1989, Oda et al. 1999]. Others focused on modifying the impedance of slave-manipulators based on teleoperator commands and local sensor information [Hannaford et al. 1991, Backes 1992]. While many systems provide force feedback to the user, the feedback is usually limited to force-feedback joysticks controlling manipulators with gripper end-effectors [Hannaford et al. 1991, Backes 1992].

Concerning dexterous telemanipulation, Turki and Coiffet [1995] discussed a bi-level control framework with low level control for fingertip mapping and high level control for grasp execution and adaptation. The work mainly concentrated on the difficulties of transforming human hand motions to slave hand motions and reproducing contact forces at the operator's fingers using a data glove and exo-skeleton. Michelman and Allen [1994] applied the concept of shared control to the Utah/MIT dexterous hand. Their system did not have provisions for haptic feedback, but focused on defining and sequencing primitives for operations such as grasping an object and inserting a peg in hole.

Li et al. [1996] developed an ambidextrous (dual arm, dual hand) tele-robot. Shared control was used in forming grasps with dexterous hands. An instrumented glove provides spatial configuration information for the hand while voice commands could call up grasp primitives. Other researchers developed interactive task-based programming for shared control of a multi-sensored hand for the DLR (German Aerospace Research Establishment) [Brunner et al. 1994]. Researchers at the NASA Johnson Space Center have developed Robonaut, a humanoid robotic system with a dexterous hand and a telepresence interface. The control architecture is based on "subautonomies" that combine controllers, sequencing, safety systems, and low-level intelligence for reflexive actions [Ambrose et al. 2000, Lovchik and Diftler 1999]. While each of these systems demonstrates the application of shared control for dexterous telemanipulation (or at least for telemanipulation involving a gripper with controllable grasp forces), none specifically addresses haptic feedback at the hand or fingertip level.

2.3 Conclusion

This chapter introduced some of the common terminology and analysis methods used to describe teleoperation and telemanipulation systems. Additionally, some of the fundamental issues that must be considered when developing a telemanipulation system and methods

for improving performance were presented. In particular, the stability-transparency trade-off inherent to force reflecting teleoperation systems was discussed. As one possible solution to overcoming some of the limitations of purely bilateral systems, the concept of shared control was introduced. As we will see in later chapters, a shared control framework provides the foundation for the development of a dexterous telemanipulation system capable of improving performance compared to a traditional bilateral telemanipulation.

3 Dexterous Telemanipulation Test-Bed

The work presented in this thesis was carried out using an hand-and-finger based telemanipulation system with haptic feedback. It is useful to first describe the telemanipulation apparatus and aspects of the individual components. As we will see after having described the system, several issues become readily apparent and motivate the work described in the following chapters.

3.1 Experimental System

The telemanipulation test-bed consists of several custom and off-the-shelf components integrated to form a complete master-slave telerobotic system. The system is designed to provide an intuitive interface for remote dexterous telemanipulation. To this end, the master system is centered around the human hand and finger motions. An instrumented glove is used to measure finger motions and a light weight exo-skeleton force feedback device provides fingertip-level force feedback to the operator (Figure 3-1a). Additionally,

a) b)

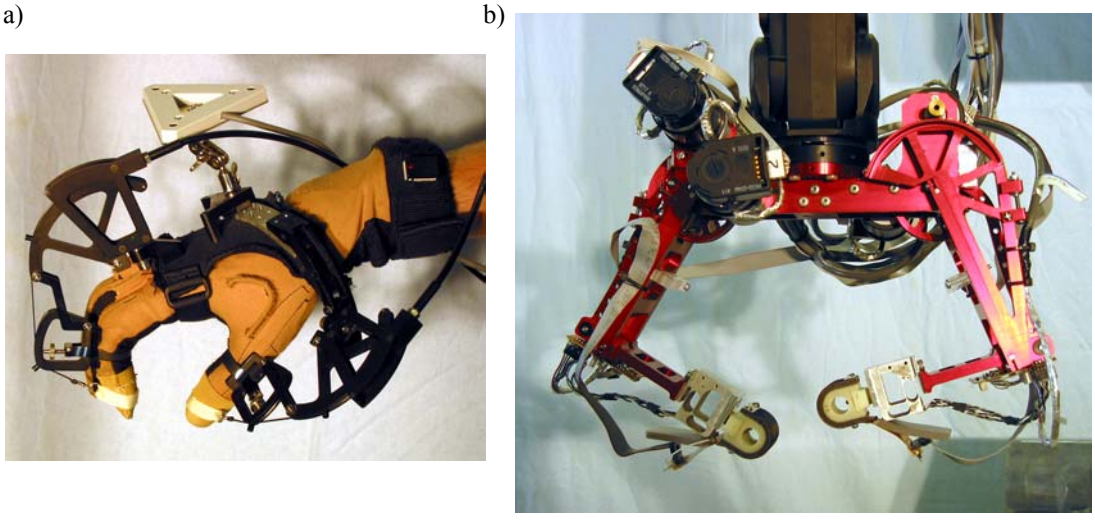

Figure 3-1. a) The master system consisting of an instrumented glove for finger motion measurement and an exoskeleton for fingertip force feedback. b) The slave robotic hand with two fingers and fingertip force sensors for relaying environmental interactions.

an ultrasonic tracking system is used to measure the position and orientation of the hand in space. The slave system consists of a custom designed two-fingered dexterous robotic hand outfitted with force and tactile sensors (Figure 3-1b). The robot hand is attached to a larger industrial robot for increased workspace. The architecture of the our bilateral system is nominally a two-channel "position-force" set-up, in which positions are sent from the master to the slave and measured forces are sent from the slave to the master. A detailed description of the components and system architecture follows.

The telemanipulation test-bed is located in the Dexterous Manipulation Laboratory at Stanford University. It should be noted that since our main focus is on testing a shared control architecture with haptic feedback, several aspects of the telemanipulation system are simplified. While small time delays are present in the existing system and must be considered when tuning the system, the effects of large time delays are not expressly investigated. Thus the local and remote systems are in the same location and controlling computers communicate with as high a bandwidth as possible. The system also differs from some other telemanipulation systems in that the operator has direct visual feedback of the slave robot's environment. However, studies have shown that direct versus video viewing may not significantly affect performance unless the direct viewing angle is limited in comparison to video images [Massimino and Sheridan 1994].

3.1.1 Master System

The CyberGlove[1] is a commercially available instrumented glove with 22 bend sensors measuring joint angles of the hand. The resolution based on each sensor is 0.2° to 0.8° depending on the particular joint's range of motion. Because the slave system is a two fingered manipulator, only the appropriate sensors for the thumb and index finger are measured, allowing for a data collection rate of 200 Hz. Running a calibration routine, specialized for dexterous manipulation, is necessary for each individual user and has been developed in previous work [Griffin et al. 2000, Turner 2001]. The calibration software ensures that each user's internal 3D hand model is as accurate enough to detect fine finger manipulation motions. The human-to-robot mapping software, detailed in Chapter 4, trans-

1. Immersion Corp., San Jose, CA www.immersion.com

forms the user's finger motions into commands for the robotic hand and maps sensor information from the robot system back to the operator.

The operator receives feedback from the tele-robot and environment through several channels. Fingertip-level force feedback is provided by the CyberGrasp[1] system, a cable driven device designed for use with the CyberGlove. Forces are applied to the fingers through cables in Teflon sheaths, which are tensioned by small motors worn in a backpack. Since the cable can only pull along a single axis, the forces applied to each individual finger are unidirectional. The device is grounded to the back of the hand, so no forces restrain arm motion. The motors can apply force up to 12 N and are updated at 1000 Hz to appear smooth and continuous to the user. The system has a resonance peak in the range of 20 Hz and a cutoff frequency on the order of 40 Hz [Turner 2001]. The forces fed back to the operator are determined by the shared control system.

Additional feedback channels exist for the display of indirect quantities, such as information about the grasp stability or other parameters monitored or controlled during shared control. An audio feedback system has been set up that can produce different audio tones that are controlled by the digital I/O ports of the shared control computer. Also since the operator is looking directly at the slave, additional visual feedback is provided by LEDs placed at the robot's fingertips. During telemanipulation, the robotic system can turn on and

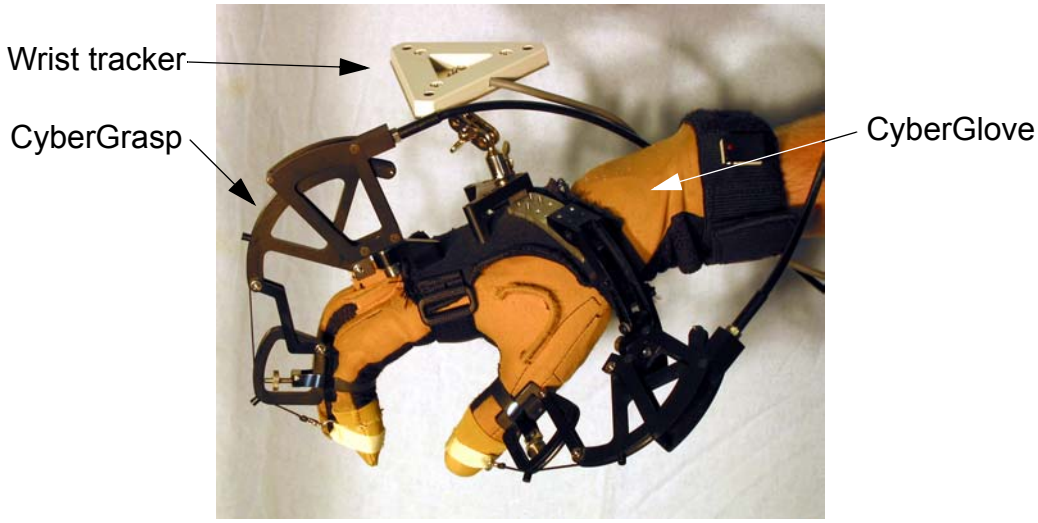

Figure 3-2. Master system major hardware components: the CyberGlove, an instrumented glove to measure thumb and index finger motions; the CyberGrasp, a lightweight hand-grounded force feedback device that applied fingertip forces through tensioned cables; and an ultrasonic tracker for measuring wrist motion in six degrees-of-freedom.

off the LEDs to display state information such as whether or not the system is intervening during the operator's task execution.

To measure the position and orientation of the operator's hand, an ultrasonic 6 DOF tracking system[2] is used. A small receiver attaches to the back of the CyberGrasp device on the operator's wrist allowing for a large natural range-of-motion of the arm. The system has a positional resolution of approximately 0.1 mm and an angular resolution of 0.1°. Accuracy is reported to be on the order of 2% of the distance from the transmitter. The tracking space is approximately a 2.3 m long 100° cone. The tracking information is updated to the master system at 50 Hz through a serial cable connection.

3.1.2 Slave System

The slave system consists of two main components, a robotic hand and an industrial five-axis robotic arm. The robotic hand, referred to as DEXTER, is a custom designed two-fingered hand with two degrees-of-freedom per finger (see Figure 3-3a). Each degree-of-freedom is powered by a low friction, low inertia DC servomotor. The motor is connected to the link through a capstan (cable and drum) drive similar to those found in haptic feedback devices such as the PHANToM[3]. As a result, the hand has low friction, is backdrivable, and has a smooth transmission system. Backdrivability is an important feature because it provides us with an experimental test-bed capable of good force control. Additionally, the drivetrain does not create significant vibrations (as compared to a geared system) that could adversely affect tactile senors at the robot's fingertips. The motors are fairly small, due to weight and space limitations, but are still capable of providing enough force at the fingertips (approximately 7 N in the center of its workspace) to pick up a 250 g object, such as a softball, which more than suffices for the purpose of these experiments. The motors are driven by four linear current amps which are controlled by the servo card D/A channels. The links are approximately 100 mm long and each has over 120° of motion[4]. The workspace of the hand is about 400 mm by 150 mm, with a positional resolution on the order of

2. Logitech 3D Headtracker, manufactured by Fakespace Labs Inc., Mountain View, CA www.fakespacelabs.com

3. SensAble Technologies Inc., Wobburn, MA. www.sensable.com

4. See Appendix C for details on the robot link parameters.

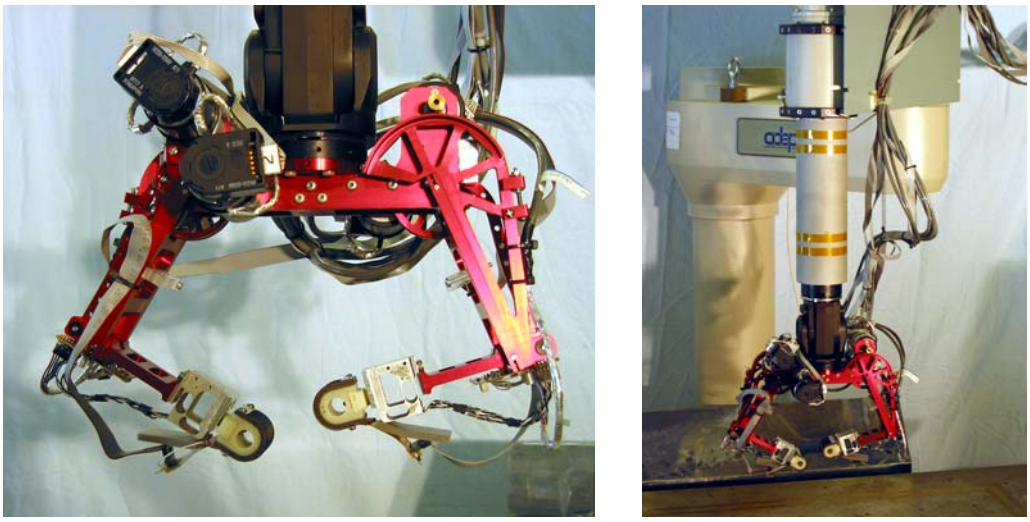

Figure 3-3. a) DEXTER, a custom design two-fingered dexterous robotic hand with fingertip sensors. Each link is driven by a cable and capstan pulley system for a smooth backdrivable transmission. b) The robotic hand mounted on a 5-axis industrial robotic arm provides increased workspace.

0.08 mm (based on 2000 cpr quadrature encoders mounted on each motor). This workspace is sized to best manipulate objects from about one to three inches in diameter or width.

Two-axis strain gage force sensors have been incorporated into the robot fingertips to measure the forces applied by the robot to the object (see Figure 3-4). The force sensors have good linearity, accuracy on the order of ±0.05 N, and are interfaced to the A/D servo card on the slave controller through custom electronics. The fingertips have a dovetail mounting system so that different fingertip types can be easily interchanged on the robot.

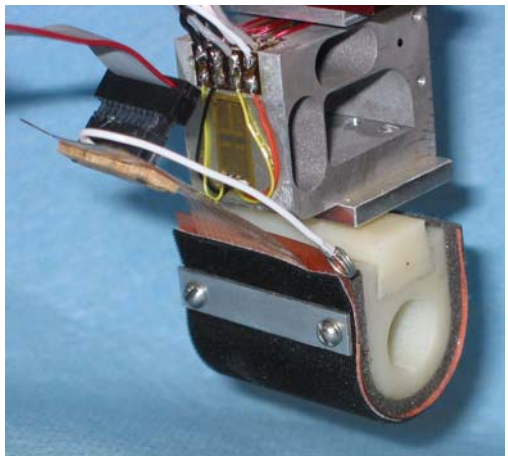

Figure 3-4. Close up of robotic fingertip assembly, showing the two-axis strain gage sensor serving a the connector between the distal robotic link and robot's fingertip. The interchangeable fingertip contains a discrete contact location sensor on a flexible circuit packaged to fit between the base and the outer skin of the fingertip.

For semi-autonomous dexterous manipulation, the robot must know the contact location of the object on the fingertips. To this end, custom contact location sensors have been developed. Our design utilizes an array of contact switches, fabricated as part of a flexible printed circuit. The conductive traces of the flex circuit are distributed around the circumference of the robot fingertip and provide discrete contact location. These sensors detect a circumferential line of contact on the fingertip, allowing us to calculate a contact angle. The contacts are more closely spaced on the inside (anterior) of the finger and less so on the backside (exterior) of the finger, giving a contact angle resolution of approximately 6° (see Figure 3-5). A custom electronics board is used to interface the contact location sensors with the digital I/O ports of the servo card on the slave controller. Software debounces and smooths the contact angle information for use with the object impedance cooperative control law and contact based control mode switching (see Chapter 5).

To increase the workspace of the robot hand, the robot hand is placed on the end of an AdeptOne-MV[5] robot, a 5 DOF SCARA industrial robotic arm, see Figure 3-3b. The Adept has a positional resolution of 0.04 mm and 0.05° in the rotational axes. The workspace is approximately 1100 mm long by 350 mm wide by 175 mm high. Specialized additional robot control software[6] allows for the real-time trajectory control of the robot end-effector and takes path updates at 62.5 Hz. The trajectory is initially created by the human

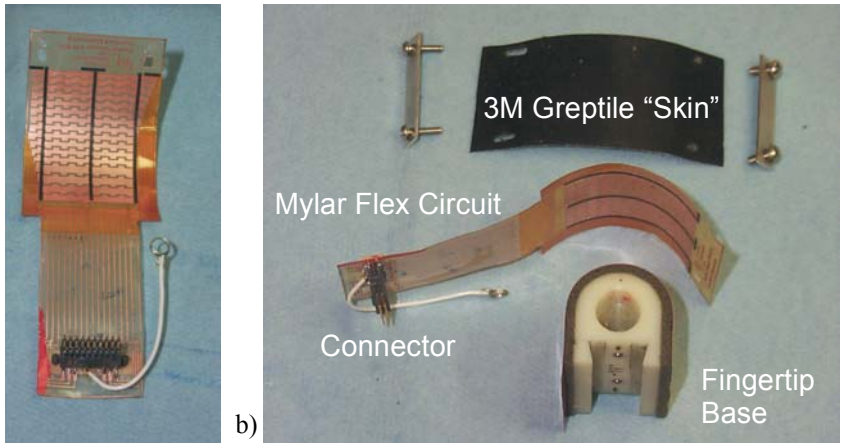

Figure 3-5. The contact location sensor components: a) the flexible circuit with conductive traces for discrete contact location measurement, b) exploded view of complete fingertip assembly.

5. Adept Technology Inc., Livermore, CA. www.adept.com

6. V+ ver 12.4 with Enhanced Trajectory Control allows Adept programmers to utilize the ALTER command, making the real-time trajectory updates possible.

operator and is measured using the ultrasonic tracker. The data are then smoothed and sent to the Adept through an ethernet socket connection approximately every 16 ms. A specialized algorithm also accounts for the asynchronous communication rate between the master system and the Adept controller (see Appendix A for details). Additional software running on the Adept controller limits the velocity and acceleration for smooth and safe motion. Four degrees of freedom are controlled on the Adept, the 3D position and the end-effector yaw (perpendicular to the plane of the table, which corresponds to the operators wrist yaw). The pitch axis on the robot arm is held constant. The workspace is also limited to prevent near singular configurations and any possible damage to the robotic hand. The x-y-z positional workspace is a rectangular box 908 mm by 281 mm by 163 mm with a yaw workspace of ±130°.

During control of the Adept robot, the operator can use a hand held switch to engage or disengage the motion of the robot arm from the motion of his arm. This ability, know as "clutching," allows the operator to take full advantage of the robots large workspace and re-position or re-orient his hand to a more comfortable position. Clutching is very similar to the methods used when attempting to move a mouse pointer across the screen. To move a large distance, a user might have to pick up the mouse and re-position it on the mouse pad. Applying this to the telemanipulation system, to move the robotic hand across the entire workspace, the operator translates in one direction as much as possible and then disengages the robot arm's motion from his own. Once he has disengaged his arm motion from that of the robots, he can re-position to a comfortable pose and then re-engage to continue the translation.

3.1.3 Software and System Architecture Overview

There are many considerations when designing a complex telemanipulation system. In particular, the system must control multiple devices for concurrent operation. Often there is a disparity in the rates of information exchanged among the different components of the system and along different channels. The highest rate, approximately 1 kHz, is used to complete servo loops and communication within the robot controller. Information gathered from the data glove and other external components occurs at lower rates (50-200Hz). Communication between the robot and the human-to-robot mapping software also occurs at a

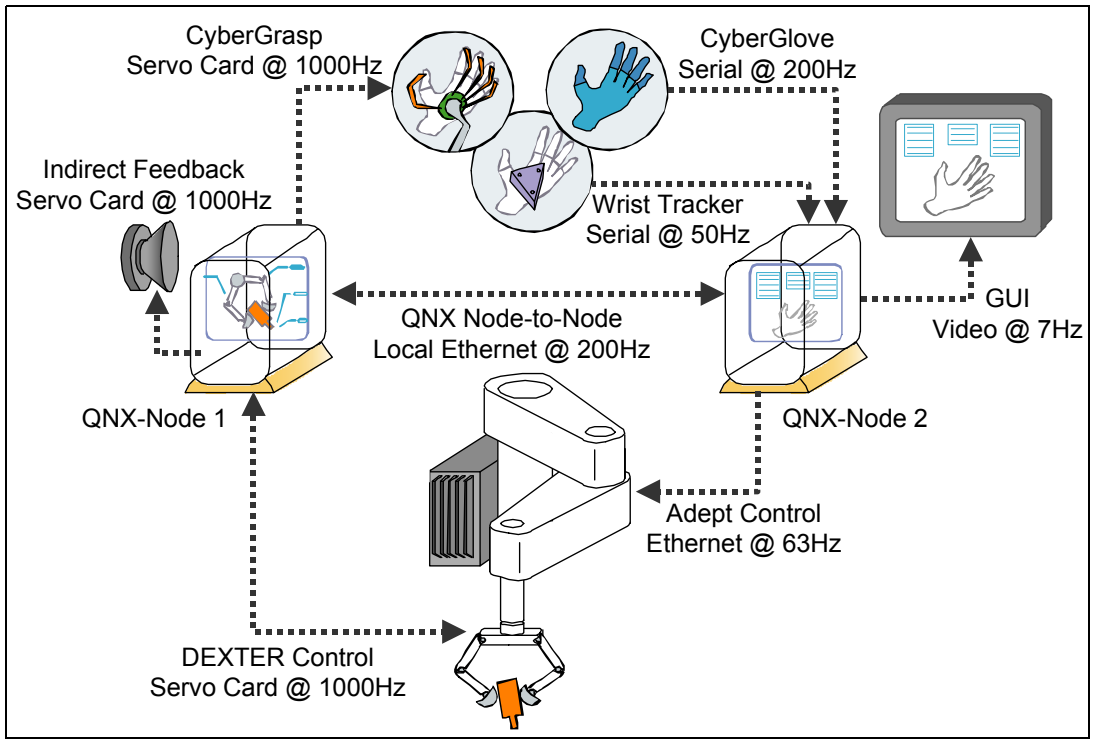

Figure 3-6. Overview of hardware components and system architecture and communication rates. Principal functions and peripherals controlled by each QNX node are described in Table 3-1.

lower rate (200Hz). The slowest data rates (less than 10Hz) are associated with graphical user interface. To accommodate the various communication rates and priorities associated with the different system components, the software framework was developed on a real-time operating system known as QNX[7].

3.1.3.1 Process Based Structure

Due to the overall system complexity, writing a single program to control all the various peripherals and take care of the computational aspects would be an overwhelming task. Instead, we have created a multi-process structure, in which each process functions as a device driver or a dedicated computational program using shared device data. We have implemented a multi-process two node system that roughly separates the slave and master systems and performs the tasks described in Table 3-1. Each of the tasks described is per-

7. Version 4.25, QNX Software Systems Ltd., Ontario, Canada www.qnx.com

Table 3-1. Descriptions of the various process tasks that are performed on each node of the telemanipulation system.

Node 1 - Pentium II 233 MHz processor		
Process Task	Description	Frequency
Robotic hand control	Performs all robot I/O, kinematics computation, control, force and tactile sensing, using an 4-axis servo board installed on the ISA bus	1000 Hz
Shared control	Creates and updates object model based on tactile sensors, monitors critical parameters for intervention, and determines appropriate control mode switching	1000 Hz
Haptic feedback	Performs I/O and simple computations to drive the CyberGrasp force-feedback system	1000 Hz
Indirect feedback	Performs I/O for audio feedback circuitry and LED indicators for visual feedback	1000 Hz
Node-to-node communication	Communicates with Node 2 through a virtual inter-process communication link using a direct ethernet connection; receives commanded robot positions and control mode information and sends status information	200 Hz
High level control	Interacts with user through command line interface that allows real-time slave robot mode switching, gain modification, and data capture; any modified variables are updated in the next servo cycle	user driven
Node 2 - Pentium II 500 MHz processor		
Process Task	Description	Frequency
Human-computer interface	Communicates with the CyberGlove interface unit over a serial line to obtain raw measurements of glove sensor data	200 Hz
Virtual hand model	Computes the virtual hand positions based on the raw glove data and calibrated kinematic model	200 Hz
Human-to-robot mapping	Performs calculations to intuitively map motions from the virtual human hand model to the planar robotic manipulator	200 Hz
Graphical user interface	Updates the virtual hand model and commanded robot position for verification of calibration and mapping for each individual user	7 Hz
Node-to-node communication	Communicates with Node 1 through a virtual inter-process communication link using a direct ethernet connection; sends commanded robot position and control mode information and receives status information	200 Hz
Wrist tracker	Communicates with the 6 DOF ultrasonic wrist tracker over a serial line	50 Hz
Slave arm control	Communicates smoothed wrist positions to AdeptOne robot controller over an ethernet socket connection	62.5 Hz
High level control	Spawns additional sub-tasks which capture data or calibrate the kinematic hand model	user driven
Process coordination	Manages process creation, communication, and miscellaneous housekeeping	n/a

formed by a separate executable process. By scheduling the processes with different priorities, the most time-critical tasks, such as the robot control, receive CPU resources when needed. Figure 3-6 illustrates the physical system architecture and the communication rates between the system components.

3.2 System Advantages and Limitations

Developing a telemanipulation system centered around the human hand and fingers has distinct advantages and disadvantages. The glove based system allows operators to make natural grasping and manipulation motions to be carried out by the remote slave hand. However, designing an anthropomorphic robot hand with similar degrees-of-freedom and range-of-motion as the human hand is an extremely difficult task. A few researchers have developed remarkably complex anthropomorphic robot hands, but a considerable amount of effort has been spent on the mechatronic design and control of the hand (e.g., [Hirzinger et al. 2000]). The robotic hand used for our investigations in telemanipulation is a planar two-fingered device with two degrees-of-freedom per finger. Using a kinematically simple robotic hand allows us to easily develop control algorithms for semi-autonomous dexterous manipulation.

One disadvantage to using a human hand-and-finger based master controller is the difficulty in designing a high fidelity force feedback device that does not significantly limit the operator's natural manipulation motions. For our telemanipulation system we use the CyberGrasp system, which provides unencumbering fingertip force feedback. Because the feedback is cable driven, the forces applied to the fingertip are uni-directional. The device allows for a fairly realistic display of forces that would typically be encountered when using an opposing grasp to manipulate small objects. However, the feedback is only an approximation of the two-axes of measured forces by the planar robotic hand fingertip sensors. Similar to the forward position mapping, the mapping of the forces back to the human hand is not one-to-one. Additionally, because the device is attached to the operators wrist, the forces are "ungrounded" or arm-grounded. This type of device is appropriate for displaying the internal force applied to an object held in the slave robot's hand, but cannot properly display the weight of the object or object interactions with a fixed environment.

Typically, arm and whole-hand force feedback devices are larger than finger-based systems and force sensors can be easily incorporated, allowing for more complex bilateral architectures such as local master force feedback or a four channel architecture. The addition of fingertip force sensing to the existing CyberGrasp exo-skeleton would be a difficult task and require significant additional hardware. The single degree-of-freedom and unidirectionality of the fingertip force feedback system also makes closed loop master control difficult. While the position of the master (the human hand) is known through glove measurements, the CyberGrasp system may not be able to apply the necessary control forces. For example, it is often desirable to add damping to the master system to help stabilize the bilateral system [Daniel and McAree 1998]. Unless the cable is under tension, it is not possible to apply damping in both directions of cable motion. The inability to reduce contact rebound or recoil motions (as the cable may go slack) limits the system performance and/or requires the operator to increase his/her finger impedance through co-contraction.

The CyberGrasp also has a relatively low bandwidth (approximately 20 Hz) compared to the human tactile sensitivity range. Thus, the device can only display "energetic" information for proprioceptive force feedback. Any higher bandwidth information must be supplied indirectly (e.g., visual or audio displays) or directly with another device (e.g., voice-coil actuator or piezo benders).

The slave system is located across the room from the master system, giving the operator direct visual feedback during task execution. To maximize system performance and stability, time delays between master and slave systems were minimized. Ideally, to eliminate communication delays, the master and slave systems would be controlled from the same computer. However, for our experimental set-up, computational requirements prevented a single computer system. As discussed previously, a two-node networked computer system was implemented and roughly separates the master and slave functions. The master glove hardware and software and the graphical user interface (used for individual operator calibration) reside on one computer while the slave is controlled on another computer with the an ethernet socket communication linking the nodes. The master force feedback device is controlled on the same computer as the slave robot, and in doing so, prevents

additional communication delays. The treatment of large time delays and its effect on system performance is beyond the scope of this thesis.

3.2.1 Tuning for System Performance

Several different methods were investigated to improve the performance of our system. For example, a certain amount of artificial damping could be added to the master system, especially in the grasping "direction." However, adding more damping caused the control of the slave fingers to feel "sluggish" and tended to be tiresome for the operator. Additionally, because the exo-skeleton is a lightweight device, the increase in apparent inertia due to the damping felt incongruous during free space motions. Limiting the damping on the master side to the damping inherent in the CyberGrasp system gave a "crisp" sense of contact when grasping an object with the slave hand.

Even though the time delay was reduced as much as possible, a certain amount existed due to signal filtering, small computer-to-computer communication delays, and the disparity in data collection rates among the different system components. A low pass filter was applied to the glove data (recorded at 200 Hz) to created smoothed commanded positions for the robot.

Another important aspect of the hand-and-fingered centered system is the difference between the master and slave masses. As Daniel and McAree [1998] point out, the master-slave mass ratio places a fundamental limitation of the performance of the teleoperated system. In particular, if the mass of the slave is larger than that of the master (as in our case), they argue the reflected force ratio must be decreased by a proportional amount to preserve stability during contact interactions. However, because the size scaling is roughly one-to-one, we desired maintain a one-to-one force scaling. Thus to ensure stability for most conditions, the slave stiffness was increased until the system showed signs of near instability and then reduced slightly. Despite somewhat conservative gains, some subjects could still cause the system to "recoil" upon grasping an object. Often this could be resolved by asking operators to make manipulation motions in a more deliberate manner. In some cases, it was necessary to ask operators to increases his or her grasp stiffness. However, with very little training and practice most operators were able to easily use the system

to perform simple tasks. Ultimately, the telemanipulation system provided an adequate amount of transparency for our experiments in telemanipulation.

3.3 Developing an Immersive Telemanipulator

Having described our system and its advantages and limitations, several issues become readily apparent and must be addressed to fully benefit from the immersive nature of the hand-based telemanipulation system.

As discussed previously, the slave hand used for our telemanipulation system is a non-anthropomorphic two fingered planar robot. While the kinematics simplify the control, especially for dexterous manipulation, significant dissimilarities between the master (a three-dimensional computer model of the human hand) and the robot hand must be accounted for. A human-to-robot mapping method was developed to allow for intuitive control over the robot hand through natural human hand motions. The mapping method is discussed in the next chapter, Chapter 4.

Several aspects of the experimental system could potentially limit the performance of an operator using our telemanipulation set-up. Limitations such as time delays, bandwidth, uni-directional actuation, compliance, and friction all conspire to make high fidelity telemanipulation difficult to achieved. The incorporation of low level "intelligence" for remote object manipulation can reduce the demands on an immersive telemanipulation system. Shared control has the potential to improve performance in telemanipulation systems and overcome limitations inherent to practical implementations. By enabling the slave to share control over some aspects of manipulation, such as grasp force regulation, the operator can focus on higher level aspects of the task such as planning. Additionally, by sharing control during critical periods of a task, time delays become less detrimental because commands from the master are supplemented by local control to prevent unwanted slip or object damage. The shared control system can also utilize additional feedback paths to the operator to display information gathered by the semi-autonomous controller that may not have a physical equivalent on the master system. The details of a shared control framework for dexterous telemanipulation are discussed in Chapter 5.

With the implementation of a shared control system, questions arise concerning the efficacy of the method and how best to provide necessary cues to the operator. As the robot assumes more control, the operator may lose the sense presence normally provided by bilateral systems. Thus, in Chapter 6, we investigate the effects of shared control and various feedback strategies.

4 Human-to-Robot Mapping

To develop a dexterous telemanipulation system, operator input to the master system must be transformed into commands for the remote slave manipulator and, ideally, information about the interactions between the slave and the environment are fed back to the operator. In the traditional two-channel bilateral setup, position commands are sent from the master to the slave and force commands are sent from the slave back to the master. The main focus of this chapter is on the forward positional mapping for a dexterous slave hand. This chapter presents a method that maps unconstrained human hand and finger motions to a non-anthropomorphic planar robotic hand.

The master system is comprised of an instrumented glove worn by the operator. Glove sensor readings are applied to a calibrated eight degree-of-freedom kinematic model of the hand to produce fingertip positions. The slave system is a planar two-finger manipulator with only two degrees-of-freedom per finger. Because of the kinematic dissimilarities between the operator's hand and the slave robot, a simple joint space mapping or projected Cartesian space mapping does not allow the operator to easily control the robot. To overcome this difference, a method was developed that determines the operator's intended manipulation motions and maps those motions to the robotic fingers. The intended motions are captured analytically by assuming the operator is manipulating a virtual object

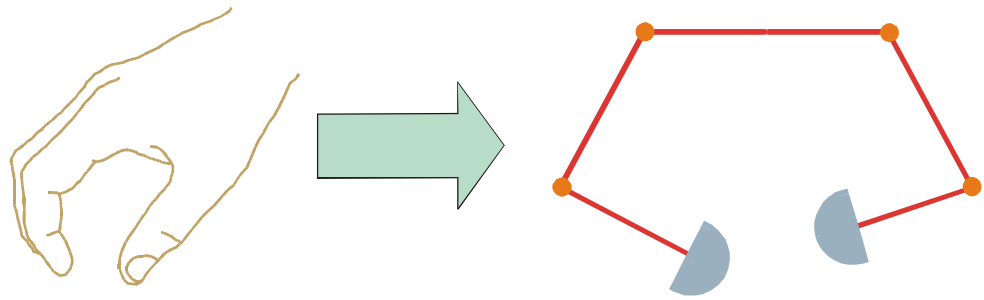

Figure 4-1. Various mapping methods exist for mapping human hand motion to robot hand motion. However, the task becomes significantly more difficult when mapping to a non-anthropomorphic robot hand. The method should produce a predictable and intuitive mapping, preserving the intent of the human motions.

between the fingers. The virtual object parameters that best describe the object motion in a plane are then mapped to the robot hand space. The parameters are independently adjusted and scaled to best match the workspace of the robotic hand and simultaneously allow the operator to intuitively control the robotic hand motions for object manipulation.

The following sections describe the development of a new mapping method for dexterous telemanipulation. We first review relevant work in the area of human-to-robot mapping. Next, a brief overview of our master and slave system is presented. An initial mapping scheme is presented and its deficiencies provide the motivation for the development of a more complex mapping method. The "virtual object based" mapping method for telemanipulation is introduced and its implementation is described in detail. Finally, the extensibility of the method is discussed and other methods for different control modes are covered.

4.1 Previous Work

Many previous mapping methods for dexterous telemanipulation have been developed for systems with anthropomorphic slave hands. The mapping of position and orientation information gathered from the human hand is simplified if the robot hand closely matches the kinematics and workspace of the human hand. However, the kinematic complexity and high number of degrees-of-freedom of the human hand make the design of a truly anthropomorphic hand a difficult task. Therefore, several mapping methods have been investigated to account for the differences between the human hands and robotic hands.

For master-slave systems with nearly identical kinematics, a joint-to-joint mapping can be used. Kyriakopoulos et al. [1997] employed a simple linear function ($^{R}\theta_i = m_i \cdot {}^{H}\theta_i + b_i$) for the mapping of human-hand joint angles to robot joint angles. Human hand motions were measured using a modified Exos Dexterous Hand Master, an aluminum exo-skeleton type "glove" that straps to the fingers to measure joint angles. The slave robot was an anatomically proportioned hand with a total of sixteen degrees-of-freedom, four for the thumb and three for each finger (matching the hand master). The gains and intercepts of the mapping functions were based on each finger's maximum extension and minimum flexion readings from the master and the corresponding robot finger range-of-motion.

The linear joint mapping method can also be applied to systems with less anthropomorphic robotic hands. A commonly used semi-anthropomorphic slave is the Utah/MIT Dextrous Hand. The flexion/extension of the robot fingers is kinematically similar to the human hand. However, a roll motion at the finger base, about an axis nearly parallel to the palm, is used to approximate human abduction/adduction motion. Additionally, the thumb is located more centrally in the palm and the base joint is simplified in comparison to the human thumb. The application of the linear joint mapping method to semi-anthropomorphic hands has typically resulted in unsatisfactory performance. Mapping from the EXOS hand master to the Utah/MIT hand, Speeter [1992] found that the kinematic dissimilarities prevented touching of the fingertips from being correspondent, although, with sufficient training operators could "attain a significant degree of effectiveness." According to Rholing et al. [1993], using a linear joint mapping method to map from the Utah Dexterous Hand Master (a device similar to the EXOS system) to the Utah/MIT Dextrous Hand required operators to make contorted poses to achieve the desired robot motions.

An interesting approach to joint-to-joint mapping, known as pose mapping, was developed by Pao and Speeter [1989]. The algorithm was based on the development of a transformation matrix to match poses of the human hand with those of the robot hand. For several distinct poses of the robot hand, the operator was asked to make a similar pose and the joint angles of the human hand and those of the Utah/MIT robot hand were recorded. (Operators poses were recorded using the VPL data glove, measuring joint angles using optical fiber based bend sensors). The transformation matrix was found through pseudo-inversion which was then used to calculated robot joint angles from the measured human hand configuration. Speeter [1992] later reported that "performance was somewhat erratic, depending critically on the quality and consistency of the training set of poses." While the method appears to be beneficial in terms of its extensibility to various master/slave configurations, Rholing et al. [1993] found the method produced unpredictable movement of the robotic fingers.

Some of the drawbacks of joint space mapping methods were addressed with the introduction of fingertip position mapping methods [Hong and Tan 1989, Speeter 1992]. Using the data obtained from a hand master applied to a kinematic model of the human

hand, forward kinematics are used to compute fingertip positions (and possibly fingertip orientations). The Cartesian position data are then translated into the slave hand frame. Using inverse kinematics, the joint angles for the robotic hand can then be computed.

Using a combination of joint-to-joint mapping for the fingers and Cartesian mapping for the thumb, Hong and Tan [1989] performed several telemanipulation tasks, mapping the VPL DataGlove data to a Utah/MIT robot hand. The linear relationships and mapping parameters were based on empirical observations. Speeter [1992] used the fingertip mapping method to control all fingers of a Utah/MIT hand with a EXOS Dexterous Hand Master. However, the author reported that the kinematic dissimilarities resulted in only a partial overlap of the fingertip workspaces. Rholing et al. [1993] addressed this problem with the introduction of an optimized fingertip mapping method. In a method similar to Speeter [1992], forward kinematics were used to determine the position and orientation of the human fingers and then mapped to a reference frame in the robotic hand. Inverse kinematics were used to determine fingertip configurations. However, if no solution was possible due to kinematic dissimilarities, the algorithm minimized the human-robot fingertip position and orientation error within the constraints of each robot finger's workspace. The algorithm also ensured continuity in robot finger motion if a mapped position was unachievable.

In cases with anthropomorphic slave hands, Cartesian fingertip mapping is simplified. Fisher et al. [1998] mapped information gathered from a CyberGlove (a wearable glove with bend sensors for joint angle measurement) to the anthropomorphic DLR robot hand using only a translation and linear scaling with a constant factor in all three degrees-of-freedom. If commanded positions were unachievable, the closest reachable position in the workspace was used.

Kang and Ikeuchi [1997] developed a grasp mapping system based on robot programming by human demonstration. Using a CyberGlove and computer vision, hand posture and object contact points were observed during human task execution. This information was then used to search for a kinematically feasible robot hand pose. Once a solution was found, the joint angles and pose were optimized based on a task oriented parameter such as a manipulability measure or sum-of-force minimization.

While many of these approaches have had success when implemented on anthropomorphic and semi-anthropomorphic slave systems, preserving the functional intent of the human hand in systems with greater kinematic dissimilarity becomes a difficult and more ambiguous task [Speeter 1992]. As the design of a slave hand diverges significantly from the kinematics of the human hand, the determination of what robot motions correspond to human hand motions becomes more functionally based than kinematically based.

To capture the intent of the operator's hand motions, Speeter [1992] developed a functional mapping method based on formal grammar and language theory. The idea considered the positional readings of the master as a stream of characters. A lexical analyzer then attempted to determine the syntactic content of the character stream. (Syntactic content determinations attempts to formulate "words" from the character stream. The "words" form phrases which are a representation of specific hand actions and poses.) The syntactic content was parsed into hand actions based on a hand-function grammar database. This classification based approach supports an incremental building of the language but suffers from the need for extensive training and the development of an adequately descriptive hand grammar.

The mapping method presented in this chapter is a combination of functional and kinematic mapping methods. The slave hand used in our telemanipulation test-bed shares some kinematic similarity with the human hand but significant differences exist and must be accounted for. The basic approach is an analytical transformation, providing a predictable and continuous mapping from hand motions to robot motions. However, the hand motions are initially transformed into key parameters which describe the operator's intent. The parameters are designed to capture the salient information necessary for object manipulation with the slave hand. The parameters are transformed to the robot's workspace and independently scaled to account for workspace size differences. The goal of our mapping method is to give the operator the ability to perform fine manipulation tasks as intuitively as possible.

4.2 System Description

The human-to-robot mapping methods discussed below were developed to support our experimental test-bed for dexterous telemanipulation. The important aspects of the master

and slave systems necessary for the development of a mapping method are described here. The full details of the experimental set-up are presented in Chapter 3.

As discussed in the previous chapter, a key component of the master system is the CyberGlove[1], a commercially available instrumented glove with 22 bend sensors measuring joint angles of the human fingers (see Figure 4-2a). The glove provides an unencumbering human-computer interface for capturing the motions of the hand and fingers.

A significant challenge for any human-hand centered master system is accurately and reliably measuring hand motions for different operators. To this end, a specialized calibration method was developed in previous work [Griffin et al. 2000, Turner 2001]. The calibration software was specifically designed for dexterous manipulation, requiring the detection of fine manipulation motions of the fingers. A kinematic model of the human hand is calibrated for each operator yielding three-dimensional fingertip position data (see Figure 4-2b). Only the thumb and index finger are modeled, calibrated, and tracked (the slave robot only has two fingers). The calibration is based on creating a closed kinematic chain with the index and thumb (by having the operator place his/her thumb and index fingertips together). Using several poses, the distance error between the fingertips is mini-

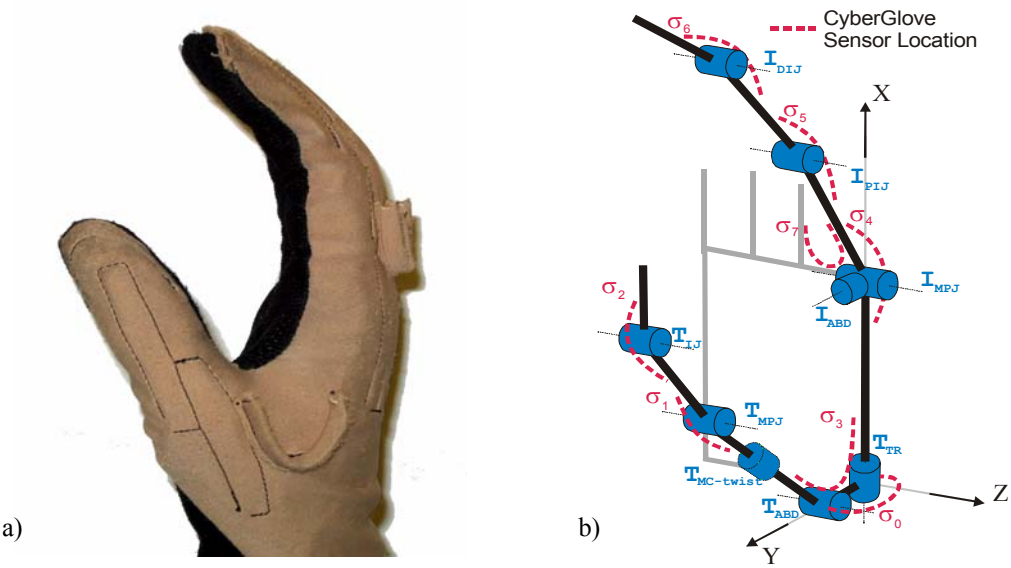

Figure 4-2. a) Instrumented glove for the measurement of human finger joint motion. b) Kinematic model of the hand used to determine tip locations of the index finger and thumb. The model was calibrated for each operator to produce a sufficiently accurate representation of finger motions necessary for dexterous manipulation.

1. Immersion Corp., San Jose, CA www.immersion.com

mized using a least squares fit. The eleven subjects in the experiment described in Chapter 6 were calibrated using this method. For these subjects, the average root-mean-square (RMS) error for the fingertip-to-fingertip distance over all the recorded poses was 4.1 mm with a standard deviation of 0.38 mm. The relative positioning of the operator's fingertips is also important for telemanipulation. The calibration method shows good linearity for a wide range of separation distances [Griffin et al. 2000]. Good linearity corresponds to a better measurement of the size of a virtual object grasped by the operator. The full details of the kinematic model and the calibration method can be found in [Turner 2001].

The slave hand in our system is a non-anthropomorphic two-fingered planar manipulator (see Figure 4-3a). The custom designed hand has two degrees-of-freedom per finger, each powered by low friction, low inertia DC motors through a capstan drive transmissions. The links are approximately 100 mm long and each has over 120° of motion. The planar workspace of the hand is approximately 250 mm by 150 mm. The kinematic model of the hand is illustrated in Figure 4-3b. The robot is controlled using the operational space formulation and an object impedance cooperative control. Thus, fingertip positions and object positions are commanded in Cartesian space (see Chapter 5 for details).

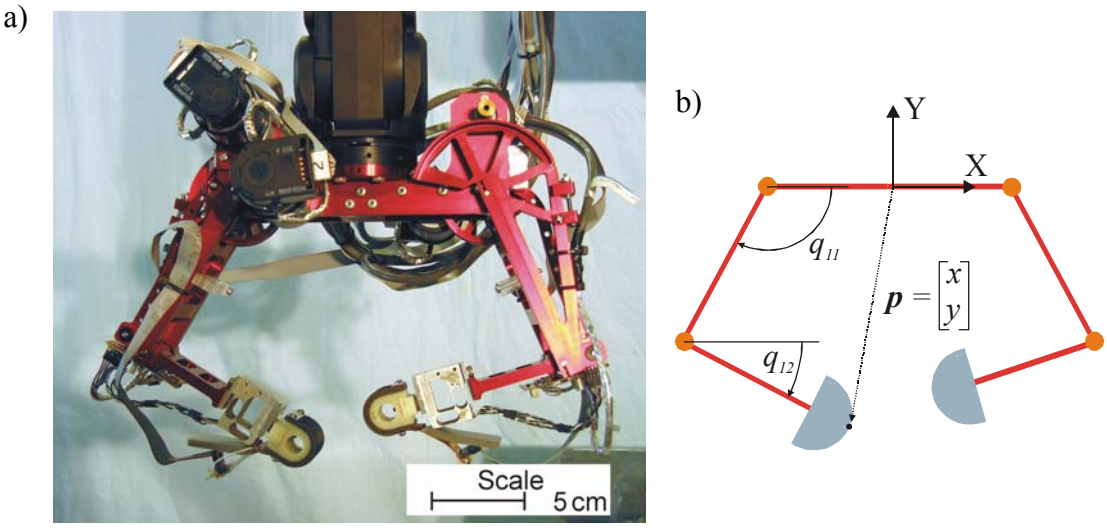

Figure 4-3. a) DEXTER, a custom designed two-fingered dexterous robotic hand with two-degrees of freedom per finger. The left and right finger correspond to the index and thumb of the human hand, respectively. b) A kinematic model of the robotic hand.

4.3 Point-to-Point Mapping

Difficulties arise when attempting to map the three-dimensional motion of the human hand to the two-dimensional motion of the planar robot hand. Due to the differences in the kinematics, a joint-to-joint mapping for both fingers is not possible. An initial solution to the mapping problem is a point-to-point mapping in which Cartesian human fingertip positions are mapped to robotic fingertip positions using a simple linear transformation. A Cartesian based mapping method allows for fingertip-to-fingertip correspondence (i.e., when the operator brings his/her fingertips together, the robot's fingertips also come together), which is important for dexterous manipulation and intuitive motion mapping. The same transformation must be applied to both fingers for fingertip-to-fingertip correspondence over the robot's entire workspace.

4.3.1 Implementation

The measured glove data are applied to the calibrated kinematic model producing three-dimensional fingertip positions for the thumb and index finger. The positions are then projected on the X-Y plane within the hand frame as in Figure 4-4. The X-Y plane was chosen because the index finger's motion lies primarily within this plane (defined by the kinematic model); abduction/adduction motions are fairly small during normal manipulation motions.

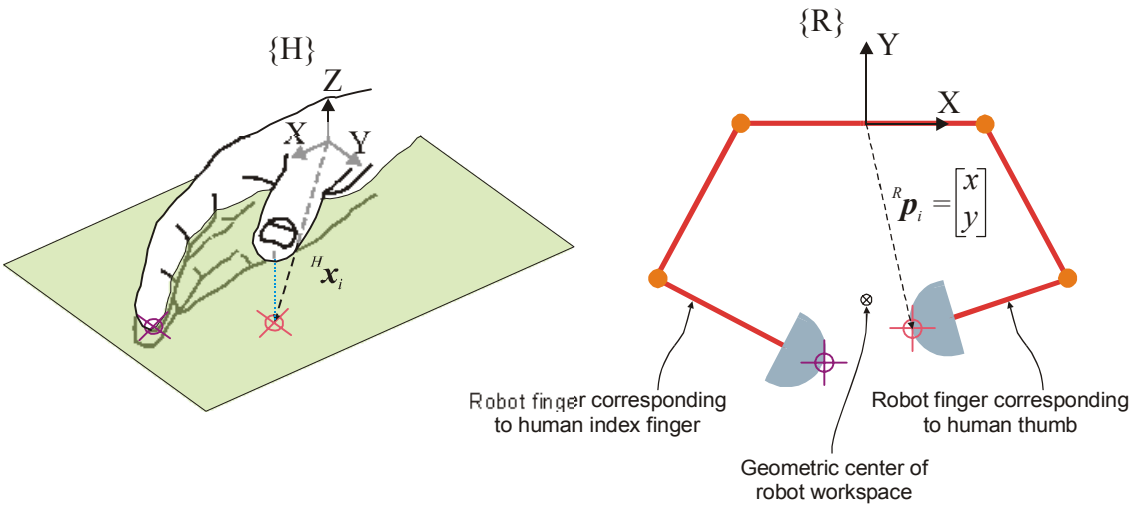

Figure 4-4. The point-to-point mapping method concept. The index and thumb tip positions are projected onto the X-Y plane within the hand workspace and then transformed to the robot workspace using a standard frame transformation with linear scaling.

The projected tip positions are transformed and scaled to the robot workspace using a standard planar frame transformation:

$$ {}^{R}\boldsymbol{p}_i = \begin{bmatrix} g_x & 0 \\ 0 & g_y \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \cdot {}^{H}\boldsymbol{p}_i + \boldsymbol{p}_o \qquad (4.1) $$

where ${}^{H}\boldsymbol{p}_i$ represents the X-Y planar projection of the i^{th} fingertip (i = 1 or 2) in the hand model frame, which is mapped to ${}^{R}\boldsymbol{p}_i$, the corresponding robot finger. For both fingers, the projected hand position first undergoes a rotation by amount θ. The rotational amount is determined such that projected open-and-close grasp motions of the human hand correspond to horizontal opposing grasp motions in the robot's workspace. The scaling factors, g_x and g_y, are chosen to scale the operator's finger motions to the larger workspace of the robot fingers. Both fingers are also translated by amount $\boldsymbol{p}_o$ such that when the operator brings his/her fingers together in a comfortable pinch pose, the robot's fingertips meet as close as possible to the geometric center of the robot's workspace (Figure 4-4). Thus, the point-to-point mapping method is defined by five transformation parameters (θ, ${}^{H}\boldsymbol{p}_i$, $\boldsymbol{p}_o$, g_x, and g_y). The five parameters are adjusted for each operator to best utilize the robot's workspace while trying to maintain an obvious correlation between human finger motions and robot finger motions.

4.3.2 Method Results

Under this mapping method, the motion of the robotic finger corresponding to the index finger was relatively easy to control. This result is primarily due to the kinematic similarities between the index finger and the robotic finger. The abduction/adduction range-of-motion for the index finger is much smaller compared to the flexion/extension range-of-motion, roughly constraining the index motions to a plane. Additionally, motion of the distal joint is coupled to the motion of the interphalangeal joint (the middle knuckle) [An et al. 1979] further reducing human index finger motion to approximately two degrees-of-freedom. The range-of-motion of the robot finger also corresponds well with the flexion extension range-of-motion of the index finger, yielding a similar workspace shape. With minimal adjustment of the mapping parameters, the operator could intuitively control the

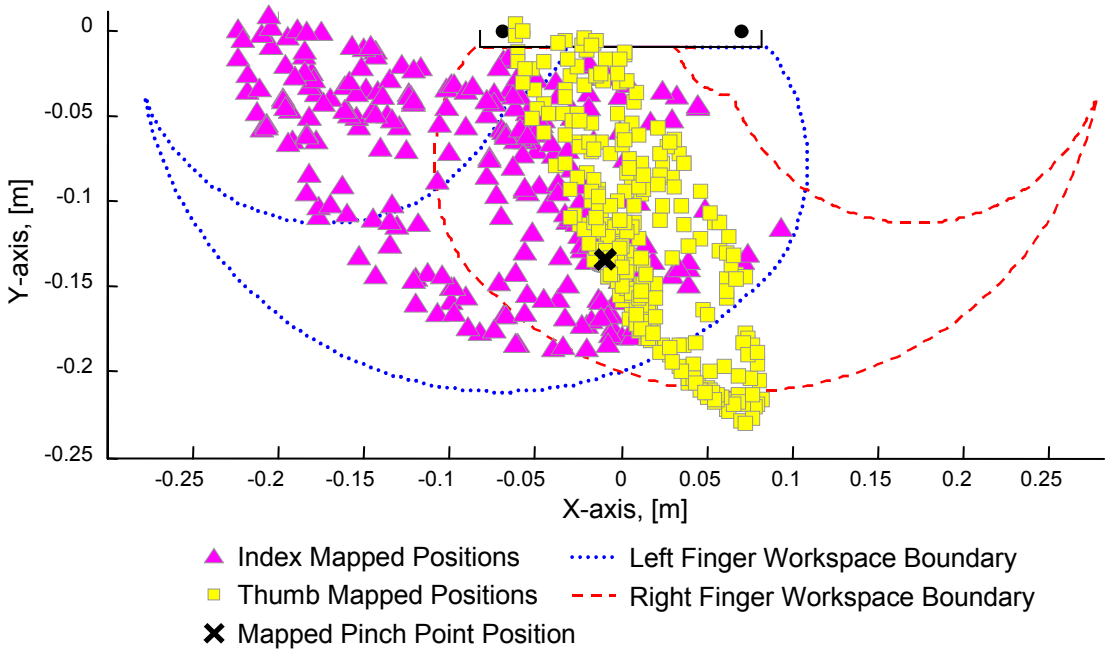

Figure 4-5. A typical plot of achievable positions under the point-to-point mapping method. The index and thumb map to the robot's left and right fingers, respectively. Also the mapped pinch point position is indicated.

motion of the index finger. However, control of the robotic finger corresponding to the thumb proved awkward.

Figure 4-5 shows an example of the achievable positions in the robot's workspace based on human hand motions as mapped under the point-to-point method. The mapped positions of the index and thumb are plotted over the workspace of the robotic hand, where the index and thumb correspond to the left and right robot fingers, respectively. As shown in the figure, the index finger positions map fairly well to the left robot finger workspace (with some shifting in the upper left of the workspace). The operator is capable of using a large percentage of the left robot finger workspace. However, the achievable positions for the thumb are mapped to a relatively small region of the corresponding robot finger workspace. Even with the thumb's large range-of-motion in the hand frame, the projected tip positions in the X-Y plane were approximately confined to a line. Interestingly, based on the robot's kinematics and the achievable positions of the thumb, the motion of the right robot finger usually resulted in motion primarily occurring at the second joint with the first

link remaining almost stationary. This tended to frustrate operators and many formed contorted thumb poses in attempting to use more of the robot's available workspace.

To improve the thumb's achievable positions in the robot's workspace one could simply increase the gains for the thumb transformation. However, because we desire a mapping with fingertip-fingertip correspondence, the gains (and offsets) must be the same for both the index and thumb. To prevent a large percentage of the index achievable positions from falling outside of the workspace and large velocities of both fingers, the gains can not be increased to an arbitrarily large value to improve the achievable thumb positions. Also, when setting the gain values to best fill the index and thumb workspaces, the mapped pinch point tends to fall in the lower portion of the robot's workspace. Operators tend to manipulate small objects near the edge of the human workspace boundary but the robot has a greater manipulation range near the geometric center of its workspace. The point-to-point mapping method couples the workspace coverage with the location of the pinch point.[2]

Also notice that the overall shape of the achievable index finger positions in Figure 4-5 is roughly the same as the shape of the robot's left finger workspace, but appears not to have been rotated enough. This illustrates an additional problem with the point-to-point mapping. The motion of opening and closing one's grasp is not necessarily perpendicular to motions made into and away from the palm with the fingertips together (motions that are ideally mapped to vertical motions in the robot's workspace). The parameters chosen for the mapping shown attempt to best match the workspace and still provide an intuitive mapping of hand motions to robot motions.

The point-to-point mapping method highlights many of the difficulties associated with mapping from the human hand to a planar robot hand. The fundamental problem with the point-to-point mapping method is the inability to give the operator intuitive control over the "thumb" motions of the robot. Because human thumb motion is not primarily planar and the thumb does not directly oppose the index finger, the planar projection approach clearly removes important information about the intended manipulation motion.

---

2. It is possible to decouple the workspace gains from the pinch-point location by applying gains to points measured relative to desired pinch-point location after properly rotating and translating the points to the robot frame. However, the problems with the mapped achievable thumb positions would still need to be addressed.

4.4 Virtual Object Mapping

To address the deficiencies of the point-to-point mapping method, a method was developed based on the intended motions of a virtual object grasped in the operator's hand. With our goal of developing an intuitive mapping, the method should allow an operator to make natural manipulation motions, such as grasping, releasing, or rolling of an object, and have the robot perform analogous motions. The fundamental idea of the virtual object mapping method assumes that motions of the human fingers are imparting motions to a virtual sphere held between the fingers (see Figure 4-6). The design of our slave robot requires a reduction in the degrees-of-freedom associated with the human fingertip data. Similar to the point-to-point mapping method, the virtual object method reduces the six degree-of-freedom hand data (three degrees-of-freedom per fingertip position) to four degrees-of-freedom (a planar object with a planar position, size, and orientation). The parameters describing the planar virtual object are then mapped to the robot hand frame such that the robot motions intuitively match the hand motions. The parameters are scaled and modified independently to account for the kinematic and workspace differences. When an object is in the robot's grasp, the virtual object motions are mapped directly to commanded motions for the object and commanded internal force. When no object is present, the motions of a (fictitious) virtual object in the operator's hand are used to create corresponding fingertip motions for the robot.

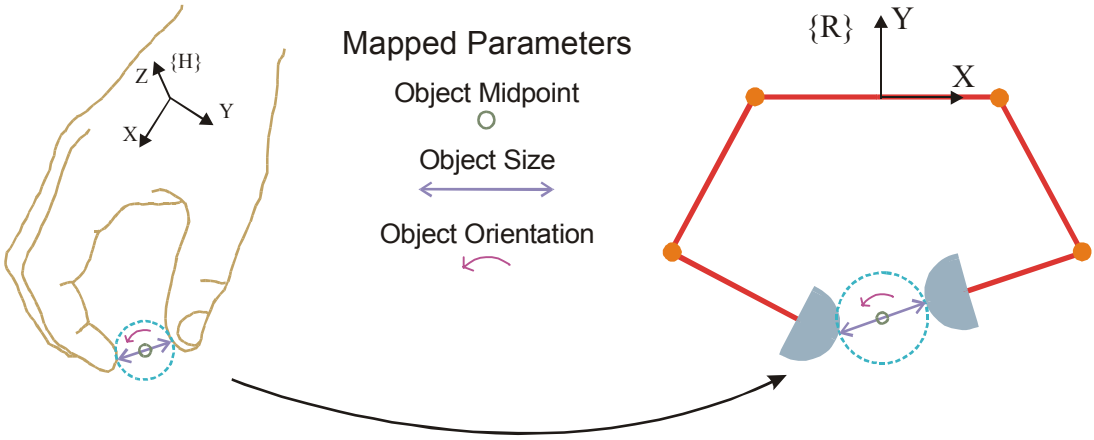

Figure 4-6. The virtual object mapping method concept. At the most basic level, the method assumes the operator's fingertip motions are imparting motions to a virtual object. The relevant virtual object parameters are then transformed from the hand frame to the robot frame.

We believe that basing the robot motions on the virtual object parameters extracts more information about operator's intent than the point-to-point mapping method. In particular, the three-dimensional motions of the thumb are incorporated into the virtual object motion description and the relevant object parameters are mapped to a plane. Additionally, the parameters of the object are scaled independently to better match the human hand workspace to the robot hand workspace and still maintain fingertip-to-fingertip correspondence necessary for dexterous manipulation. Since the virtual object approach is fundamentally an analytical method based on the Cartesian fingertip positions, the mapping produces smooth, continuous, and predictable motions of the robot fingers.

4.4.1 Implementation

The implementation of the virtual object mapping method is significantly more complex than the point-to-point mapping method. However, extensive empirical testing has shown the method enables operators to easily and intuitively control the motions of the robotic hand despite the kinematic differences. A brief summary of the steps is presented here, followed by comparison of mapping results with the point-to-point method. The full details of the virtual object mapping method can be found in Appendix B.

4.4.1.1 Computing the Virtual Object Parameters

The first step is to compute the virtual object parameters based on the operator's fingertip motions. Because the robotic hand is planar and non-anthropomorphic, we must first care-

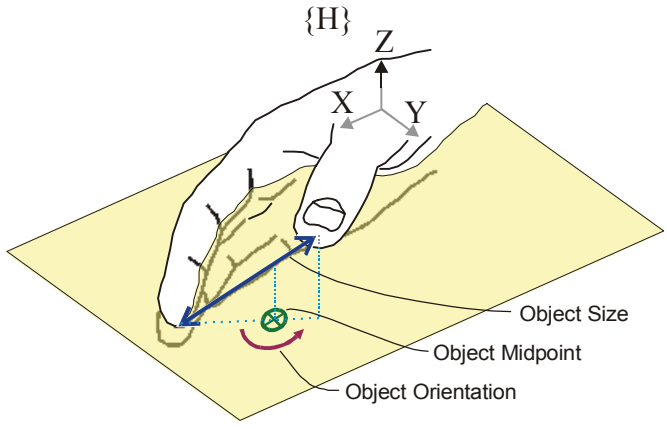

Figure 4-7. The virtual object parameters are defined based on the thumb and index fingertip positions. The size of the object is defined by the distance between the fingertips. The object position is defined by the midpoint between the fingertips and projected on the X-Y plane. The object orientation is also defined by the angle of the projected line between the fingertips.

fully chose what virtual object information should be mapped from the higher dimensional hand space to the lower dimensional space of the robot.

In the general case, the object is defined by seven parameters (size plus six parameters to define position and orientation). However, by projecting the object size, midpoint position, and orientation onto the hand's assumed plane of manipulation (as in Figure 4-7), we reduce the number of parameters to four, which matches the number of degree-of-freedom in the robot hand. Additionally, the computed location of the object midpoint is shifted towards the thumb by a small amount. During method development, shifting the midpoint helped to compensate for natural asymmetric motions of the human hand, which was particularly important because of the symmetric kinematics of the robot hand. The methods for calculating the virtual object parameters based on the operator's finger motions are fully discussed in Appendix B, Section B.1, "Computing the Virtual Object Parameters".

Figure 4-7 illustrates the virtual object parameters as defined by the thumb and index fingertip positions. The Cartesian thumb and index fingertip positions are reduced to the following planar virtual object parameters (representing four values):

- object size
- object midpoint (shifted)
- object orientation

4.4.1.2 Computing Robot Positions from the Virtual Object Parameters

The virtual object mapping method discussed thus far transforms human hand motions into a planar virtual object. For clarity, it is useful to first discuss the method in which the robot fingertip positions are created from the virtual object parameters. With an understanding of how the robot motions are created from virtual object data, it is easier to follow the development of the transformation and scaling methods used to match the workspaces (discussed in the following sections). For the following discussion, assume that the virtual object parameters have been mapped to the robot hand frame such that hand motions have an obvious correspondence with the robot motions, e.g., if the operator opens his/her grasp, the robot creates an opposing grasp.

For free-space motions of the robot fingers, the mapped virtual object parameters are used to compute robot fingertip positions. A simple planar transformation is used to compute the location of each fingertip. The transformation is of the form:

$$ {}^{R}\boldsymbol{p}_i = \underline{R} \cdot \begin{bmatrix} \pm \frac{{}^{R}d}{2} \\ 0 \end{bmatrix} + {}^{R}\boldsymbol{p}_{midpoint} \tag{4.2} $$

where ${}^{R}\boldsymbol{p}_i$ is the location of the i^{th} robot fingertip and $\underline{R}$ represents a rotation matrix that is a function of the mapped virtual object orientation. The variable ${}^{R}d$ represents the mapped virtual object size and ${}^{R}\boldsymbol{p}_{midpoint}$ is the mapped virtual object midpoint. (The specific robot finger, left or right, determines the sign of the ${}^{R}d/2$ term.) See Appendix B, Section B.2, "Computing Robot Positions" for details.

Figure 4-8 graphically illustrates the method used to obtain the robot fingertip positions based on the mapped virtual object. It is important to note that the calculation of the robot fingertips is simplified considerably by assuming the robot maintains a point contact with the virtual object (as opposed to assuming rolling contact between the robot fingers and the virtual object). In other words, the virtual object can also be thought of as a virtual "toothpick" of varying length, position, and orientation in the plane. If the mapped virtual object parameters cause a computed fingertip location that falls outside of the workspace, the nearest achievable position (in Cartesian space) is used.

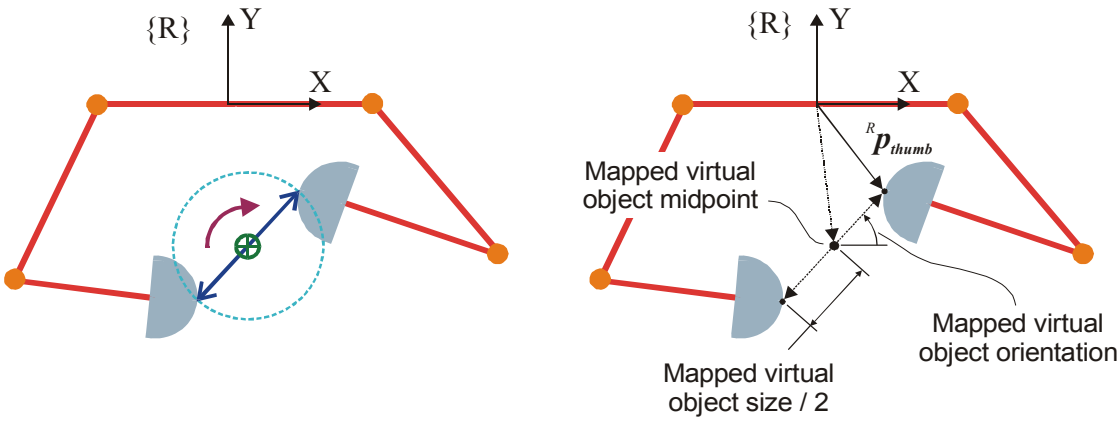

Figure 4-8. Computing robot fingertip positions from virtual object parameters for free-space motion.

When an actual object is detected between the fingers, the virtual object parameters are used to compute set-points for a cooperative object impedance controller (see Chapter 5 for controller details). At the time the object is detected, the controller creates an object model that is continuously updated by the tactile sensors. The changes in the virtual object position and orientation (computed each servo cycle) are added to the initial model to created desired positions and orientations for the actual object. The virtual object size parameter is used to create a desired internal force applied to the object. The amount of force applied is proportional to the difference between the actual object size and the commanded virtual object size.

4.4.1.3 Transformation to the Robot Hand Frame

Before the virtual object parameters (now planar) are used to create robot position or object commands, the parameters must be properly mapped to the robot frame from the hand frame. Our desire is to transform the object information in such a way that the operator can intuitively control the robotic fingers. Figure 4-9 shows the desired correspondence between human hand poses and robot finger configuration.

The transformation of the virtual object parameters is defined such that the comfortable pinch-point of the operator maps roughly to the geometric center of the robot's workspace (Figure 4-9, pose D) and natural human grasping motions are mapped to horizontal grasping motions in the robot's workspace (Figure 4-9, poses A, B, and C). Ideally, human finger motion towards and away from the palm will also map to vertical motion of the robot fingers (Figure 4-9, poses D, E, and F).

Because every operator's hand is slightly different, the virtual object transformation variables are generated for each individual operator and based on data recorded from a few specific hand motions and poses. Automating the calculation of the transformation variables increases the consistency of the mapping for each subject and prevents tedious trial-and-error variable adjustment. The virtual object transformation to the robot frame is a function of the virtual object *orientation*, virtual object *midpoint* data, and a set of variables defined by the recorded hand motions.

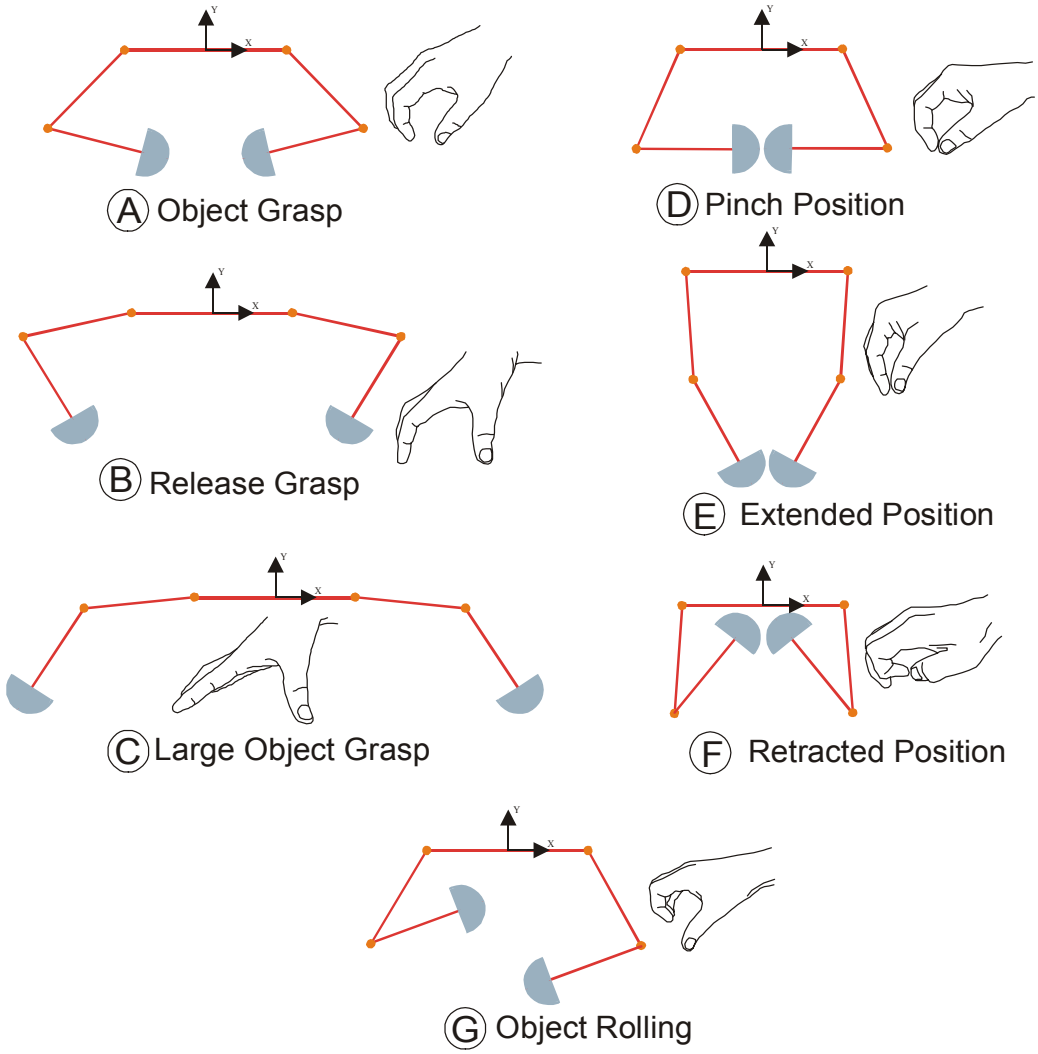

Figure 4-9. Desired correspondence between human hand poses and robotic hand configuration. The motion of enlarging one's grasp is mapped to an increased separation between the robots fingers along the horizontal (poses A, B, and C). The motion of moving one's fingers towards and away from the palm is mapped to vertical motion of the robot's fingertips (poses D, E, and F).

The virtual object *orientation* in the hand frame is mapped to the robot frame using only an angular offset. The offset is based on data recorded for each individual operator and the requires a certain amount of preprocessing. Each operator is asked to open and close his/her grasp as in poses A, B, and C in Figure 4-9 (a natural grasping motion). The data recorded was used to match horizontal grasping motions in the robot frame to human grasping motions (poses A, B, and C in Figure 4-9). The recorded fingertip positions are used to create the virtual object in the hand frame. In a manner similar to the method by which the commanded robot fingertip positions are computed, the virtual object parameters are used to re-create planar index and thumb positions. The modified fingertip data are then used to

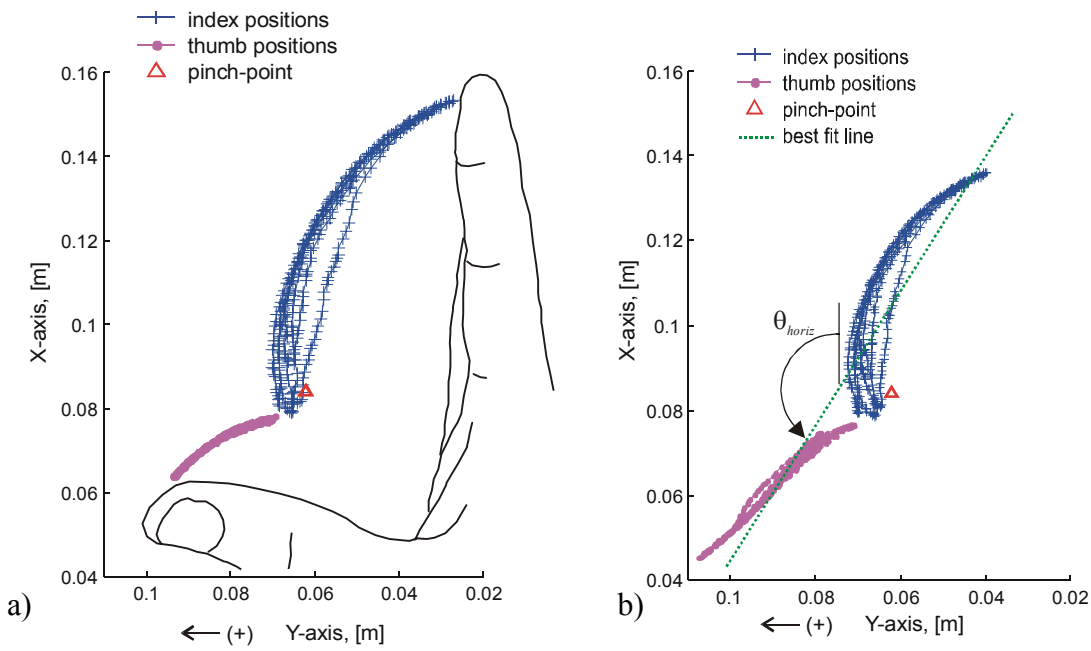

Figure 4-10. a) Planar projection of typical thumb and index fingertip data for an open and close grasping motion. Note: the X-Y frame is rotated for clarity. b) The same thumb and index fingertip data modified based on the virtual object; where the fingertip data are used to create a virtual object and then the virtual object data are used to recreate planar fingertip positions. A best fit line is placed through the data and the angle θ_{horiz} is used to determine the offset for the virtual object orientation.

compute a best-fit line and the angle of the line determines the offset. Figure 4-10a shows a typical planar projection of actual fingertip positions. Figure 4-10b shows the re-created planar fingertip positions based on the virtual object parameters. The graph illustrates the effect the shifted midpoint has on making the projected hand motions more symmetric with respect to the pinch-point location. The details of the calculation of the offset angle can be found in Appendix B, Section B.3, "Transformation to the Robot Hand Frame".

The virtual object *midpoint* data is mapped using a standard frame transformation similar to the point-to-point method. We define the rotation angle such that object motion along the open-and-close grasp line (poses A, B, and C of Figure 4-9) maps to horizontal virtual object motions in the robot frame. The computed rotation angle is based on the previously calculated "horizontal angle," θ_{horiz} (see Figure 4-10b). As a result, any object motion parallel to the grasp motion will map to horizontal motion in the robot frame and any X-Y motions of the virtual object perpendicular to the grasp motion will map to vertical motion in the robot frame (ideally ensuring the motion correspondence shown in poses D, E, and F of Figure 4-9).

However, in applying a simple rotation transform to align the horizontal motions, we noticed that vertical motions created by moving the fingers into and away from the palm (poses D, E, and F of Figure 4-9) were not always perpendicular to the grasping motions (poses A, B, and C). Therefore, in addition to rotating the data into the robot workspace, a skew transformation was added to orthogonalize the these two motions. The skew matrix orthogonalizes natural human grasping motions and finger motions that are into and away from the palm. The general form of the frame transformation for mapping the virtual object midpoint data is as follows:

$$ {}^{R}\boldsymbol{p} \;=\; \underline{S} \cdot {}^{H}_{R}\underline{R} \cdot {}^{H}\boldsymbol{p} + {}^{R}\boldsymbol{p}_{o} \tag{4.3} $$

where ${}^{H}\boldsymbol{p}$ is a vector representing the virtual object midpoint location in the hand frame at each servo cycle. To simplify the implementation, the midpoint vector is computed relative to the operator's natural pinch-point position which is recorded during mapping calibration. The matrix ${}^{H}_{R}\underline{R}$ is a 2×2 rotation matrix from the hand frame to the robot frame, which is defined by θ_{horiz}. The vector ${}^{R}\boldsymbol{p}_{o}$ is an offset vector that maps the operator's pinch-point position to the geometric center of the robot's workspace (as in pose D of Figure 4-9). The matrix $\underline{S}$ represents the skew transformation matrix. Finally, ${}^{R}\boldsymbol{p}$ represents the virtual object midpoint position mapped to the robot hand frame. The transformation variables are based on each operator's individual data recorded during mapping calibration. The open-and-close grasp motion data is used to compute the rotation angle which is defined by θ_{horiz}. Motion data is also recorded to determine pinch-point location. Similar to the determination of θ_{horiz}, additional motion data is recorded and a line fit is used to determine the skew matrix variables. The calculation of each of these variables is presented in Appendix B, Section B.3, "Transformation to the Robot Hand Frame".

4.4.1.4 Workspace Matching

The virtual object size, orientation, and midpoint position are now represented in the robot hand frame. The transformation method discussed thus far addresses the kinematic differences between the human hand and the robot hand. In particular, the natural hand motions are modified to account for the symmetric and non-anthropomorphic design of the robotic hand. However, the robotic hand is physically larger than the human hand allowing for a

much greater workspace. To better utilize the workspace of the robot hand, the virtual object parameters are scaled. To achieve the desired correspondence in poses as shown in Figure 4-9, the virtual object size and virtual object Y-axis midpoint data are scaled.

The virtual object *size* is scaled such that the maximum span of the human hand pose matches the maximum grasp size the robot can achieve (pose C, Figure 4-9). Initially, a simple linear scaling was applied to the virtual object size using the maximum span achieved during the open-and-close grasp motions. Based on normal human fingertip velocities during free-space motions (i.e., not during object manipulation) and average hand sizes, the required gain for matching workspaces was usually on the order of four. A gain this large tends to cause very fast motions of the robot fingers. Yet, operators rarely used the large grasp pose. However, if the gain was decreased (thus preventing an operator from creating the large grasp pose with the robot) operators often became frustrated in attempting to open the robot to a large grasp. To address these problems, a quadratic gain function is used. A quadratic gain function is beneficial because it decreases the robot tip velocities in the range near typical object sizes but still allows an operator to achieve a large grasp. Of course, the disadvantage is that the robot tip velocities are much higher in the large grasp poses, but as mentioned, this pose was not commonly used, especially during object manipulation. For details, see Appendix B, Section B.4, “Workspace Matching”.

The last step in the virtual object mapping is to scale the motions of the virtual object *midpoint* such than an operator can utilize the full workspace of the robot. Through preliminary testing of the mapping method, we found it necessary only to scale the Y-axis (vertical) motion of the virtual object midpoint in the robot frame. A unity gain was applied to the midpoint in the X-axis (along the horizontal). Not scaling the X-axis midpoint motions did not significantly affect the operator’s ability to achieve the desired pose correspondence shown in Figure 4-9 because of the object size scaling.

Scaling of the mapped virtual motion of the midpoint was necessary to achieve the correspondence shown in poses E and F of Figure 4-9. In order to maintain the mapping of the operator’s pinch-point to the center of the robot’s workspace (pose D), the vertical midpoint gain was applied to midpoint location measured relative to the desired pinch-point location.

Since the scaling occurs relative to the desired pinch-point location, it is possible to use different scaling functions for the upper and lower portions of the workspace. During preliminary testing of the mapping method, a quadratic gain for both regions produced the best mapping results. Using a quadratic gain tends to center the operator's motions about the desired pinch-point location and thus in the center of the robot's workspace. The centering effect of the quadratic gain is especially useful during rolling motions (pose G in Figure 4-9) because it was found to be difficult to maintain a fixed rotation axis when virtually rolling an object (even with midpoint shifting). With the quadratic gain, operators could still reach the workspace limits if desired. The quadratic gains were computed based on the minimum and maximum grasp sizes each operator could achieve. See Appendix B, Section B.4, "Workspace Matching" for details.

4.4.2 Method Summary

To achieve an intuitive mapping for each operator is a somewhat complicated multi-step process. The virtual object mapping method steps can be summed up as follows:

- Prior to a telemanipulation session, an operator's hand model is calibrated using the method outlined in [Turner 2001]. Once calibrated, fingertip position data are recorded for three separate hand poses and motions: the natural pinch-point pose (fingertips together), open-and-close grasping motions, and motions towards and away from the palm with the fingertips together. The collected data (as well as predefined robot parameters) are used to computed the necessary transformation and scaling parameters.
- During a telemanipulation session, the sampled human fingertip position data are used to computed the planar virtual object parameters (size, orientation, midpoint) in the hand frame.
- The virtual object orientation is offset such that the natural open-and-close grasp motion in the hand frame maps to horizontal motion in the robot frame.

- The virtual object midpoint vector is computed relative to the natural pinch-point position and then rotated into the robot hand frame. A skew matrix is applied in addition to the rotation transform to orthogonalize grasping motions with motions towards and away from the palm.
- The rotated relative midpoint vector is translated to the desired pinch-point position in the robot's workspace.
- The virtual object size is scaled to roughly match the maximum human hand span with that of the robot.
- The virtual object midpoint location is scaled (relative to the desired pinch-point) to better match the workspace of the robot.
- The mapped and scaled virtual object parameters are used to compute the desired robot fingertip locations for free-space motions or are used to create desired motions of (and forces applied to) a grasped object.

4.4.3 Method Results

Figure 4-11 shows an example of the achievable positions in the robot's workspace based on human hand motions as mapped under the virtual object method (the point-to-point mapping from Figure 4-4 is also included for reference). Notice that the achievable positions for the index finger map very well to the left robot finger workspace. Additionally, and perhaps more importantly, the motion of the robot's right finger (corresponding to the thumb) is greatly expanded compared to the point-to-point mapping. Also, the natural pinch-point position maps to the center of the robot's workspace for a greater manipulation range-of-motion.

Typically some of the achievable positions fall outside of the robot's workspace. This is primarily due to the fact that an operator may not reach his/her finger workspace limits during the recorded motions used for mapping parameter identification. As a result, during a telemanipulation session, the operator may be able to reach positions outside the pre-defined limits. To prevent this from occurring, the gain parameters can be manually modified through a graphical user interface running on the master computer or the simple set of hand motions can be captured again to recompute the mapping parameters. Addition-

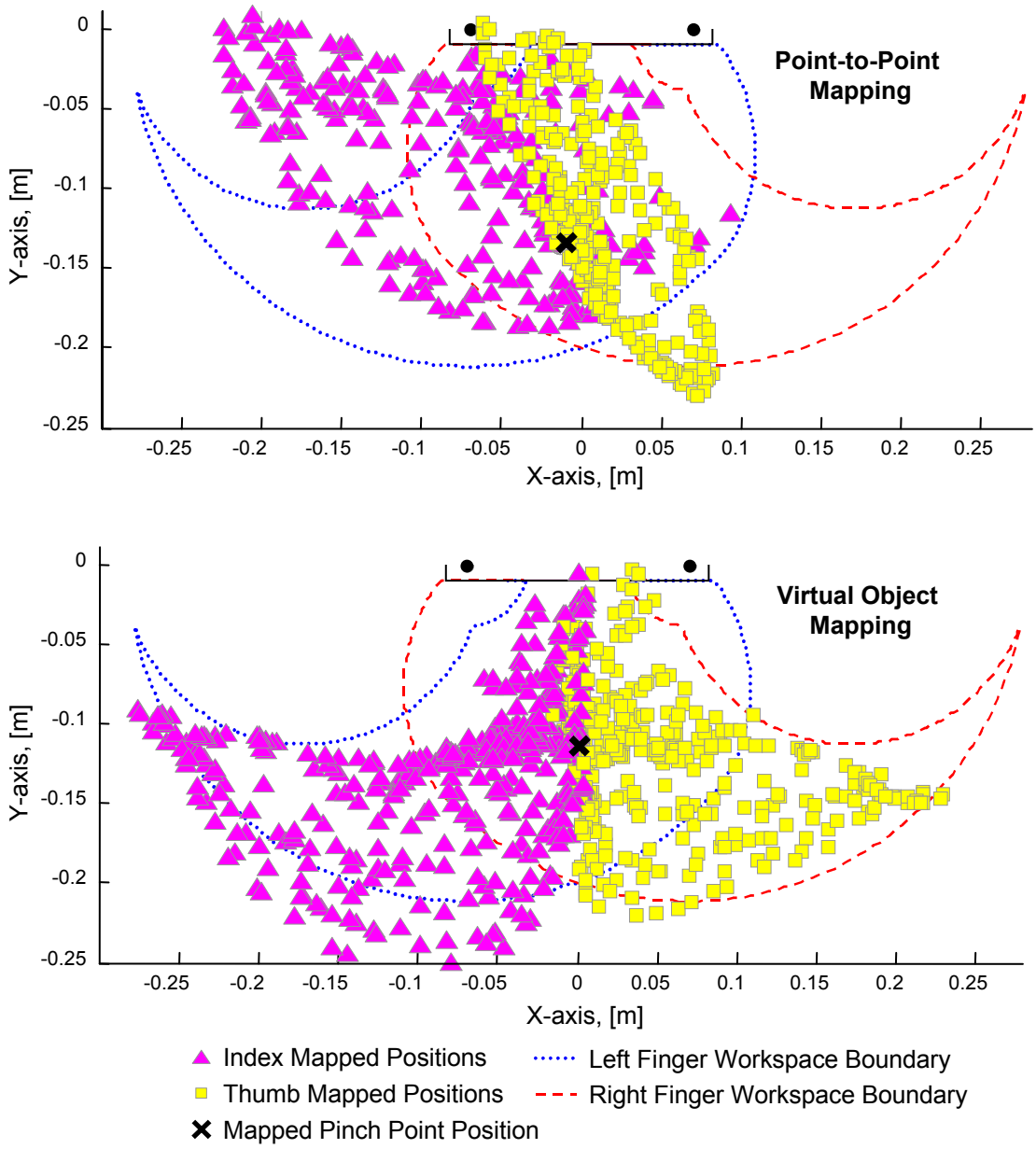

Figure 4-11. A typical plot of achievable positions under the virtual object mapping method. The index and thumb map to the robot's left and right fingers, respectively. Also the mapped pinch point position is indicated.

ally, any positions mapped outside the robot's workspace are mapped to the closest Cartesian position the robot can achieve. However, it is better to tune the gain parameters than rely on the workspace limiting so that operators are not constantly operating at or near the workspace limits.

One caveat that must be observed when using the virtual object mapping method is that the motions of the robot's fingers are coupled. In other words, if the operator only moves one of his/her fingers, both fingers of the robot will move. This is due to the fact that the virtual object is created using both human fingertip positions, and as one of the fingers move, the virtual object size, position, and orientation can change. While the coupled motion did not appear to affect operators during object manipulation, if an operator tried to explore the environment or an object with a single finger the coupled motion was very noticeable (and made single finger exploration difficult).

Because of the large difference in physical size between the robot hand and the human hand, it was necessary to amplify the motion of the human hand as described in section 4.4.1.4, Workspace Matching. Depending on the operator's range-of-motion and mapping parameters, the motion scaling gains could allow the operator to achieve high velocities at the robot fingertips possibly causing damage to the fingertips. However, this did not cause any significant problems because operator's tend to use slow finger motions when manipulating an object.

4.4.4 Extensibility

While the virtual object mapping concept has been demonstrated using a particular robotic hand, the mapping method is a general technique and is extensible to most planar symmetric robot hands. Utilizing the steps presented in Section 4.4.2 and by modifying the mapping parameters, particularly the scaling functions, the motions of the human hand can be mapped to a different robot workspace. If a robot hand has additional degrees-of-freedom per finger, there is not a unique configuration to achieve a desired virtual object position and orientation. The redundancy may be used to control contact location, extend the reachable workspace or other optimizations.

4.5 Conclusion

In summary, the virtual object based mapping method provides a solution to the problems associated with the kinematic differences between human hands and planar robot hands. Using the three-dimensional fingertip data to define a virtual object yields additional information about the intent of the manipulation compared to a simple planar projection of fin-

gertip positions. With very little training, operators were easily able to grasp, manipulate, and release objects using the robotic hand.

5 Shared Control

This chapter describes a shared control framework used for dexterous telemanipulation. As discussed in Chapter 2, shared control represents a middle ground between supervisory control and traditional bilateral control in which the remote system can exert control over some aspects of the task while the human operator maintains access to low-level forces and motions. The operator has the ability to control, and receive feedback from, the remote teleoperator at a low level (e.g., position commands and force feedback). Additionally, the operator has can supply high-level commands utilizing the supervisory aspects of the controller. The shared control two-level hierarchy supplies the operator with a greater sense of telepresence compared to a purely supervisory system while overcoming some the limitations associated with direct telemanipulation systems (such as time delays and limitations in master feedback fidelity).

The approach presented in this chapter is based on combining direct telemanipulation commands from a human operator with those from a semi-autonomous dexterous controller to guide the robot and its hand. The direct commands from the operator are generated using an instrumented data glove. Fingertip level force feedback device is used as the primary form of feedback to the operator. Utilizing a human hand based input/feedback system taps into the operator's natural manipulation ability. The semi-autonomous controller utilizes force and tactile sensors and the established theory of dexterous manipulation, including the control of internal grasp forces and the kinematics of rolling contact. Additional visual and audible feedback channels exist to aid the operator in controlling the telemanipulator in conjunction with the shared control system. The concept is shown in Figure 5-1. The shared control framework provides the foundation for the development of a dexterous telemanipulation system capable of improving performance compared to a traditional bilateral telemanipulation.

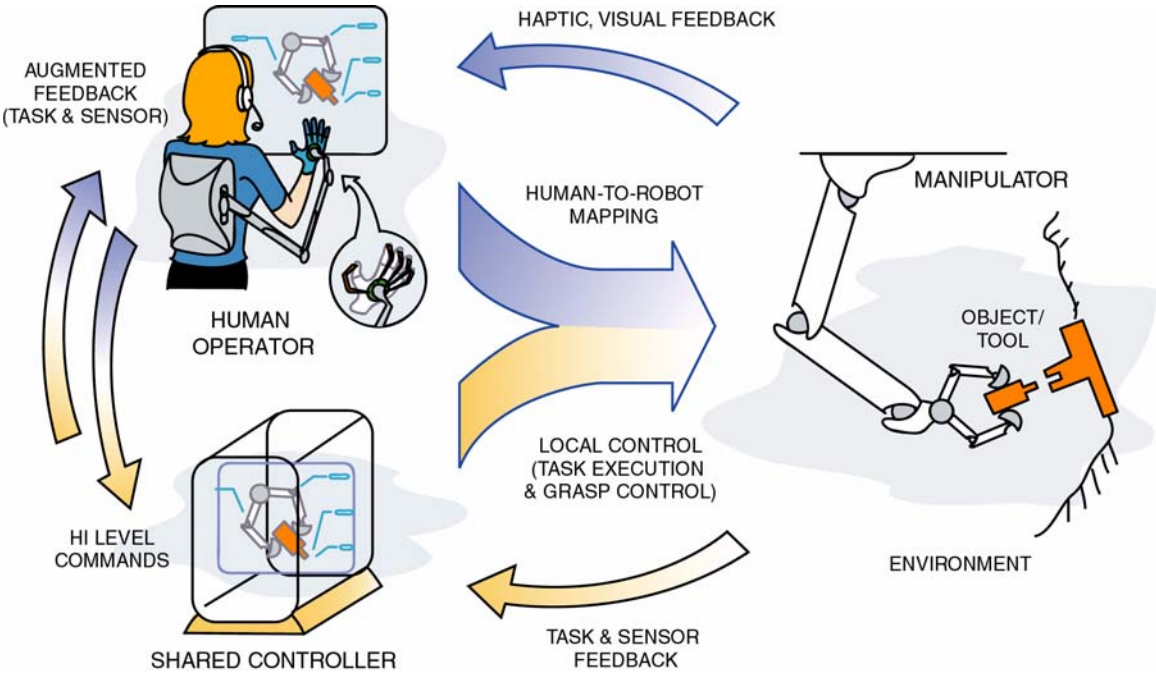

Figure 5-1. The shared control concept for dexterous telemanipulation in an assembly task. The essence of shared control is in the combination of operator commands, both high level and low level, with the commands from a semi-autonomous controller. Haptic, tactile, and visual information is fed back to the operator from the remote manipulator. Additionally, augmented feedback of direct quantities (e.g., force) and indirect quantities (e.g., grasp stability) can be supplied to the operator from the shared controller. The shared controller uses task and sensor feedback with a dexterous manipulation control foundation for remote task execution and grasp control. High level commands from the operator are also used for supervisory control over the shared controller.

The operator can interact with the remote robot and its dexterous hand at multiple levels. At the lowest level, the operator can control the individual finger motions of the dextrous hand to explore or gently grasp an unknown object. At the highest level, the operator can direct the controller to change control modes, e.g., from independent finger exploration to cooperative object manipulation.

At an intermediate level, the human operator and remote manipulator share control at the servo level during task execution. At this level the operator imparts motions to a virtual object. The virtual object parameters are mapped to a real object by the controller and the robot is responsible for coordinating its fingers to apply the appropriately scaled motions and forces on the real object to achieve the desired effect (see Chapter 4).

There are advantages to letting the robot hand take over force regulation and object manipulation when the task is sufficiently simple and well defined. By providing local control of forces, stiffness, and motions within the hand, the robot allows the human supervisor

to focus on the task itself, concentrating on the desired motions and behavior of the grasped object or tool. Time delays and limitations in the accuracy of haptic feedback through the master become less detrimental because master commands are no longer essential to prevent unwanted slips or damaged objects. However, the operator still maintains the ability to override the dexterous controller to release the object or grasp it more tightly if necessary.

The following sections of this chapter describe the methods used in our approach to implement a shared control framework for dexterous telemanipulation. The control of the slave manipulator is first discussed. Building upon the semi-autonomous dexterous manipulation capabilities, the shared control monitoring and intervention capabilities are described. Finally, multi-modal (i.e., force, vibration, acoustic, and visual) feedback used for the display of both direct and indirect quantities is described.

5.1 Slave Control Framework

The control laws for the slave manipulator lay the foundation for the implementation of a shared control telemanipulation system. The control laws were initially developed for traditional position-force bilateral telemanipulation [Turner et al. 2000], wherein, set-points generated by the master system are mapped to desired positions for the robotic fingers and interaction forces due to environmental contact are fed back to the operator. We build upon this bilateral framework towards the goal of developing a shared control system.

We are interested in both independent control of the fingers and coordinated control for manipulating grasped objects. In each case, the control law is based on Khatib's [1987] operational space formulation and Hogan's [1985a, 1985b] impedance control laws.

Independent fingertip control may be used for such tasks as single finger exploration and free space motion[1]. Our remote manipulator is a two fingered, two degrees-of-freedom per finger, robotic hand. However, because the kinematics differ significantly from those of the human hand, a direct joint to joint mapping is not possible. Therefore, the Cartesian end points of the human fingers, as measured by the instrumented glove, are trans-

1. Due to the human-to-robot mapping method implemented, the motions of the index and thumb may not be independent of each other. However, once the desired tip point positions are received by the slave, the controlled variables for tip position are independent for each finger.

formed into desired positions for the robotic fingers (as described in Chapter 4). Operational space control allows us to express the dynamics of the manipulator (in this case a finger) at the point of interest, namely, the fingertip. Doing so allows us to specify the endpoint impedance of the robotic fingers for free space motion and/or environment interaction. Under these circumstances it is possible to develop a shared control system that allows the operator and remote manipulator to share control in terms of slave impedance modification, based on sensor information provided to the robotic hand and expressed in terms of a coordinate frame embedded at the fingertip.

However, our focus is on the development of a dexterous telemanipulator capable of shared control of a grasped object. While the operator is capable of object manipulation by controlling the fingers independently, development of an object centered control formulation better supports semi-autonomous control for monitoring and intervention.

The implementations of the control laws for independent and cooperative modes are discussed below. The methods are based upon established results in the literature but are included for completeness and to provide the necessary background for describing the shared control capabilities.

5.1.1 Independent Control

Given our desire to work in Cartesian space and our concern with the behavior of the fingertip, operational space control is a natural choice. The operational space formulation provides a basis that allows us to modify the dynamic response or impact force at the tips of the robotic fingers (see Figure 5-2).

The general equations of motion representing the actual fingertip dynamics are:

$$\underline{M}_a(\boldsymbol{x}) \cdot \ddot{\boldsymbol{x}} + \mu(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \boldsymbol{p}(\boldsymbol{x}) = \boldsymbol{F} + \boldsymbol{F}_{ext} \tag{5.1}$$

where $\underline{M}_a(\boldsymbol{x})$ represents the configuration-dependent mass matrix for our two link finger, $\boldsymbol{x}$ is a vector denoting generalized coordinates specifying position and orientation in a given reference frame (in our case a 2×1 vector of positions for planar motion), $\mu(\boldsymbol{x}, \dot{\boldsymbol{x}})$ is a vector expressing the centrifugal and Coriolis forces, and $\boldsymbol{p}(\boldsymbol{x})$ describes the gravitational forces. For clarity, the variable dependence on $\boldsymbol{x}$ will be dropped in subsequent equa-

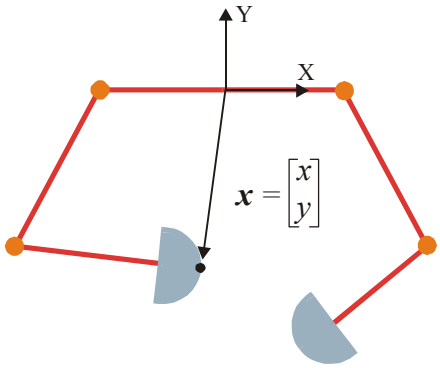

Figure 5-2. For our two-fingered robot, the desired Cartesian fingertip positions are computed from the human-to-robot mapping method. The operational space formulation provides a framework for simplifying the control of the end-point position of the serial linkage. Additionally, it is possible to decouple the control of the tip position from the configuration-dependent dynamics of the linkage through feedforward techniques.

tions except where needed for emphasis. The vector $\boldsymbol{F}$ represents the forces applied to the system by the actuators and $\boldsymbol{F}_{ext}$ describes any external forces acting on the fingertips. The terms $\underline{M}_a$, μ, and $\boldsymbol{p}$ are functions of $\boldsymbol{x}$ but are formed from expressions in joint space and transformed to operational space using the Lagrangian formalism and by equating the two quadratic forms of kinetic energy. For further details on the equations of motion and the formulation of these expressions see Appendix C and [Khatib 1987].

Utilizing this approach we can modify the robotic system and transform the actual dynamics into a desired system of the form:

$$\underline{M}_d \cdot \ddot{\boldsymbol{x}} = \boldsymbol{F}^* + \boldsymbol{F}_{ext} \tag{5.2}$$

where $\underline{M}_d$ represents the desired mass matrix of the system and the vector $\boldsymbol{F}^*$ describes the selected control law applied to the desired system. Various controls can be employed, such as, potential fields, velocity limited goal positioning, trajectory tracking, and impedance control [Khatib 1987].

To achieve the desired system in Equation 5.2, we must first develop the control structure for $\boldsymbol{F}$ shown in Equation 5.1. If we solve for the acceleration, $\ddot{\boldsymbol{x}}$, in Equation 5.2 and substitute the resulting expression into Equation 5.1, we obtain:

$$\boldsymbol{F} = (\hat{\underline{M}}_a \underline{M}_d^{-1}) \cdot \boldsymbol{F}^* + (\hat{\underline{M}}_a \underline{M}_d^{-1} - \underline{I}_d) \cdot \hat{\boldsymbol{F}}_{ext(sensed)} + \hat{\mu} + \hat{\boldsymbol{p}} \tag{5.3}$$

where variables denoted with a circumflex are estimates of the actual quantities as obtained by system models (or sensor measurements in the case of $\hat{\boldsymbol{F}}_{ext(sensed)}$), and $\underline{I}_d$ is the identity matrix of size $\underline{M}_d$.

Because we wish to control the behavior of the fingertip, an impedance control law is implemented for $\boldsymbol{F}^*$. Based upon methods originally developed by Hogan [1985a, 1985b], the impedance method specifies the Cartesian position-force relationship for a given system. The expression for the impedance force can be arbitrarily complex and non-linear; however, we wish to create a simple second order linear dynamic system. Intuitively, the desired positions (and orientations) serve as an equilibrium point for a virtual spring-and-damper attached to the fingertip. An error between the desired and actual position creates an impedance force in the virtual spring that is applied to the point of attachment. The interaction force with an object in the environment is controlled by simply setting the virtual equilibrium point inside the object. The impedance of the system can be modified by gains which control the stiffness and damping. The resulting formulation for $\boldsymbol{F}^*$ is given by:

$$\boldsymbol{F}^* = \underline{K}_p \cdot (\boldsymbol{x}_d - \boldsymbol{x}) + \underline{K}_v \cdot (\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}) \tag{5.4}$$

in which $\boldsymbol{x}_d$ and $\dot{\boldsymbol{x}}_d$ correspond to the desired end-point position and velocity. The matrices, $\underline{K}_p$ and $\underline{K}_v$, represent the proportional and derivative gains for impedance control in the directions (and/or orientations) specified by $\boldsymbol{x}$.

To formulate our expression for the actuator force, $\boldsymbol{F}$, we must not only specify $\boldsymbol{F}^*$ but also the desired mass matrix, $\underline{M}_d$. If we choose our desired system to have the same mass as the actual system (i.e., $\underline{M}_d = \underline{M}_a$), then our equation for the actuator force, from Equation 5.3, reduces to:

$$\boldsymbol{F} = \boldsymbol{F}^* + \hat{\mu} + \hat{\boldsymbol{p}} \tag{5.5}$$

We can then transform this desired operational space actuator force into joint torques using:

$$\tau = \underline{J}^{\mathrm{T}}(\boldsymbol{q}) \cdot \boldsymbol{F} \tag{5.6}$$

with $\underline{J}^{\mathrm{T}}(\boldsymbol{q})$ as the joint-position-dependent Jacobian matrix and τ as the vector of applied joint torques.

Notice that the external force term in Equation 5.5 drops out and relieves the need to sense the disturbance forces. In essence, the operational space equations reduce to Hogan's formulation [Hogan 1985b]. While this formulation does allow us to develop and control our system in operational space, the system's free-space behavior will be configuration dependent, as given in:

$$\underline{M}_a(\boldsymbol{x}) \cdot \ddot{\boldsymbol{x}} = \boldsymbol{F}^* + \boldsymbol{F}_{ext} \tag{5.7}$$

At steady state, the impedance forces, $\boldsymbol{F}^*$, will balance the applied external forces. However, inertial forces are not compensated for and can lead to tracking errors due to the dependence of the mass matrix on configuration.

If we choose our desired system to have a unit mass (i.e., $\underline{M}_d = \underline{I}_d$, where $\underline{I}_d$ is the identity matrix of size $\underline{M}_d$), then we can decouple the dynamics of the system from the control law using feed-forward estimates, resulting in the expression:

$$\boldsymbol{F} = \hat{\underline{M}}_a \cdot \boldsymbol{F}^* + (\hat{\underline{M}}_a - \underline{I}_d) \cdot \hat{\boldsymbol{F}}_{ext(sensed)} + \hat{\mu} + \hat{\boldsymbol{p}} \tag{5.8}$$

By pre-multiplying the impedance force by the configuration dependent mass matrix, $\hat{\underline{M}}_a$, were are providing a feed-forward term, in additional to the centrifugal, Coriolis, and gravitational terms, to compensate for the effects of the changing system mass. Intuitively, the $(\hat{\underline{M}}_a - \underline{I}_d) \cdot \hat{\boldsymbol{F}}_{ext(sensed)}$ term first cancels out the disturbance force and then adds in the scaled force necessary to make the system behave dynamically like a unit mass in the presence of an external force. Substituting this expression into Equation 5.1, and assuming that

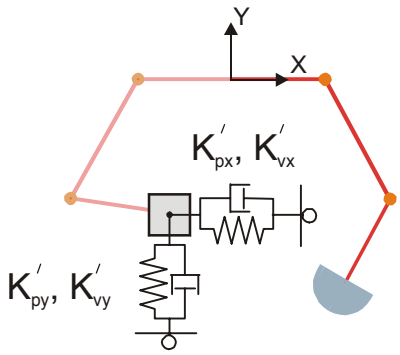

Figure 5-3. Using the operational space formulation with dynamic decoupling, the end-effector point of interest becomes equivalent to a single unit mass. Under the impedance control law, a virtual spring and damper are used to control the tip point position in Cartesian space. The the desired position specifies the equilibrium points for the springs.

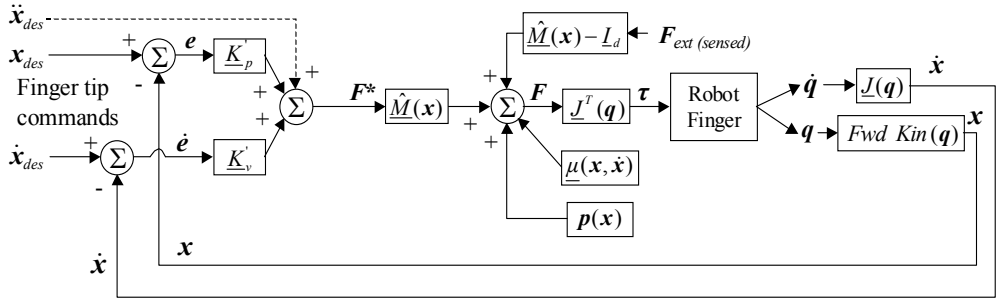

Figure 5-4. Control diagram for operational space impedance control for a dynamically decoupled system in which the desired mass is a unit mass. The differences between the desired position/velocity and the measured position/velocity are multiplied by the gain matrices, which determine the second order response of the system and the interaction stiffness. To compensate for the configuration dependent mass, the impedance force, $\boldsymbol{F}^*$, is multiplied by the operational space mass matrix. Additionally, feedforward terms for centrifugal, Coriolis, and gravity forces are added. To ensure the system behaves like a unit mass in the presence of external forces, the measured force is a function of the mass matrix. Finally, the joint torques are computed from the control forces applied to the end-effector using the Jacobian relationship.

our estimates are perfect, we obtain the following equation of motion describing a unit-mass system:

$$\underline{I}_d \cdot \ddot{\boldsymbol{x}} = \boldsymbol{F}^* + \boldsymbol{F}_{ext} \tag{5.9}$$

The gain matrices in Equation 5.4 now specify the impedance behavior for the unit-mass system. Figure 5-3 illustrates the operational space framework for a desired unit-mass controlled with an impedance law. The desired tip positions are created by the human-to-robot virtual object mapping method. Figure 5-4 shows a the block control diagram for individual fingertip control based on the impedance control laws and unit-mass decoupling as formulated in Equations 5.4 and 5.8.

When implementing the operational space control formulation for a telemanipulation system there are several important considerations. In Equation 5.8 we can see that the applied control force to the fingertip is a function of the sensed external forces at the operational point. Unfortunately, this also implies that un-modeled sensor dynamics can negatively affect system performance. Furthermore, if our actual mass, $\hat{\underline{M}}_a$, is relatively smaller than a unit mass, the product of $\hat{\underline{M}}_a \underline{M}_d^{-1}$ will also be smaller than one. For our robot hand, the mass at the operational point in a typical manipulation pose is on the order of 0.1 Kg. If the product of $\hat{\underline{M}}_a \underline{M}_d^{-1}$ is small, then Equation 5.3 is approximated by:

$$\boldsymbol{F} = -\hat{\boldsymbol{F}}_{ext(sensed)} + \hat{\mu} + \hat{\boldsymbol{p}} \tag{5.10}$$

in which case, errors in $\hat{\boldsymbol{F}}_{ext(sensed)}$ have a significant effect. Given our system and force sensors, the effect was significant enough to cause visible oscillations during freespace tracking motions. Additionally, by choosing a unit mass for $\underline{M}_d$, we are effectively increasing the mass of our slave system. As discussed in Chapter 2, the master-slave mass ratio (m_m / m_s) places a fundamental limitation on "position-force" telemanipulation systems. As the mass of the slave becomes larger than the mass of the master, the force reflection ratio must be attenuated or the stiffness of the slave must decreased to maintain stability during contact with stiff environments. In our system, the mass of the master system (the human finger and exo-skeleton device attached to the finger) is significantly smaller than actual mass of the slave system. It may seem appropriate to simply decrease the stiffness of the unit mass system to gain some stability; however, this may cause a decrease the achievable motion bandwidth of the fingertip. Thus in choosing the desired mass to equal a unit mass penalizes the performance of the system in terms of both an increase in apparent mass and the inclusion of the force sensor dynamics into the feedback loop.

For these reasons, the external disturbance force was removed from the feedback path (setting $\boldsymbol{F}_{ext(sensed)}$ to zero in Equation 5.8); yielding the system equation: $\ddot{\boldsymbol{x}} = \boldsymbol{F}^* + \underline{M}_a^{-1}\boldsymbol{F}_{ext}$. This will cause errors in the impedance force during contact (effectively reducing $\underline{K}_p$ if $\underline{M}_a$ is approximately less than one), but will not adversely affect the freespace performance of the system. However, the main focus of this thesis is on shared control during object manipulation. Since the robot switches to a cooperative control law for object manipulation, removing the sensed forces from the feedback path does not adversely affect overall performance.

5.1.2 Object Impedance Cooperative Control

The goal of a cooperative controller is to stably manipulate a grasped object. To develop a shared control telemanipulation system, the method used must support direct commands from the master system and be capable of semi-autonomous manipulation. The semi-autonomous framework allows the system to monitor the actions of the operator and intervene if necessary.

The control method described here draws heavily from previous work in autonomous dexterous manipulation. An important aspect of the cooperative control law is the grasp transformation matrix (also know as the grasp matrix). The grasp matrix provides a mathematical construct to determine the necessary fingertip forces required to create a desired net force on a grasped object [Mason and Salisbury 1985]. The force on the object is computed as follows:

$$ {}^{o}\boldsymbol{F}_{obj} = \underline{G}^{-T} \cdot {}^{o}\boldsymbol{F}_{tip} \tag{5.11} $$

where ${}^{o}\boldsymbol{F}_{obj}$ is a vector, expressed with respect to the object, of external and internal forces and moments on the object, and ${}^{o}\boldsymbol{F}_{tip}$ is a vector of fingertip forces and moments, the size of which is determined by the number of fingers and the type of contact. The inverse of the transpose of the grasp matrix, $\underline{G}^{-T}$, may be found geometrically and is based on vectors from the object's reference frame to the contact points. See [Mason and Salisbury 1985] and Appendix D for further details. The transpose of the grasp matrix, $\underline{G}^{T}$, is often referred to as the "forward grasp transform." Similar to the force-velocity duality of the standard Jacobian relation, one can derive a relationship between the velocity of the contact points and the velocity of the grasped object:

$$ {}^{o}\dot{\boldsymbol{x}}_{obj} = \underline{G} \cdot {}^{o}\dot{\boldsymbol{x}}_{tip} \tag{5.12} $$

Again, both vectors are expressed in the object frame.

Determination of the internal force on the object must be computed using the force sensor data. As advocated by Yoshikawa and Nagai [1991], care must be taking when decomposing the measured forces into manipulation and internal forces, i.e., when forming $\underline{G}^{-T}$. In the two finger manipulation model, their method is based on choosing the internal force from the minimum of the forces projected along the line of contact. This information is used to modify the grasp matrix and ensures proper decomposition if the forces at the fingertips are unequal and the object is accelerating (see Appendix D).

In a manner similar to the independent fingertip control discussed in the previous section, the goal is now to control the dynamic behavior, or impedance of the object during manipulation and environmental interactions. The object impedance control law follows

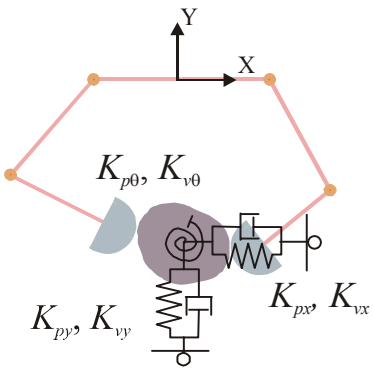

Figure 5-5. Object impedance framework concept for an object grasped by a planar two-fingered manipulator. The Cartesian stiffness and damping of the object can be specified along with the rotational stiffness and damping. The equilibrium points of the virtual spring-dampers are determined by the desired position of the object based on the human-to-robot mapping.

the approach developed by Schneider and Cannon [1992] and implemented by Hyde [1995]. In this approach, desired object positions are equilibrium points for a virtual spring-damper attached to the object, while desired orientations are equilibrium points for a virtual torsional spring-damper (see Figure 5-5). It is also possible to modify the apparent inertia of the object and to attach virtual spring-and-dampers to an arbitrary point on the object. It is important to note that the forces computed by the impedance control and the internal force on the object are independent of each other. Therefore, a separate control law can be applied to the internal force.

Object tracking during manipulation is partially based on work by Maekawa et al. [1995]. The authors present an algorithm for manipulation with rolling. Given tactile sensors at the fingertips to detect contact location, their control system determines the velocities of the fingers required to move the object. The advantage of this method lies in its ability to manipulate unknown objects, thus removing the need for having an *a priori* object model. The method presented here takes from their method the concept of summing differential motions computed by the grasp transform to update the object location and the use of tactile sensors to update the contact points on the object.

5.1.2.1 Implementation

The cooperative control loop begins with the assumption that an object is held in the robot's fingers. The contact-based mode switching and transition periods are discussed in the following section. At the onset of object detection, the object tracker assigns a reference frame

to the object and establishes contact vectors in this frame based on the hand configuration and the fingertip contact locations obtained from the tactile sensors. From these data the inverse-transpose grasp matrix $\underline{G}^{-T}$ is calculated (see Appendix D for details).

The next step is to use the object impedance framework to compute the external force on the object based on the current position and the operator's desired position. In the full implementation, the object impedance controller can modify the inertia and center-of-mass of the object [Schneider and Cannon 1992]. This capability could be used for advanced task level shared control techniques (such as providing the object with a remote center-of-compliance for an assembly task); however this is beyond the scope of this thesis. As in the implementation of Hyde [1995], choosing the desired inertia to be equal to the actual object inertia serves a dual purpose. This choice eliminates the need to sense disturbance forces on the object. Additionally, it is not necessary to have *a priori* knowledge of the object. Replacing $\boldsymbol{x}$ in Equation 5.4 with $\boldsymbol{x}_{obj}$ forms the virtual spring-damper impedance controller. The desired object position can be determined from the operators' commands or can be modified by the shared controller (see following section for details). $\boldsymbol{F}^*$ now represents the applied force and moments on the object in the world frame.

To determine the necessary fingertip forces and moments based on the desired impedance force and internal force commands, the following equality is used:

$$ {}^{w}\boldsymbol{F}_{tip} = {}^{w}_{o}\underline{R} \cdot \underline{G}^{T} \cdot \begin{bmatrix} {}^{o}_{w}\underline{R} \cdot {}^{w}\boldsymbol{F}^{*} \\ \boldsymbol{F}_{int} \end{bmatrix}. \tag{5.13} $$

Notice that it is necessary to rotate the force/moment vectors into the object reference frame due to the formulation of the grasp transform in the object frame. For our planar two-fingered manipulator, the equation above reduces to:

$$ \begin{bmatrix} f_{x1} \\ f_{y1} \\ f_{x2} \\ f_{y2} \end{bmatrix} = \begin{bmatrix} {}^{w}_{o}\underline{R} & \underline{0} \\ \underline{0} & {}^{w}_{o}\underline{R} \end{bmatrix} \cdot \underline{G}^{T} \cdot \begin{bmatrix} {}^{o}_{w}\underline{R} & \underline{0} \\ \underline{0} & \underline{I}_2 \end{bmatrix} \cdot \begin{bmatrix} f_x \\ f_y \\ m_z \\ f_{int} \end{bmatrix} \tag{5.14} $$

where f_{xi} and f_{yi} specify the fingertip forces for the i^{th} finger ($^{w}\boldsymbol{F}_{tip}$), the $\underline{R}$'s are 2×2 rotation matrices from the object to the world frame and its inverse, and $\underline{0}$ and $\underline{I}_2$ are the 2×2 zero and identity matrix, respectively. The vector on the right represents the impedance forces applied to the object and the commanded internal force, f_{int} (see Figure 5-6).

A simple proportional-integral control law is used to closed the loop on the desired internal force (f_{int} in Equation 5.14) as compared to the measured internal force. The desired internal force can be either be commanded by the operator or by the shared controller (see Section 5.2.1 for details). Following Yoshikawa and Nagai's [1991] method for two fingers, the measured internal force is the minimum of the dot product of the fingertip forces and a vector along the line of contact. In this convention the formulation of the grasp transform is dependent upon the which finger has the minimum force, thus actually creating two grasp transforms, a left map $\underline{G}_L^{-T}$ and right map $\underline{G}_R^{-T}$ (see Appendix D for the complete form).

The computed fingertip forces can now be combined with any desired feed-forward terms for each finger, such as robot finger gravity compensation. Because manipulation occurs at relatively low speeds, the feed-forward term for the contact point acceleration is dropped (see [Hyde 1995] and [Schneider and Cannon 1992] for full implementations). Also, the centrifugal force estimate is assumed small and therefore ignored. The following equation represents the forces to be applied by the actuators using the Jacobian transform

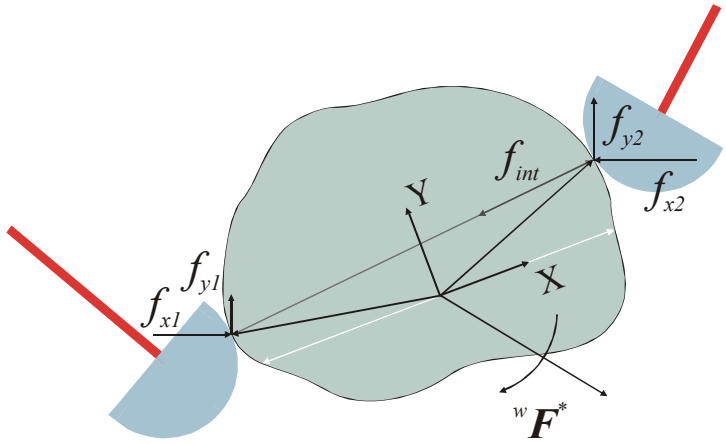

Figure 5-6. Forces applied to an object grasped by a planar two-fingered manipulator. A reference frame is attached to the object and the contact vectors extend from the origin to the contact locations. The contact location vectors are updated based on fingertip motions and the tracked object position. The grasp matrix is then updated based on the contact vectors. The internal force lies along the line between the two contact points. The impedance force, $\boldsymbol{F}^{*}$, is applied at the origin of the object's reference frame and is based on the desired object motion. The reference frame location on the object is based on the initial contact locations, shown in white.

(Equation 5.6) for each finger (two fingers in this case) based on the fingertip forces and gravity compensation:

$$\boldsymbol{F}_{1,2} = {}^{w}\boldsymbol{F}_{tip} + \begin{bmatrix} \hat{\boldsymbol{p}}_1 \\ \hat{\boldsymbol{p}}_2 \end{bmatrix} \tag{5.15}$$

The next step is to update the object position and orientation so the correct impedance force is applied to the object. Using the tactile contact location information and the change in robot position, the grasp transform can be used to compute changes in object position. It is assumed that the object undergoes pure rolling without slip and maintains a single continuous contact point[2] at each finger. The expression for the change in object position, ${}^{w}\Delta\boldsymbol{x}_{obj}$, is given by first computing the object velocity based on the fingertip velocity, ${}^{w}\dot{\boldsymbol{x}}_{tip}$. For the planar two-finger case, we have:

$${}^{w}\dot{\boldsymbol{x}}_{obj} = \begin{bmatrix} {}^{w}_{o}\underline{R} & \underline{0} \\ \underline{0} & \underline{I}_2 \end{bmatrix} \cdot \underline{G} \cdot \begin{bmatrix} {}^{o}_{w}\underline{R} & \underline{0} \\ \underline{0} & {}^{o}_{w}\underline{R} \end{bmatrix} \cdot {}^{w}\dot{\boldsymbol{x}}_{tip} \tag{5.16}$$

multiplying the result by T_s, the sample time, yields:

$${}^{w}\Delta\boldsymbol{x}_{obj} = T_s \cdot {}^{w}\dot{\boldsymbol{x}}_{obj} \tag{5.17}$$

At $t = 0$, when the object is grasped, the initial object location with respect to the world is defined and the reference frame for the object is assigned; both are based on the contact locations at the fingertips. Summing ${}^{w}\Delta\boldsymbol{x}_{obj}$ at each servo cycle produces the current object position relative to the position assigned at the initial grasp. The object position is computed as follows:

$${}^{w}\boldsymbol{x}_{obj} = {}^{w}\boldsymbol{x}_{obj}\Big|_{t=0} + \sum_{n=1}^{N} {}^{w}\Delta\boldsymbol{x}_{obj}\Big|_{nT} \tag{5.18}$$

2. While our robot is planar, for the real world case the contact condition is a line of contact in which the line is perpendicular to the plane of motion.

With the new object position information and the robot fingertip positions, the contact vectors within the object are computed. This information is then used to update the grasp transform for the next cycle's calculations.

The *desired* position of the object and the *desired* internal force are computed from the mapped virtual object parameters (refer to Chapter 4 for details). To determine the desired position, the mapped virtual object planar position and orientation changes are used to update the desired position of the grasped object; as in to Equation 5.18 but replacing $\boldsymbol{x}_{obj}$ with $\boldsymbol{x}_{obj,des}$, and with the initial desired object position being coincident with the initial object position (i.e., ${}^{w}\boldsymbol{x}_{obj,des}\big|_{t=0} = {}^{w}\boldsymbol{x}_{obj}\big|_{t=0}$). The mapped virtual object size is compared to measured object size and the difference is multiplied by a gain to compute the desired internal force:

$$f_{int,des} = K_f \cdot (x_{int,obj} - x_{int,des}) \tag{5.19}$$

where K_f is the internal force proportional gain determined empirically to ensure stable object interaction. A positive desired internal force is created when the operator commands a desired object size less than the actual object size.

Using the desired object size, position, and orientation obtained from the mapping method and the cooperative control framework, there is a direct correlation between an operator's virtual object motions and the actual object manipulated by the robot. In this

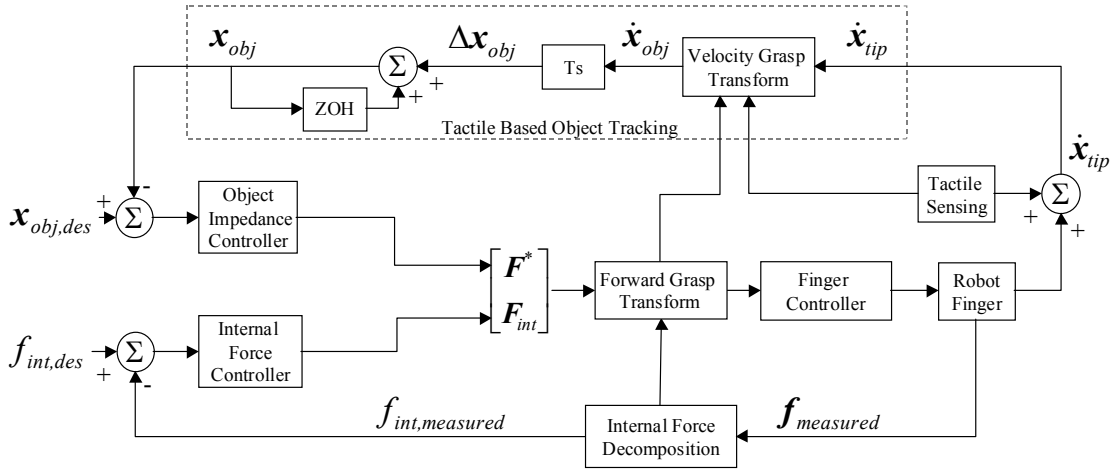

Figure 5-7. Control diagram for the object impedance cooperative control framework. Separate laws are used for controlling the impedance force and internal force on the object. The forces on the object are concatenated and multiplied by the forward grasp transform to determine the appropriate fingertip forces. A tactile based object tracking method is used to update the modeled position of the object and the contact vectors within the object.

way, if the operator commands the object to translate and rotate a specified amount, the actual object will undergo equivalent motions. This result differs from applying the virtual object motion data to the fingers under independent control in that, for a given virtual object motion, the actual object motion will now be dependent upon the object's shape.

Figure 5-7 depicts the control system diagram for the cooperative control method described above.

5.1.3 Transitions

The switch to and from independent control and object impedance cooperative control is completely determined by the tactile contact location sensors. If contact is detected on both fingers, the shared controller automatically switches to the cooperative control law. If during manipulation, a single finger loses contact with the object the control mode is switched back to independent control.

To ensure smooth transitions from independent control to cooperative control, the internal force proportional gain is slowly increased to the full desired value. Also, due to limited contact sensor sensitivity, an operator must apply a small force to the object in the independent control mode before the sensors can detect the contact. Thus, to smooth the applied force reflected to the operator during a contact transition, a small bias internal force is added to the desired internal force.

Since the object position is determined by the estimated position of the virtual object (using the kinematic mapping presented in Chapter 4), the location of the fingertips are not explicitly controlled. Due to object-fingertip rolling, the position of the fingertips at the end of a cooperative control maneuver may differ from the corresponding commanded positions in independent control. Therefore, to smooth switching to independent control, the proportional gain of the fingertip impedance controller is ramped to full value over a short period of time.

5.2 Shared Control Capabilities

The object impedance architecture provides the basis for extending the shared control capabilities of the traditional bilateral telemanipulation system. By monitoring operator com-

mands and local sensory information, the slave system can make an informed choice for task intervention or can simply display state information back to the operator.

A shared control system can also warn the operator if any quantities deviate beyond an acceptable range. The quantities monitored and displayed may not have a direct feedback path to the operator. As an example, in the described telemanipulation test-bed (see Chapter 3), the CyberGrasp force feedback system is not capable of displaying incipient slip information, such as the small vibrations associated with the onset of sliding. However, the robot can track the nearness to slip utilizing its force sensors and can display this indirect information using audio feedback or by modifying the forces fed back to the operator.

In some circumstances it is preferable to have the remote system perform actions based on local sensor information instead of informing the user and waiting for a response. Continuing the example above, the slave robot could immediately adjust the forces on the object to prevent slipping. Intervention, or modification of operator commands requires a more "intelligent" slave system. The system must not only know when it is appropriate to intervene but also know when it should stop. Thus during the intervention, it is important to be able to determine the intent of the operator's desired commands. The shared control implementation and capabilities described here are built upon the cooperative control framework and its dexterous manipulation foundations.

5.2.1 Grasp Monitoring and Intervention

One of the benefits to using a dexterous manipulation foundation for shared control is in the ability to monitor and regulate the internal grasp force on an object. Suppose we have a task in which the goal is to manipulate a fragile object. A reasonable approach for autonomous grasp force regulation would be to minimize the internal force while ensuring that the object is not dropped. As discussed in the cooperative control implementation section, the measured internal force is computed based on a decomposition of the measured fingertip forces. However, the internal force commanded by the operator may be larger than necessary to delicately carry the object. Several methods exist for computing the minimum internal force given friction constraints. Kumar and Waldron [1989] discuss a fast and efficient suboptimal method to compute the necessary grasping forces. Kerr and Roth [1986] developed a method to find the optimal internal force given both friction and maximum

torque constraints using linear programming techniques. This method can be used to find the absolute minimum internal force necessary given friction parameters.

A computed minimum internal force can compared to the operator's commanded force and action can be taken to warn the operator if the commanded forces are too low or too high. Suppose the operator is handling a fragile object during some manipulation task. While transporting the object, the operator begins to relax his grasp possibly resulting in dropping the object. At a given threshold (for example, 10% greater then the computed minimum internal force), the system can warn the operator that his commanded force is dangerously low. In addition to minimum internal force monitoring, if information about the object's structural properties is known, the shared control system can warn the operator when the applied internal force is too high.

As mentioned previously, the use of the object impedance cooperative control separates the internal force control from the motion control of the object. This provides us with a means for extending the capabilities beyond a monitoring and warning system. The shared control system can also intervene to take over control of the object internal force regulation as the system deems necessary. For example, a minimum force can be applied to handle an object or the maximum force applied to the object can be limited. Although the operator and the telemanipulation system share control over the internal force magnitude, the operator still maintains the ability to manipulate the object with commanded positions.

5.2.1.1 Operator Intent

During the intervention periods, the shared controller must constantly monitor the desired motions of the operator and decide if continuing the intervention is the proper course of action. This requires determining the operator's intent. In the case where the robot system assumes control over the internal force regulation, the desired internal force of the operator plays an important role. In our implementation, if the operator commands an internal force slightly below a pre-set threshold then the shared system will take over and maintain the force at the threshold, even if the user commands a lower force. The threshold level for internal force is computed and continuously updated based on measured forces. If the operator increases his desired force above the intervention threshold, the shared system will

allow the internal force to increase. In this approach, it is possible for the operator to delegate regulation of the internal force by commanding a force that is slightly below the threshold. As we shall see in Chapter 6, the result is finer regulation of the internal force than the human users can achieve in direct bilateral telemanipulation.

During intervention, the shared system must also be able to detect when the operator desires to release the object. Otherwise, following the rules above, the operator would never be able to command an object release. One strategy is for the controller to continue to apply the minimum safe internal force until the operator's desired force falls below a second threshold - a release threshold. For example, as in Figure 5-8a, the shared system takes control over the internal force regulation until the operator commands zero internal force. At this point, the robot may initiate the object release by setting the commanded internal force to zero or a slightly negative value. When the fingertip sensors lose contact with the object, the controller transitions to the independent control mode.

The point at which the controller determines that the operator wants to release the object can actually be set to any value (e.g., Figure 5-8b). A negative desired internal force implies that the desired object size is larger than the actual object size, see Equation 5.19. In either case (Figure 5-8a or b), the operator's grasp size at release is larger than that for which the object would normally slip from the grasp, effectively adding hysteresis to the

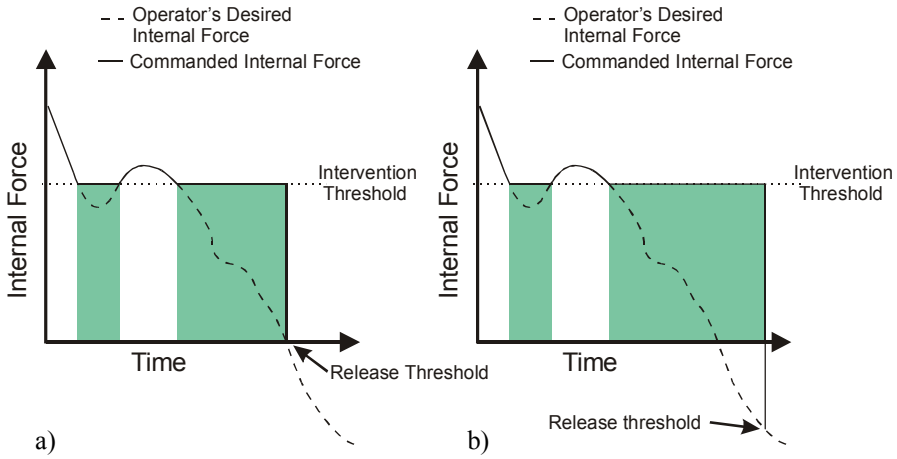

Figure 5-8. The two plots demonstrate the intervention concept applied to internal force and the differences associated with varying the threshold for discerning the operator's intent to release a grasped object. For both cases, when the operator decreases his desired internal force below the intervention threshold, the robot assumes control. However, in (a) the robot releases the object when the desired internal force reaches zero. In (b) the robot does not release the object until the desired force drops below some negative threshold giving the operator a larger acceptable desired force range for intervention (adding hysteresis to the grasp size).

grasped object size. As the release threshold moves from a possible positive value to a negative value, the "release hysteresis" is increased, providing the operator with a larger "window" of acceptable grasp sizes. However, if the window is too large, the operator may find the object "sticky" and have difficulty commanding an object release.

Discerning the operator's object release intent need not be based purely on desired force. The motions during object manipulation are relatively slow, however the actual release motion where the operator separates his/her fingers usually occurs at a higher velocity. A desired object size velocity-based release threshold complements the desired force based method. Thus, the operator can also release an object with a quick motion without the need to open his grasp beyond some release threshold. In essence, the velocity based threshold applies simple gesture recognition techniques to determine the user's intent.

5.2.2 Other Shared Control Benefits

The shared controller serves as a framework to which many features can be easily added. An example capability drawing upon the dexterous manipulation foundation and sharing of control is presented below and has been tested on the telemanipulation system. However, the point is to demonstrate that with little additional sensors or environment information, the object based cooperative control method can be used to extend the capabilities of a traditional telemanipulation system.

In any robotic system capable of semi-autonomous dexterous manipulation, the tactile sensor information is vital for accurate control and stable manipulation. However, fingertip sensors typically have a limited contact area or region in which the object must not roll or slide past during manipulation. Utilizing the object impedance controller described previously, it is possible to limit the motion of the object to prevent object contact outside of the fingertip sensor areas during manipulation. If we assume we have an estimate of the object curvature at contact (possibly obtained through direct sensor measurement [Provancher and Cutkosky 2002] or through indirect estimates [Charlebois et al. 1996]) and a planar pure rolling motion, the kinematics of contact [Montana 1988] reduce to a simple equation that describes the change in fingertip contact based on relative object and finger motion. The desired object motion can then be applied to the kinematic equations to predict the future contact locations on the fingertip. If the desired object motion causes the sensor

limits to be exceeded, the shared system can limit the object motion (and notify the operator) until an acceptable desired command is received.

One can envision many possible features that can be added to a dexterous shared control system to enhance the operator's performance. As mentioned previously, it would also be possible to modify the impedance of the object during an assembly for task level sharing, such as adding a remote center-of-compliance. With the dexterous manipulation foundation, short duration autonomous tasks could be initiated by the operator once an object is found or grasped. Furthermore, the framework supports additional sensors and displays to aid in manipulation and further immerse the operator in the remote environment. For example, vibrotactile sensors could be used to measure stick-slip vibrations encountered when grasping an object. The vibrations could be fed back to the operator or used by the slave manipulator to enhance grasp stability. A vibrotactile display could also be used to alert the operator before actual slip occurs when the commanded grasp force becomes too low.

5.3 Multi-Modal Feedback

Thus far the discussion of a shared control framework has mainly focused on the slave control aspects and implementation details. The feedback provided to the operator, based on the shared controller's actions, is also an important part of the framework. The types of feedback can be separated into two categories: direct feedback and indirect feedback.

The most obvious form of direct feedback is the visual information about the remote environment and manipulator. An operator can observe the scene either in person or through a video system. While visual information is heavily relied upon, an effective telemanipulation system can also tap into the other senses of the operator. Since telemanipulation tasks involve interacting with the physical world, taking advantage of the sense of touch can provide an immersive interface for the operator.

Direct haptic feedback implies the display of quantities measured or computed by the remote robot in a manner that closely matches what the operator would feel if doing the task himself (with a tool or through direct manipulation). A common haptic feedback method is to relay force information back to the operator. Numerous experiments have

shown the beneficial effects on operator performance with the addition of haptic feedback (e.g., [Hannaford 1991], [Howe and Kontarinis 1992], [Massimino and Sheridan 1994], [Wagner et al. 2002]). In a shared control telemanipulation system, the appropriate display of forces to the operator becomes more complex. The shared system may be sharing control over the same variables that are being fed back to the operator and a natural question that arises is; should the direct haptic feedback be modified and if so, in what way? This question is addressed in the next chapter.

With shared control over internal grasp force, there is a larger range for the operator's desired internal force that leads to a stable grasp. However, without additional haptic (especially tactile) information the operator may slowly relax his grasp (as seen in experiments by [Edin et al. 1993]) until he triggers a release. This is one example of how the operator may need additional information in order to maintain good control during intervention. As the robot assumes more control there is a desire to ensure that the operator still feels a part of the telemanipulation loop. One option may be to modify the forces fed back; however, force feedback alone may not provide enough cues nor enough flexibility to enhance operator performance with a shared control system.

Indirect feedback refers to the display of quantities measured or computed by the remote robot which do not necessarily have a physical analog for the operator. Computed quantities such as estimated grasp stability or nearness-to-slip are typical examples. Indirect feedback could also be used to relay information about the state of intervention, such as whether or not the robot is intervening. During intervention, indirect feedback could warn the operator that he is about to trigger the object's release, an assumption which is based on the robot's interpretation of the operator's commanded forces and motions.

The types of indirect feedback that can be employed in a telemanipulation system are numerous. The primary sources of feedback are sight and sound. Additional information can range from complex video-overlays, for multi-dimensional data, to simple LED indicators. Audio signals also provide a way to display indirect quantities that can range from sophisticated synthetic speech systems to tones of varying pitch. It is also possible to display information through an auxiliary haptic device such as a vibrotactile simulator.

The wide range of possible direct and indirect feedback mechanisms provides a motivation for some controlled experiments to investigate the effect of multi-modal feedback in shared telemanipulation. The next chapter describes an experiment utilizing the shared control framework described here and is evaluated based on the performance of human subjects during a prototypical manipulation task.

6 Shared Control Experiment

To evaluate a telemanipulation system, a set of experiments can be designed to measure the performance of an operator using the system. The operator is a critical "component" of any telemanipulation system, thus the motivation for testing how well an operator completes a task under different circumstances is clear. The performance metric depends on the desired aspects of the system that are being tested.

As discussed in previous chapters, extensive experimental work has been done in the area of telemanipulation and shared control. Of particular relevance, Hannaford et al. [1991] evaluated a six-axis generalized teleoperation system with arm/hand force feedback. Along with evaluation of the force feedback, a case was tested in which control was shared with the robot (utilizing local force/torque sensing) during a peg-in-hole insertion task. In this task, the operator controlled end-effector position while task-space orientation control was shared with the robot. The authors observed a reduction in task completion time and sum-of-squared forces with the addition of shared control.

Similar to the task chosen for our experiment, Howe and Kontarinis [1992] had operators treat an object as fragile while performing a peg-in-hole insertion task. Using a two-fingered teleoperated hand system with finger-level force feedback, the performance of the operator during task execution was examined at various levels of force reflection bandwidth. Operators were instructed to perform the task as "gently as possible" and avoid excessive forces. If a preset threshold for the grasp force was exceeded, an audio buzzer sounded indicating task failure and the system momentarily shutdown to simulate object breakage. Despite this "strong incentive," force feedback alone did not enable operators to minimize the grasp force applied to the object. In fact, operators used grasp forces approximately four times the minimum necessary amount to prevent slippage, regardless of the tested force reflection bandwidth. The authors concluded that a means of relaying the fric-

tional conditions at the slave finger tips to the operator is necessary for proper grasp force regulation

In another study, Howe [1992] investigated the effects of augmenting the forces fed back to the operator based on tactile sensing during telemanipulation. Using the same master-slave apparatus mentioned above, a sensorized fingertip could detect slip on the slave system and was used to alert the operator in a manner to take advantage of the operator's natural reflexive response to prevent slipping by increasing grasp force. The operator was alerted to the possibility of slip by a small change in the force fed back to the fingers. Preliminary results showed that grasp forced increased to prevent slipping and occurred at a rate faster than a voluntary response. Later experiments [Kontarinis and Howe 1995] showed that the addition of high frequency vibration feedback in addition to force feedback minimized peak forces for a puncture task.

In earlier experiments, we investigated the effects using an arm-grounded force feedback device on the performance during dexterous telemanipulation tasks. The force feedback device used, the CyberGrasp (see Chapter 3), was a light weight exo-skeleton worn on the back of the hand and applied resistive fingertip forces using tensioned cables. The results indicated that the fingertip force feedback aided the operators' with grasp stability and confidence but did not dramatically improve task completion times. We also found that during a rolling manipulation task, the force feedback device could present misleading clues to the operator and actually caused an increase in task performance time as compared to no force feedback [Turner et al. 2000, Turner 2001].

For our experiment described in this chapter, a telemanipulation task was chosen to be simple enough that the a novice operator could complete the task with a reasonable success rate but not necessarily be able to easily master the task. The task was specifically designed to investigate the possible advantages of shared control and to provide an opportunity to evaluate its effects on performance. For this, the operators were asked to grasp and transport an object described as "delicate," in other words, they were asked to grasp the object with a minimum force but not to drop it. The main performance parameters were based on the applied internal force to the object and number of task failures (drops).

The initial hypothesis for our experiment was that the addition of a dexterous shared controller to a traditional bilateral telemanipulation system will improve an operator's performance during task execution. As discussed in Chapter 5, the shared control telemanipulation system combines some of the autonomy of supervised systems with the telepresence found in direct master-slave bilateral systems. In situations where the task is sufficiently well defined, such as our delicate manipulation task, there are potential advantages to having the robot hand take over grasp force regulation. We believe by providing local control of internal forces, the robot allows the human operator to focus on the task itself, concentrating on the desired motions and behavior of the grasped object. Limitations in the accuracy and fidelity of haptic feedback through the master become less detrimental because commands from the master are supplemented by local control to prevent unwanted slips or object drops. However, there is some concern that the operator's sense of presence will be reduced as the slave system takes over more of the control. Moreover, there is an indication from the physiology literature [Westling and Johansson 1984] that without appropriate tactile cues, the human grasp force will gradually relax, leading to a divergence between conditions at the master and slave.

The shared control framework was implemented on a laboratory experimental telemanipulation system for evaluation. A diverse set of subjects were asked to perform the delicate object manipulation task. During the task, the operator and the robotic hand share control of the grasp force when handling an object. To prevent accidental drops, the robot can intervene and assume control over the internal force. The operator maintains the ability to override the dexterous controller and release the object or grasp it tightly if desired.

The subjects performed the delicate handling task for several different cases. The case conditions tested are described below and were designed to investigate several of the issues associated with shared control, including effects of robot task intervention, indirect feedback, and force feedback. The primary questions that the experiment sought to answer were:

- Is task performance improved when an operator is warned of a possible failure though indirect feedback (i.e., through audio tones or a visual indicator)?

- Is task performance improved if the robot can intervene to prevent accidental object drops during a delicate handling task?
- If the robot intervenes, is it necessary to inform the operator that the robot is assuming control?
- During robot intervention, the robot is constantly monitoring the master input to determine the operator's intent, such as the desire to release the grasped object. Is it helpful to feed back information to the operator that the operator's actions are about to trigger a state change in the robot's control (based on the measured intent)?
- With haptic feedback in a force control task, what forces fed back to the operator are most effective (especially during robot intervention)? In particular, should we feed back forces based on the measured fingertip values or should the force information be modified?
- Which combination of indirect and force feedback methods leads to the best performance compared to the control case of a bilateral telemanipulation system with only force and visual feedback?

As we will show through objective data analysis, our implementation of a shared control dexterous telemanipulation system improves the operator's task performance when compared to our telemanipulation system with only haptic and visual feedback. The performance is based on the amount of force applied to the object compared to the minimum amount necessary to carry the object. Also, feeding back information about the state of intervention and signaling the operator of impending state changes has a significant effect on performance. Additionally, trends in the data indicate that the method of feeding forces back to the operator during intervention plays an important role.

In the following sections, the experimental task is explained and the specific cases used to test the hypothesis are discussed. Next, the procedure and data collection methods are covered. The data gathered from the subject experimental trials are analyzed and the results are discussed. Additionally, the subjective data from a post-experiment questionnaire are presented and analyzed.

6.1 Experiment Description

We chose to test our hypothesis on shared control using a delicate object handling task. Specifically, operators were instructed to pick up and carry an object across the workspace and place the object on the other side using the telemanipulation system. The operators were asked to treat an object as fragile and thus use a minimum amount of force to carry the object but to take care not to drop the object during manipulation. The object, a 200 g wooden block, was picked up and placed on a target 65 cm away. Two identical placement targets were located on opposite sides of the Adept robot's workspace (see Figure 6-1), thus the block could be carried in either direction. The target area was sufficiently larger than the base of the wooden block that precise placement was not necessary, but it did require the orientation of the block to remain approximately the same. The main purpose of the targets was to provided a degree of repeatability to the experiments and not to test the operators' targeting ability. The distance from the start and end locations was large enough that all operators had to use the clutching ability of the Adept arm controller (described in Chapter 3). This provided a degree of consistency in the major motion part of the task. If

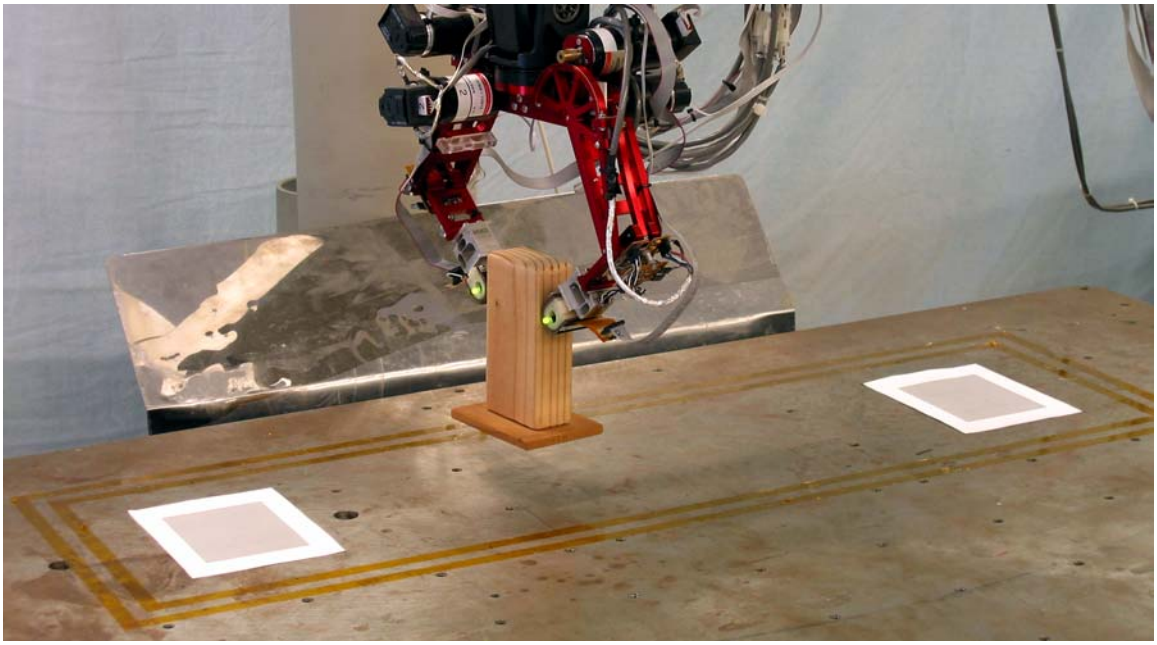

Figure 6-1. Slave system environment with target areas for pick-carry-and-place task using a wooden block as the "delicate" object.

the carry motion had been any shorter some subjects would have been able to move the object without needing to re-position (clutch) while others would have to re-position.

The following statement was read by the subjects prior to starting the experiments and gave a motivation for the chosen task:

> Imagine that you are handling something valuable and delicate by robotic teleoperation; perhaps you are recovering ancient Greek vases on the floor of the Mediterranean Sea. You want to grasp gently to avoid damaging the objects but you also want to avoid having them slip and drop. In other words, you want to stay within a target range or "window" of desired grasp forces. This is something that humans do naturally when using their hands but it can be a challenge when controlling a robot via teleoperation.

The task of regulating internal forces on a grasped object provides the opportunity to apply the shared control framework for dexterous telemanipulation. The shared control capabilities being tested were grasp monitoring and intervention, with respect to the internal forces applied to an object, and the effects of direct and indirect feedback.

6.1.1 Case Descriptions

To answer the central hypothesis on the effects of shared control and the questions posed in the chapter introduction, several test cases were designed. A total of seven cases were tested by each subject. The cases differ by the amount and type of feedback supplied to the operator and the use of the robot's intervention capabilities.

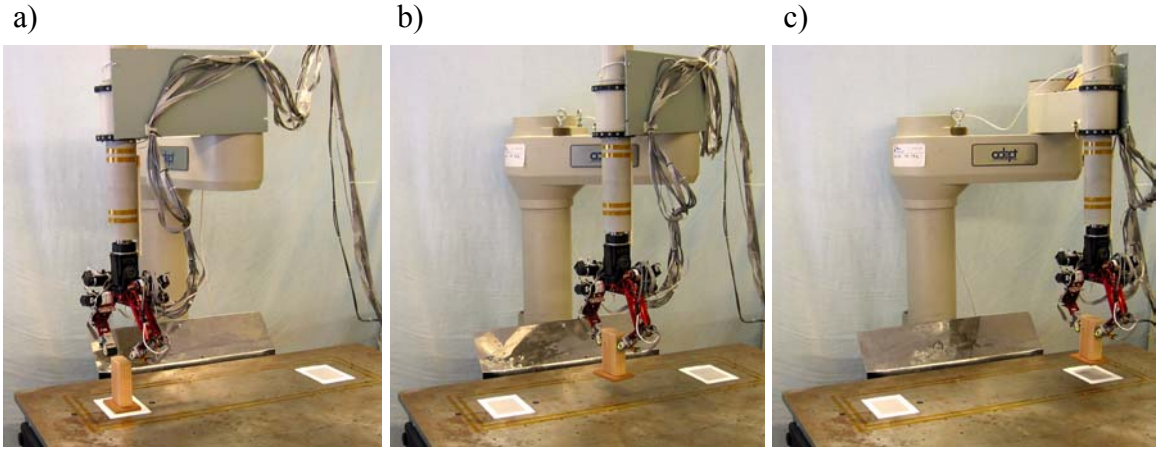

Figure 6-2. Typical task sequence: a) object grasp, b) carrying of object to target, c) object placement and release.

6.1.1.1 Overview

All of the test cases require the robot to be able to monitor the internal forces on the object. Since the task is to manipulate a fragile object, the calculation of the minimum internal force applied to the object is necessary. When grasping objects with their own hands, humans readily identify the minimum force required to prevent slipping and generally maintain a safety margin of 10%-30% [Westling and Johansson 1984]. For the robot, we estimate the minimum internal force based on *a priori* friction estimates and robot fingertip force measurements[1]. Recall that in a multifingered grasp, the contact forces can be decomposed into $f_{external}$, which balance the object weight, inertial forces and contact with the environment, and $f_{internal}$, which produce no net resultant and can be adjusted independently to prevent slipping [Yoshikawa and Nagai 1991]. For a two-fingered grasp on a block that is held approximately level, the minimum internal force becomes:

$$f_{int,min} = \max\left(\frac{f_{\text{tangential},\, i}}{\mu_i}\right) \tag{6.1}$$

where μ_i is the static friction estimate for the i^{th} finger and $f_{\text{tangential},\, i}$, the tangential force, is computed from the measured fingertip force sensor data, transformed to the contact frame. The contact frame is defined by the fingertip geometry and the contact location sensor information. The static coefficient of friction was determined empirically prior to the experiments. In general, operators carried the object with a level opposing grasp, which simplified calculation of the minimum internal force.

It is also important to note that under the virtual object mapping method, the operator's grasp size determines the desired internal force on the object[2]. The desired internal force is compared against the calculated minimum internal force for monitoring and intervention purposes. Thus, if the desired internal force falls below a given threshold, as compared to the calculated minimum internal force, a warning can be produced and/or the robot may intervene. Conversely, if the operator applies too much internal force, a warning can also be produced.

---

1. Similar to the assumptions used for the cooperative control law in Chapter 5, the object-fingertip contact is assumed to be a line contact perpendicular to the plane of robot finger motion.

2. See Chapter 4 for details on the virtual object mapping. Also see Chapter 5, Equation 5.19 on page 84.

We are interested in evaluating shared control as compared to unassisted telemanipulation and in determining what kinds of feedback are most useful to the operator. Accordingly, we examined various cases in which one or more of the following conditions were applied. The combination of conditions and implementation details for each cases tested are discussed in the following sections.

- direct bilateral control (baseline case) – the desired internal force from the operator (expressed as a reduction in the virtual distance between the fingertips, following the impedance control formulation [Hogan 1985a] - see Equation 5.19 on page 84) is used directly to control the grasp force on the object. The magnitude of the measured grasp force at each finger is fed directly back to the operator via the CyberGrasp.
- robot assisted control – when the desired grasp force drops below 110% of the minimum internal force (Equation 6.1), the robot controller intervenes to maintain the grasp force at 110% of the required minimum force, until the operator releases the object (desired grasp force < 0). When robot intervention is active, the force fed back to the user can be either the magnitude of the *actual* grasp force measured at the fingertips or proportional to the *desired* grasp force. The latter case is referred to as "reduced force feedback."
- visual indicators – when robot intervention is enabled, LEDs on the robot hand are illuminated when the controller is actively maintaining the internal force at 110% of $f_{int,min}$ from Equation 6.1.
- high audio tone – a 500 Hz warning is sounded when the object is in danger of slipping. For unassisted telemanipulation, the tone is sounded whenever the desired grasp force approaches $f_{int,min}$. If intervention is active, an audio tone is emitted when the desired grasp force approaches zero, which would trigger the robot to release the object.
- low audio tone – to discourage the user from squeezing the object too hard, a 50 Hz tone can be emitted when the desired grasp force exceeds 170% of the minimum required.

Table 6-1. Experimental case matrix indicating the different effects tested for each individual case.

Aid / Case	**Audio Alarms** when subject force too low or too high	**Robot Intervention** when subject commanded force too low	**LED Indicator** during Robot Intervention	**Reduced Force Feedback** during Robot Intervention when subject commanded force too low
Case 1	no	no	no	no
Case 2	YES	no	no	no
Case 3	no	YES	no	no
Case 4	YES	YES	YES	no
Case 5	no	YES	no	YES
Case 6	YES	YES	YES	YES
Case 7	no	YES	YES	YES

Table 6-1 shows a matrix of the different cases tested and the effects used for each case. A detailed description of each case follows.

6.1.1.2 Implementation Details - Cases without Intervention

The control case, Case 1, is the standard bilateral telemanipulation set-up for our system and serves as the base case to which shared control capabilities are added. In Case 1, the operator controls the fingers under the cooperative object impedance control law (Section 5.1.2 on page 78) once the object is detected. The forces fed back are based on the measured forces at the robot fingertips. Because the CyberGrasp system only allows unidirectional feedback, the force fed back to the operator's index finger is the magnitude of the grasp forces measured at robot's corresponding index finger (f_{xi}, f_{yi} for the i^{th} finger, from Equation 5.14 on page 81) and similarly for the thumb. Thus for the pick-carry-and-place task, the operator must use only visual and force feedback for task completion (while trying to use a minimum internal force without dropping the object).

Case 2 implements the warning capabilities of the shared control system. Utilizing the control framework and indirect audio feedback system, the operators are warned if the desired internal force on the object is too low, indicating that the operator is very close to

dropping the object. An audio alarm occurs if the desired force is within 1% of the minimum force necessary to carry the object. In other words, if the desired internal force falls below 101% of the required minimum internal force, audio feedback is used to notify the operator. To convey a sense of urgency to the operator that he/she might drop the object, a high frequency audio tone set at 500 Hz is sounded until the operator increases the grasp force, the object is dropped, or the object is intentionally released. Also, through preliminary testing we found it useful to supply the operator with a signal when the internal force on the object is excessive. If the operator applies too much internal force, a low frequency tone set at 50 Hz is sounded. This somewhat harsh tone informs the operator to relax his grasp on the object, i.e., decrease the desired internal force. The threshold for the excessive force tone is set to 170% of the minimum internal force, a suitable value determined by preliminary testing. Thus if the minimum internal force was calculated to be 2.0 N, the high frequency alarm would sound if the operator's commanded force dropped below 2.02 N and the low frequency alarm would sound if the operator's commanded force was above 3.44 N.

At first glance, setting the high frequency audio alarm for Case 2 to 101% may seem a bit peculiar. There are several reasons why this level is appropriate. First, we desired to use the same "excessive force" low frequency audio tone for all cases with audio alarms. Through preliminary testing we found that if the high frequency alarm was set much higher than 101%, operators felt the separation between alarms was too small and were confused as to whether they should increase or decrease the grasp force. This is mainly due to the fact that there was a large amount of variability in the grasp force which often triggered the alarms during task execution. Also, even with a warning set at 101%, analysis shows the average internal force the operators applied was 150% of the minimum internal force; indicating that operators' internal force levels were very close to the excessive alarm level.

6.1.1.3 Implementation Details - Cases with Intervention

Cases 3-7 all employ the intervention capability of the shared controller. This requires that the robot slave hand not only monitor the internal force applied to the object but also take control over the commanded internal force to prevent unwanted slips. When the operator's desired force drops below 110% of the calculated minimum internal force, the robotic hand

assumes control over the internal force. The operator still has full control over the object's in-hand position and orientation. The shared controller sets the internal force to 110% of the calculated minimum until the operator increases his force above this level or the system determines that the operator intends to release the object. The 110% value was empirically determined but is similar to the safe minimum internal force used by humans when manipulating an object [Westling and Johansson 1984]. We should also note that this level puts the intervention cases at a theoretical disadvantage as compared to the first two cases, in terms of the applied force performance metric. In other words, it is possible under Cases 1 and 2 for the operator to command a force less than 110% of the absolute minimum; while the lowest possible force for Cases 3 through 7 is 110% of the absolute minimum internal force. However, we will see that even with this bias, the performance for the cases with intervention is generally better.

An important part of the cases with intervention is determining the operator's intent, especially during periods in which the robot has assumed control. As discussed in Section 5.2.1.1 of the previous chapter, the level at which the controller determines the operator desires to release the object is somewhat arbitrary and can be based on more than one quantity. If the parameters are not chosen carefully, the operator may have trouble releasing the object or the object may be released unexpectedly. For this set of experiments, the determination of the operator's intent to release an object is based only on the operator's object grasp size. While this quantity directly relates to the desired internal force, by Equation 5.19 on page 84, the thresholds are more easily specified based on desired object size rather than desired internal force.

A threshold value of approximately 6 mm was determined through preliminary testing to be an appropriate value. To illustrate the threshold effect, the following numerical example is given. Assume an operator is grasping an object using the master-slave system. If the operator initially grasps an object with a hand grasp size of 5 cm, assume that the operator must maintain a smaller grasp size of 4.8 cm to apply a force greater than the required minimum internal force (to prevent the object from slipping). In other words, he/she has to squeeze the virtual object between his/her fingertips. If the robot is under normal bilateral control (Case 1 or 2), the operator would drop the block if the grasp size increases

above the 4.8 cm grasp size. However, if the intervention capability is enabled, the operator may enlarge his/her grasp 6 mm beyond this grasp size that is normally needed to maintain the minimum internal force on the object. As the operator enlarges his/her grasp beyond 4.8 cm, the robot hand assumes control and regulates the internal grasp force to an appropriate level. If the operator opens his/her grasp beyond the 6 mm window (i.e., larger than 5.4 cm), the shared controller assumes the operator wishes to release the object and the object is dropped.

By having the shared controller release the object when the robot's commanded grasp size is some amount larger than when the object would normally drop, we can effectively add hysteresis, making the object release less sensitive to human grasp size changes. In other words, we expand the operator's acceptable commanded grasp size window, by increasing the desired object size at which the robot releases the object. This permits the operator to apply low grasp forces but not drop the object. If the operator stays within this target window, the robot will maintain the grasp force level at 110% of the minimum required internal force. Figure 6-3 illustrates the described effect.[3]

As mentioned previously, the operators' grasp size changes are used to compute the desired internal force on the object by Equation 5.19 on page 84. Given our standard object, its friction properties, and the internal force gain, the minimum required internal force is approximately 1.7 N. Based on the human-to-robot mapping parameters and the a necessary internal force bias (See "Transitions" on page 85.), a 6 mm increase in grasp size is equivalent to the operator commanding a desired force on the order of -0.25 N. This may appear to be a rather large change in desired force; however, the 6 mm "release hysteresis" was empirically determined to be the best value. During preliminary testing it was found that operators could become confused as to the necessary action to maintain the grasp in the target window if the threshold was reduced too much. This is similar to the effect seen in Case 2 if the low tone and high tone audio alarms thresholds were too close together.

3. We should point out that the size of the target window (of 6 mm) varied slightly due to an operator's individual mapping parameters. However, the object size mapping parameters were fairly consistent among subjects so the size of the window was on the order of 6 mm, ± 0.5 mm.

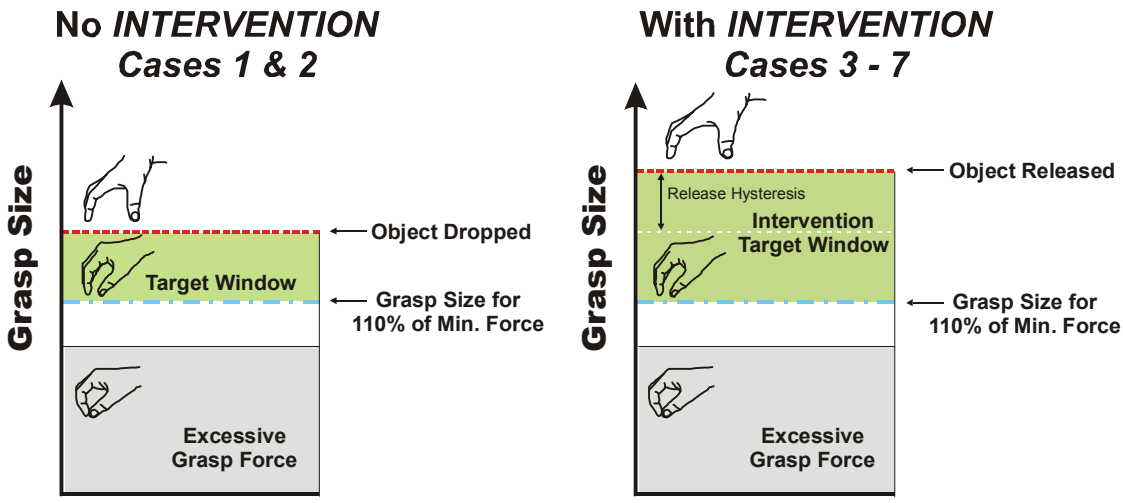

Figure 6-3. Illustration shows the effects of intervention on object release with respect to the human operator's hand grasp size. The left graphic shows the various grasp sizes and the corresponding actions on the object. The right graphic indicates that when an operator increases grasp size beyond a preset threshold the robot will intervene. By opening his grasp slightly larger than what would normally drop the object, the operator can command the robot to release the object. The intervention gives the operator a larger target grasp size window to command low forces (110% of the minimum internal force) but not drop the object.

Case 3 implements the intervention method to aid the operator in maintaining low grasp forces while also helping to prevent dropping the object. During this case, if the operator decreases his/her applied force level below 110% of the calculated minimum internal force the robot will assume control over the applied internal force on the object. If during the intervention period the operator increases his/her grasp force above 110% level, he/she will assume control over the grasp force. Additionally, the allowable grasp size is increased by setting the threshold to be larger (by the 6 mm threshold) than the initial grasp size. At the point when the operator exceeds the release threshold, the robot simply commands zero grasp force on the object. In Case 3, no audio feedback is supplied to the operator.

It is also important to consider the forces to be fed back to the operator. For Case 3, the forces fed back are based on the magnitude of the measured forces at the robot's fingertips, exactly in Cases 1 and 2. However, during intervention the robot is applying 110% of the computed minimum internal force, therefore the measured force at the robot's fingertips and thus the forces fed back to operator's fingertips will be nearly constant.

Case 4 is similar to Case 3 in that the robot intervention capability is enabled using the same levels (110%) and the same force feedback (based on the magnitude of measured tip forces). However, additional feedback is supplied to the operator about the intervention

state and deduced user intent. Specifically, if the operator reduces his desired grasp force (enlarges his grasp size) such that the robot intervenes, LEDs in the robot's fingertips are illuminated. The LEDs indicate to the operator that the robotic hand has assumed control over the internal grasp force (i.e, the operator is within the target window for grasp size). As with Case 3, because of intervention, the operator can open his grasp larger than what would normally cause the object to be dropped. To give the operator more information about how close he is to commanding the robot to release the object, audio feedback is used. If the grasp size is at a point that is greater than approximately 75%[4] of the 6 mm release threshold, a high frequency tone is used to warn the operator that he is close to triggering an object release. (This threshold level is equivalent to a desired force on the order of 0.35 N.) The low frequency audio tone is also used to warn the operator that desired internal force is excessive. The trigger level for the excessive force alarm is set to 170% of the minimum internal force.

Case 5 enables the intervention capability of the robot during the task. However, as in Case 3, no audio alarms or LED indicators were used. Instead, the haptic feedback was augmented in an attempt to inform the operator as to where he is within the intervention state. To accomplish this, the magnitude of the measured grasp force applied to each finger was reduced in proportion to the size of the operators grasp (or equivalently, the desired internal force) during intervention. The forces at the fingertips started to reduce at the point when the operator commanded less than the minimum internal force (once in the intervention target window). The fed back forces reduced to zero, in proportion to the grasp size, at the point when the object release threshold was exceeded. Thus, as the operator opens his/her grasp within the intervention target window, the force applied to the operator's fingertips reduces but the force applied to the object is maintained at 110% of the calculated minimum internal force.

Case 6 is similar to Case 4 in that intervention capability is enabled, the robot fingertip LEDs are used to indicate intervention, and the audio alarms are enabled to alert the operator of excessive desired force and low desired force. Case 6 also includes reduced

4. Because the actual size of the "release hysteresis" window, the 6 mm window, varied slightly due to an individuals mapping parameters, the alarm threshold was set with respect to a fixed lower limit.

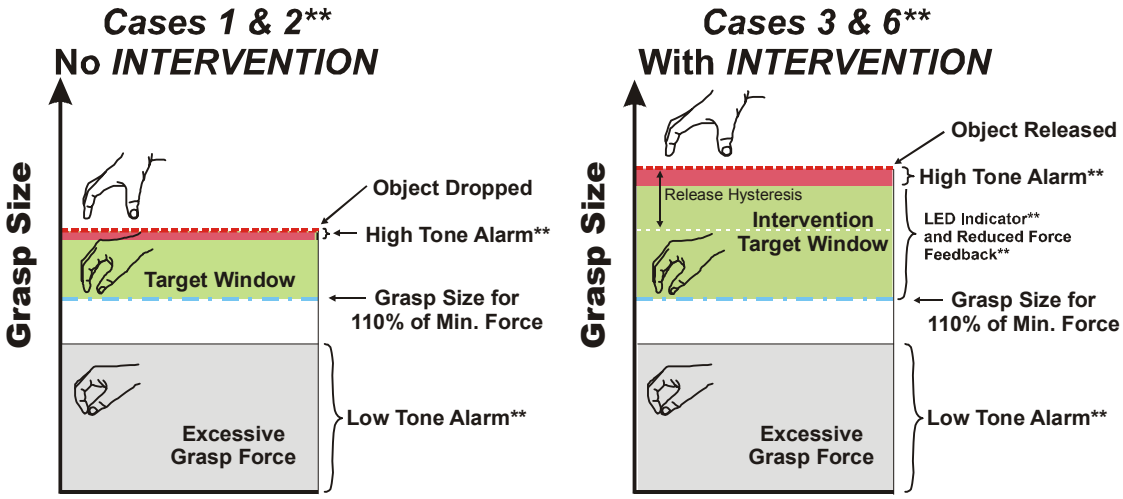

Figure 6-4. Comparison plots of feedback to operator for Cases 1 and 2 to Cases 3 and 6 based on operator grasp size. Relative levels are shown for high tone and low tone audio alarms. For Cases 3 and 6, the robot intervenes when the grasp size causes the desired force to be less than 110% of the minimum internal force. During intervention in Case 6, an LED indicator at the robot fingertip is used. Also for Case 6, the forces fed back are reduced by an amount proportional to the size of the grasp within the intervention target window. For the intervention cases, there is a release hysteresis. In other words, the operator has to open his grasp beyond what would normally cause the object to drop. Also the high tone alarm during intervention occurs when the operator is near commanding the robot to release the object.

force feedback to the operator during the intervention as in Case 5 (effectively feeding back the desired internal force). Figure 6-4 compares Cases 1 and 2 to Cases 3 and 6, illustrating the differences in object release compared to operator grasp size and the relative alarm levels.

The last case, Case 7, was similar to Case 6, however the audio alarms were not enabled. Thus, during intervention, the operator could use the LED indicators and the reduced force feedback for task completion. Refer back to Table 6-1 for a comparison matrix of the different cases tested and the effects used for each case.

6.1.2 Procedure

A diverse set of eleven subjects was recruited for the experimental tests. Eight males and three females were tested using the experimental set-up and the cases described in the previous section. Several preliminary subjects were tested to gain feedback about experimental procedure and refine the parameters for each case.

Each subject attended two sessions to complete the experiment. During the first session, the subject was calibrated to the glove system using the method discussed in Griffin

et al. [2000]. The human-to-robot mapping parameters were then defined based on the calibrated virtual hand model and the operator's natural range-of-motion. The details of the mapping method are discussed in Chapter 4. Subjects then performed simple manipulation motions; any necessary adjustments to the mapping parameters were made by the experimenters. The subjects were also introduced to the force feedback and asked to grasp, manipulate, and release an object within the robotic hand. Next, the subjects were familiarized with controlling the robotic slave arm using the ultrasonic wrist tracker. Finally, the subjects were given a chance to pick, carry, and place a wooden block under Case 1, with only visual and force feedback to aid the operator. Once the they felt comfortable with the telemanipulation system, the session was concluded. The first session lasted approximately one hour.

The second session occurred two to four days after the first session. Calibration and mapping parameters for each subject were loaded from their previous session. The subjects were then re-familiarized with the robotic hand, the force feedback, and the control of the slave arm. The subjects were asked to perform the pick-carry-and-place task under Case 1 until they again felt comfortable with the system. At this point the subjects were asked to read the passage describing the motivation for the experiments (see Section 6.1). The seven cases were explained to the operator using graphics similar to Figure 6-4, and demonstrated by having the operator grasp the wooden object and slowly release it from his grasp, thus observing the variation in the case parameters.

Each subject was asked to carry the wooden block from one side of the robotic arm's workspace to the other (65 cm from target to target) using a minimum force necessary but not dropping the object. If the block was dropped or the subject did not place the object within the target on the opposite side, the task was marked as a failure and the block was moved to the center of the intended target for the next trial. Subjects were told that trial completion time was not considered in scoring the task performance.

The subjects were asked to complete the task under the seven different cases. The case order for each subject was randomized to reduce any possible presentation-order bias on the overall subject data. At the start of each case being tested, the subject was re-familiarized with the case by grasping and slowly releasing the object. Each case consisted of a

practice trial and four test trials in which the subject would carry the block from one end of the workspace to the other end. During the practice trial, subjects were encouraged to explore the different types of feedback that could be elicited for that particular case, and the aids and/or limitations that may be present for the case. No data or failures were recorded during the practice trial.

The start of each test trial was initiated by the experimenters once the subject positioned the robotic hand above the object and indicated he or she was ready. The four trials were tested in sets of two, allowing the subject to rest after the completion of the second trial. At the start of each trial, the subjects followed the procedure of positioning the robot hand above the block and indicating readiness. Following the four recorded trials, the subjects were given a brief rest period (2-5 minutes) and the next case was explained.

6.1.2.1 Data Collection

Data were recorded for each trial, both manually by the experimenters and by the computer controlling the slave robot. When the subject indicated he/she was ready, the trial began and the data collection using the computer was started. The data collection was ended when the subject placed the object in the target successfully or when the subject failed the task by misplacing or dropping the object. The following data were collected and recorded at 200 Hz by the computer: time, measured internal force on the object, the operator commanded (or desired) internal force, the robot commanded internal force, the calculated minimum internal force, the robot fingertip forces, the applied forces to the operator using the CyberGrasp, the intervention state, and the state of the audio alarms and LED indicators. The experimenters manually recorded failures.

Once all the cases and trials were completed, each subject was asked to complete a post-experiment questionnaire. The purpose of the questionnaire was to obtain qualitative feedback on the differences among the cases through specific questions that required the subject to rank the cases tested in terms of preference and ease. Questions were also asked about the perceived effects on performance for the different types of feedback used. The questionnaire is reprinted in Appendix E.

6.2 Results

To evaluate the telemanipulation system we must carefully review the results from the human subject experiments. Initial data exploration is often based on observing trends in typical subject graphical data. While trends can provide clues for areas of investigation, clear objective criteria must be defined for assessing the task performance. Objective data analysis can be based on many different quantities but the purpose of our experiment dictates the performance measurements. The objective data analysis is primarily based on the measured internal force applied to the object and the number of failures recorded, and comparing these quantities for each case. These particular variables are used because of the task goal: handling a fragile object with a minimum internal force but not dropping it. A statistical analysis of the data allows us to draw conclusions regarding the effects on subject task performance. In addition to objective data analysis, subjective criteria (such as the users' expressed preferences) can provide additional information for system evaluation [Burdea 1996].

At the most basic level, we would like to be able to determine which features of the shared control system improve task performance. The following sections address this question and presents typical graphical data of subject performance, the objective data with a statistical analysis, and the subjective data analysis.

6.2.1 Typical Subject Data

Plotting key variables during each trial for each case clearly reveals several trends in the data. Figure 6-5 shows typical trials of one subject for Cases 1, 2, and 6. Plotted for each case is the measured internal force computed from the robot's fingertip sensor data using the Yoshikawa and Nagai [1991] decomposition method, and the operator's desired force, which is based on the operator's grasp size and the virtual object mapping parameters. Also plotted for Case 1 is the calculated minimum internal force based on the force sensor data and friction estimates. For Case 2, 101% of the minimum internal force is plotted. If the operator drops below this force level, the high frequency audio tone is sounded as indicated on the plot. For Case 6, 110% of the calculated minimum internal force is plotted instead because this is the threshold at which intervention is triggered. Also for Case 6, the periods in which the operator is warned (with a high tone alarm) that he/she may cause the robot to

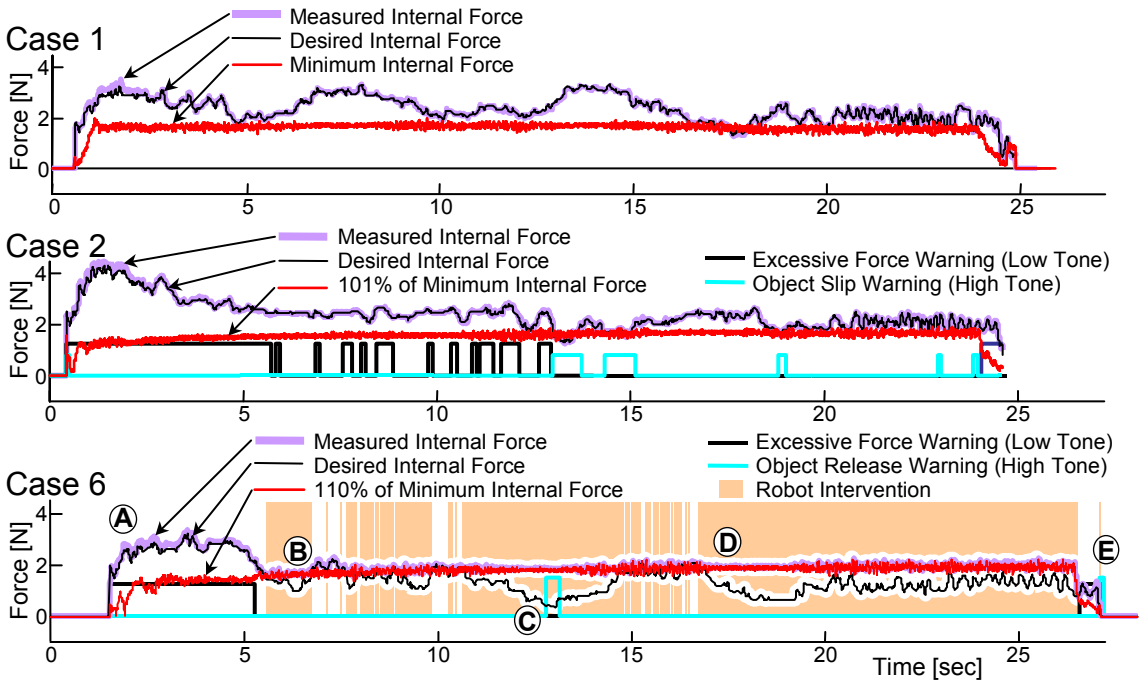

Figure 6-5. Typical subject data recorded during a single task trial for Cases 1, 2, and 6. Each plot shows the measured internal force (computed using force sensor data), the desired internal force (based on operator's grasp size), and the minimum internal force trigger level (100% for Case 1, 101% for Case 2, and 110% for case 6, all computed based on force sensor data and friction estimates). During Case 1, the subject's desired force is on average larger then the calculated minimum internal force necessary. During Case 2, we can see that the subject receives repeated excessive force warnings (low tone alarm) during the first half of the trial. For the second half, the subject does much better and effectively uses the object slip warning (high tone alarm). During Case 6, the subject is initially warned of excessive force and informed when the robot is intervening. The subject allows the robot to assist in task completion and therefore applies lower average internal force on the object. Markers A-E are described in the text.

release the object are also shown. For Cases 2 and 6, the periods in which the excessive force warning occurred are plotted. The low frequency tone was sounded when the operators applied more than 170% of the minimum internal force.

In the Case 1 plot, one can see that the operator initially commands a relatively large internal force. The measured internal force tracks the operator's desired force very closely due to the applied PI control law for internal force. Also for Case 1, the measured internal force, on average, is higher than the minimum internal force necessary to carry the object. Near the end of the Case 1 trial, the subject reduces the applied force.

For Case 2, the operator is warned if the commanded internal force is too low or too high. During the first half of the trial, the operator is repeatedly warned that the applied force is excessive (i.e., greater than 170% of the calculated minimum internal force). The operator is warned with a low frequency audio tone. During the second half of the trial, the

operator reduces the desired force and is informed with a high frequency tone when his/her force drops below 101% of the minimum internal force required to carry the object.

In addition to the forces and alarms, the Case 6 graph shows when the robot intervened. At the start of the trial, once the object is lifted off of the table, the operator applies a much larger force than necessary (Marker A). The excessive force warning (a low tone) is sounded until the operator reduces the desired force below 170% of the minimum required internal force. As the operator reduces the desired force below 110% of the minimum required internal force, the robot intervenes by taking over the commanded force relayed to the internal force controller (Marker B). The operator is notified of the intervention by LED indicators in the robot's fingertips. During intervention the measured force follows the 110% minimum internal force level despite the operator's lower desired internal force. In the middle of the carry phase the operator begins to slowly reduce the desired force. At Marker C the shared controller informs the operator with a high frequency audio tone that the desired force is very low and could trigger an object release if the desired force continues to reduce (equivalent to the operator opening his/her grasp). Also, the force fed back to the operator is reduced in proportion to the decrease in desired force. Because the operator is in the middle of the carry phase of the task, the operator increases the desired force enough to stop the release warning. After Marker D the operator continues to complete the task and allows the robot to intervene and command a minimum safe internal force. At the end of the task the operator releases the object by opening his grasp a substantial amount (Marker E).

6.2.2 Objective Data Analysis

The objective data analysis is based on the measured internal force applied to the object. Specifically, the average measured internal force applied during each trial (for each case) serves as the objective task performance measure. Based on the goal of the telemanipulation task, using the measured internal force is a logical choice. However, for a complete picture of task performance for each case, one must also consider the number of failures. A comparison analysis of the performance measures allows us to determine which case and/or case features maximize operator performance. Additionally, statistical analysis allows us to state with a high degree of confidence that the results found are as they appear and not

the result of chance. Also using statistical methods, we wish to examine if the results are applicable to the general population. Furthermore, we can create a model of the data to examine if it is necessary to account for effects from learning and/or fatigue.

An initial comparison of the cases is based on the subjects' averaged internal force applied to the object. Each subject completed four trials for each case; however, in some trials subjects failed to complete the task. Thus, each subject's case performance was based on the average of the trials excluding the trials with failures. In this way, we obtain eleven performance values for each of the seven cases from the eleven subjects tested. Figure 6-6 shows a boxplot of the average measured internal force on the object for each case. A box-plot is a common statistics graphical tool for easy visual comparison of data sets. The box stretches from the lower quartile (25^{th} percentile) to the upper quartile (75^{th} percentile). Thus, the box makes up the inter-quartile range, IQR, which contains the middle half of the data. Also shown on the boxplot is the median (horizontal line within the box) and the smallest and largest observations within $1.5 \times \text{IQR}$ from the edge of the box (the horizontal lines at the end of the "whiskers," dashed vertical lines). Any points further than $1.5 \times \text{IQR}$

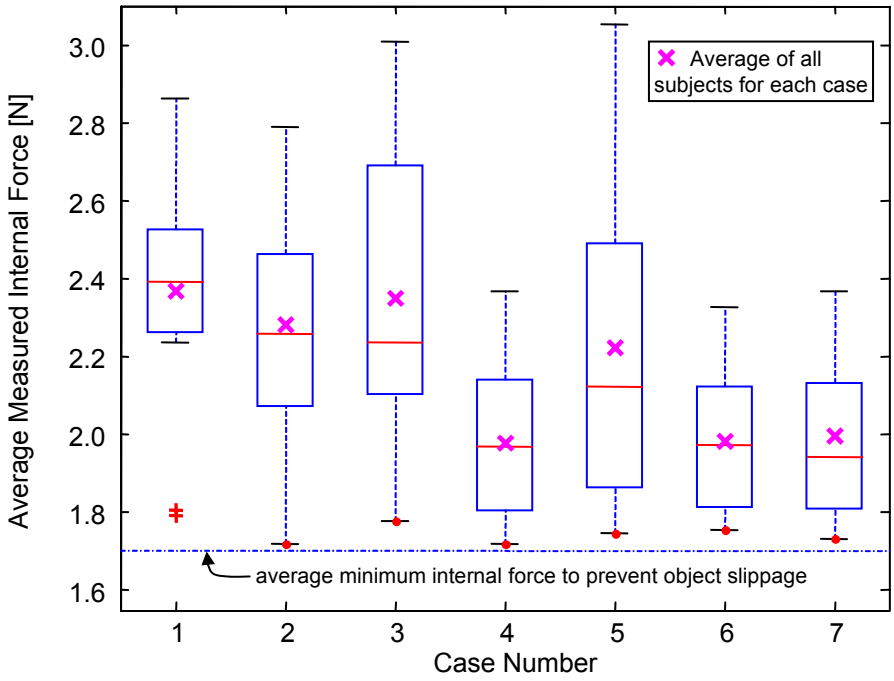

Figure 6-6. Box plot showing medians, quartiles, extremes, and outliers for the average measured internal force, using each subjects case average (over all trials except failed trials where the object was dropped or misplaced). The box stretches from the lower quartile to the upper quartile (forming the IQR) with the median line in the middle. The whiskers show the smallest and largest observations within $1.5 \times \text{IQR}$ from the box edge. Points beyond this are considered outliers and are marked with a '+'. The average measured internal force of all subjects for each case is indicated with an 'x'.

from the box edge are typically considered outliers. The boxplot is also useful because there is no assumption of a normal distribution of the data (i.e., as in a plot of the average and standard deviation). Thus if any extreme outliers exist, an easy visual comparison can still be made. There is clearly a reduction in the measured force when comparing all cases to the control case, Case 1. In particular, Cases 4, 6, and 7 have a distribution that is much lower than the control case. This gives us a preliminary indication that the averages for these cases may be significantly lower than the control case, thus supporting our hypothesis that the addition of a dexterous shared controller added to a traditional bilateral system will improve operator performance. Recall again that in theory it is possible to complete the task under Case 1 with a lower overall force level than in any of the cases with intervention (Cases 3-7). In essence, there is a bias against the cases with intervention (in terms of achieving better performance) because the lowest force the robot can apply is 110% of the minimum internal force necessary to carry the object, while it is possible to apply less force in Cases 1 or 2 and still successfully complete the task.[5]

To determine if the differences in the cases are statistically significant, an analysis of variance (ANOVA) can performed. The ANOVA test allows for a statistical comparison of differences between two or more effects by comparing the variation within effects to the variation between effects. In this experiment we have a single factor with seven fixed-effects or levels (i.e., the cases) and we wish to determine if the mean measured internal force (the response) for each case is the same or different. Each case mean for the measured internal force is averaged over all subjects and the standard deviation for each case is presented in Table 6-2. A single-factor balanced ANOVA test was run on the averaged measured internal force of each subject for each case (removing trials that were considered failures) yielding eleven data points per case (76 degrees of freedom). Thus we are testing a null hypothesis that the different cases have no effect on the measured internal force and the means for each case can be considered equivalent. The alternate hypothesis is that at least two case means are significantly different from each other. The results of the ANOVA test are in Table 6-3. The ANOVA test results in a p-value of 0.003. The p-value represents

5. The line in Figure 6-6 indicating the minimum internal force to prevent slippage is an *average* based on all trials. Thus, in cases with intervention, not all subjects' average measured force is strictly 10% above this average value.

Table 6-2. Mean and standard deviation of measured internal force for each case based on each subjects' average performance.

	Measured Internal Force Applied to Object[a] [N]	
Case Number	Mean [N]	Std. dev. [N]
1	2.37	0.34
2	2.28	0.30
3	2.35	0.40
4	1.98	0.21
5	2.22	0.42
6	1.98	0.18
7	2.00	0.20

a. Averaged over all subjects using each subjects' averaged case force for all trials (excluding failures).

Table 6-3. ANOVA results table for measured internal force.

Source	DF	Sum of Squares	Mean Square	F	Pr > F, (p -value)
case	6	2.08	0.35	3.71	0.003
error	70	6.54	0.09		
corrected total	76	8.63			

the probability of having a test statistic, F, as extreme as the one observed. If the p -value is less than our significance level (a 95% confidence level or $\alpha = 0.05$), we reject the null hypothesis. With $(p = 0.003) < (\alpha = 0.05)$, we can conclude that at least two cases have a statistically different mean. For a review of the ANOVA procedure see [Devore and Farnum 1999].

Unfortunately, when the ANOVA test is performed for a factor with more than one level, we can only confidently state that at least two means are different; but we do not know which ones. For this we must apply some type of comparison procedure to all the cases. The t -test is a common statistical tool for comparing two population means with a small sample size, for which the null hypothesis is that the two means are the same and we test to see if the data show otherwise. However, if we apply the t -test multiple times to compare each case to every other case we increase our chances of falsely finding that two cases are different (also known as a false-positive or a Type I error). This possibility is due

to the fact that the t-test must be performed at some significance level (typically $\alpha = 0.05$) and therefore the 5% chance of making a single comparison error is increased by repeated comparisons. To this end, multicomparison procedures have been developed to place an upper bound on the Type I error while allowing one to make several comparisons at a given overall significance level.

For our experiment, we can consider Case 1 as a control case. The control case represents the basic bilateral telemanipulation system, while the other cases (Cases 2 through 7) add features enabled by the shared control framework. We chose to use the multicomparison procedure known as Dunnett's method to analyze the force data. Dunnett's method is designed for the comparison of several effects to a control effect while limiting the possibility of the Type I error to the desired significance level [Devore and Farnum 1999]. Applying this method to the measured internal force for each case, we can state with a 95% statistical confidence that Cases 4, 6, and 7 have a mean different from Case 1 (with the minimum significant difference = 0.30 N). This result and the averages in Table 6-2 indicate that the shared control framework as implemented in Cases 4,6, and 7 result in significantly lower forces applied to the object than the control case; a reduction of approximately 15%.

To develop a complete picture of the effects of shared control on task performance, we must also analyze other parameters than the measured internal force. While the results show a noticeable improvement in force regulation, we must also consider the amount of time the subjects took to complete the task. Even though subjects were not informed that task completion time was a factor in task performance, the task time may reveal information about the mental or physical difficulty associated with completing the task under the various conditions. A box plot of the task completion time for each case, based on each subject's averaged trial completion time, is shown in Figure 6-7. Based on the medians and spread in the data, it is difficult to identify any trends in the task time. An ANOVA test was performed using the eleven subjects' task times for each case to confirm the lack of any significant difference in the completion time means (see Table 6-4). The analysis resulted in a p-value of 0.82 ($F(6,70) = 0.48$) indicating that the mean task completion times are not statistically different. The lack of any significant difference in the completion time is impor-

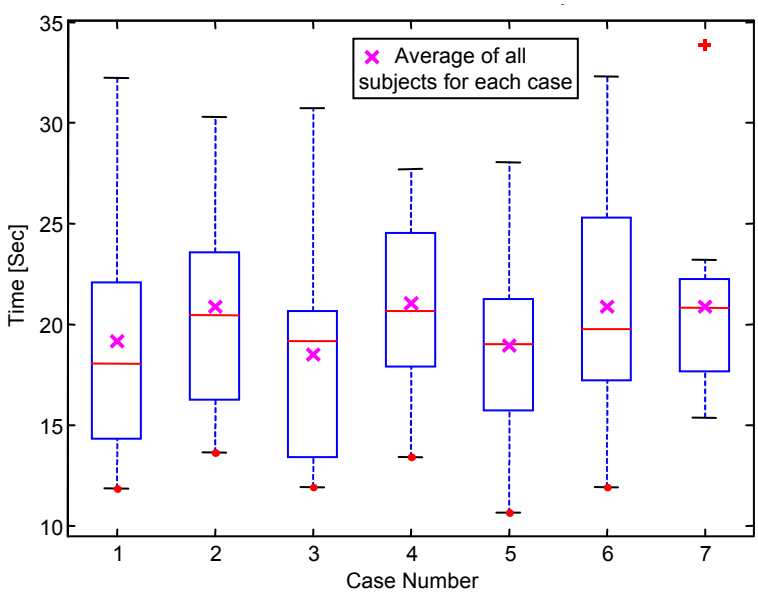

Figure 6-7. Box plot of subject task completion time for each case. Subject task completion time for each case is computed by averaging trial completion time (except for failure trials). The average completion time for all subjects for each case is indicated with an 'x'.

Table 6-4. Mean and standard deviation of task completion time for each case.

	Task Completion Time[a] [sec]	
Case Number	Mean [sec]	Std. dev. [sec]
1	19.17	6.11
2	20.92	5.73
3	18.48	5.49
4	21.07	4.56
5	18.97	4.83
6	20.90	6.01
7	20.89	5.05

a. Averaged over all subjects using each subjects' averaged case time for all trials (excluding failures).

tant; while there is no improvement in task time for cases with shared control, there is not an increase in task time either.

In addition to observing task completion time, the number of failures that occurred during each case must be examined. Figure 6-8a shows the total of number of failures for each case. With eleven subjects and four trials per case, each case was attempted 44 times, with an average failure rate for all cases on the order of 10%. More importantly, the number of failures for Cases 5 and 6 are the lowest overall, with only two failures out of 44

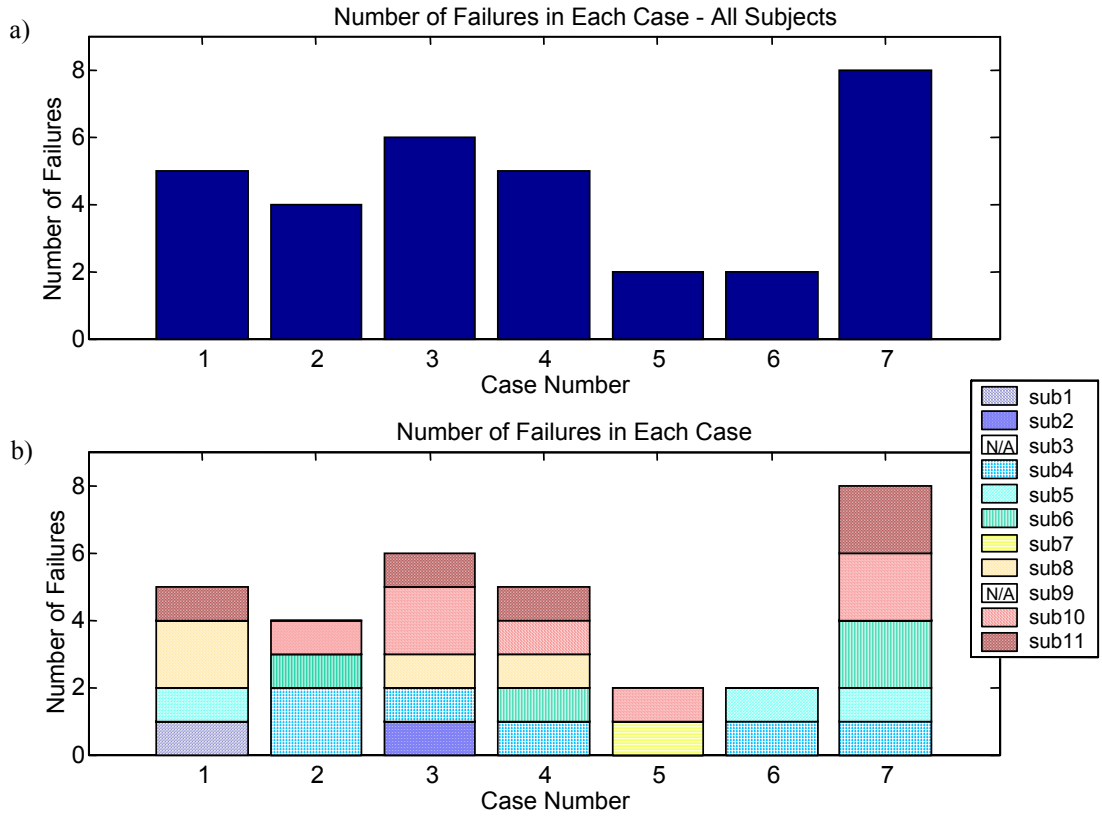

Figure 6-8. a) Total number of failures for each case for all subjects. b) Failures for each case showing individual subject failures.

attempts. The number of failures for Case 1, the control case, was more than twice that of Cases 5 and 6.

6.2.2.1 Synthesis of Results

Taking into account the task goals and the objective performance criteria based on measured internal force, task completion time, and number of failures, it is clear that the addition of a dexterous shared controller to a traditional bilateral telemanipulation system improves the operators' performance for the tested telemanipulation task, thus supporting our initial hypothesis.

Based on the significant differences in measured internal force and on observing the number of failures, we can conclude that Case 6 especially improves the operator performance in a delicate object handling task. While Cases 4, 6, and 7 all had statistically lower measured forces than the control case (Case 1), Case 6 had the least failures. Therefore we

can state that the combination of using the robot intervention capabilities, informing the operator that the robot is intervening through a visual indicator, supplying audio alarms to inform the operator that he/she is close to triggering object release, supplying audio alarms when the desired force was excessive, and reducing the force feedback to the operator based on the desired force, can significantly improve an operator's performance.

By observing trends in the differences in the other cases, we can try to determine which features of the shared control system have the most influence over operator performance. A comparison of Case 1 (the control case) to Case 2 (audio feedback, but no intervention) shows a slight reduction in both measured force and number of failures. This trend indicates that warning the operator of a possible failure through indirect feedback may be helpful. Of course this is dependent on the type of indirect feedback and the levels at which the warnings occur. During preliminary testing we found that if the low force alarm (the high tone) was set any higher than 101% of the minimum internal force, the operator complained that the high force and low force warnings appeared to be "too close together" causing confusion over whether to squeeze harder or release his/her grasp. This confusion may result from the fact that during task execution there was a substantial fluctuation in the desired force level, causing both audio alarms to sound frequently. Thus, even with an average force at 150% of the minimum internal force (for Case 2), the subjects' force often dropped below 101% and increased above 170% of the minimum internal force[6].

Comparing Case 1 performance to cases with intervention (3-7) shows that if the robot assists or intervenes during a delicate object handling task the operator's performance can be improved. However, the data indicate that the presence and type of direct and indirect feedback have a marked effect. For example, in Case 3 trials, the robot could intervene if the operator reduced the desired force on the object below the critical level; however, no feedback was available to the operator about the state of the intervention other than the robot fingertip forces. We find the mean measured internal force is virtually the same as in

6. The minimum internal force was on average about 1.7 N; therefore the excessive force alarm and the "slip" alarm (the high frequency alarm) for Case 2 would sound at the desired force levels of about 2.89 N and 1.72 N, respectively. We can see that from Table 6-2 the average measured force was 2.287 N with a standard deviation of 0.30 N.

Case 1, showing no improvement over the control case. Additionally, the number of failures for Case 3 is the second highest for all the cases.

The cases which informed the operator that the intervention was occurring (Cases 4, 6, and 7) had lower internal forces than the control case (Case 1), whereas Cases 3 and 5 (no indication of intervention) did not. The differences in performance demonstrate the need to inform the operator that the operator is assuming control. In Cases 3 and 5, the operator could not rely on audio or visual indicators to provide information about the state of intervention. The standard deviations of the measured internal forces for these cases were the highest, indicating that operator performance varied a great deal. With the additional information provided in Cases 4 and 6, the mean forces were the lowest and both cases had much lower standard deviations.

If we examine the number of failures, we find that simply informing the operator the intervention was occurring, using LEDs as a visual indicator, was not adequate. Comparing the number of failures for Cases 6 and 7, we can see that it is beneficial to feed back information to the operator about the possible triggering of object release. In other words, the audio high frequency audio alarm that alerts the operator of the nearness to object release is an important component of the indirect feedback for preventing dropped object failures. In Figure 6-8a, we can see the number of failures for Case 7 is four times that of Case 6. During Case 6, indirect feedback was also used to inform the operator that excessive force was being applied to the object with the low frequency audio tone. However, operators typically triggered the excessive force alarm only at the start of the trial. While the excessive force alarm was not tested separately from the high frequency alarm, the use of the high frequency alarm as a release warning was most likely the dominate factor in reducing the number of failures in Case 6.

Another question of interest is: What type of force feedback is appropriate during robot intervention? During intervention periods, the forces fed back to the operator in Cases 3 and 4 were essentially constant. The forces were based on the measured internal force, which was servoed by the robot to a safe minimum value (110% of the calculated minimum internal force required). Force feedback for Cases 5 and 6 differed from Cases 3 and 4 in that the force feedback was reduced based on the operator's desired internal force. Thus, as

the operator relaxed and opened his grasp, he would receive a haptic cue that this was occurring. A comparison of Cases 3 and 5 isolates this effect; neither of the cases had audio alarms or LED indicators during intervention. Analyzing the average measured internal force in Table 6-2, operators used slightly less force in Case 5 than in Case 3. Furthermore, Case 5 had fewer failures (3 times less, see Figure 6-8a) than Case 3. While neither are statistically significant differences, the data trends indicate that providing a haptic cue that more closely resembles the operator's desired force can improve operator performance.

6.2.2.2 Additional Statistical Observations

The initial statistical analysis performed makes several assumptions that can be validated with a more complex analysis. For the ANOVA tests performed, the data collected for each case were based on the average performance of each subject. It is assumed that the "testing device" used to gauge each case is the same. However, the "testing device" for our telemanipulation system is based on human subject performance, and each subject has different innate abilities. Therefore, to ensure our results are not an artifact of subject-to-subject variation, a more complex analysis can control for the fact that the subjects are only a sample drawn from a larger population. A two-factor ANOVA test was set up such that cases are considered the fixed factor (i.e., a factor whose specific levels are the ones of interest) and the subjects are considered a random factor (i.e., the factor levels are samples of a larger population). This type of test is known as a mixed-model analysis. Again, for the fixed effects (the cases), the null hypothesis is that all the means are the same. However, for the random effects, the null hypothesis is that variability between effects is equal to zero. If the subject-to-subject variability is a significant source of random variation it must be taken into account when comparing the fixed-effects.

Using the SAS/STAT statistical software package, a mixed-model analysis was completed. To include the random effects, it was necessary to include each subject's individual trial data (i.e., the trial mean measured internal force), excluding trials that were marked as failures. A total of 274 observations were used, with a maximum of four observations per subject per case. The results again showed that there was a significant difference among the case means for measured internal force (a p-value less than 0.0001). The random variation due to subject-to-subject variability was found to be statistically signifi-

Table 6-5. Results of the Tukey multi-comparison analysis on the mixed-model adjusted case mean data.[a]

Case (ordered)	1	3	2	5	7	6	4
L.S. Mean Force [N]	2.374	2.359	2.286	2.227	2.000	1.985	1.979
	A	**A**	**A**	**A**			
			B	**B**	**B**		
				C	**C**	**C**	**C**

a. Cases with the same letter are *not* statistically different.

cant. Therefore, the random variation must be accounted for when using a multi-comparison procedure to determine which cases are statistically different. Similar to the results for the one-factor ANOVA, using Dunnett's method for comparison to a control on the adjusted data showed Cases 4, 6, and 7 to be statistically different from Case 1 at the 95% confidence level. Thus, accounting for the subject-to-subject variability, the statistical results for the cases are unaffected. Interestingly, using a multi-comparison procedure known as the Tukey method, it is also possible to make comparisons among all cases. Similar to Dunnett's method, the Tukey method ensures that the Type-I error is bounded to the 95% confidence level. Table 6-5 shows the results from a Tukey multi-comparison analysis on the adjusted mixed-model data. The cases are ordered based on the adjusted means, and cases with the same letter are *not* statistically different. Thus we can see that Cases 4, 6, and 7 have means statistically different from Cases 1 and 3, which gives a little more information about case differences than a normal Dunnett's test.

6.2.3 Model Description and Analysis

Another useful tool for data analysis is to develop a model to describe the observed data. In this way, a small set of parameters can be applied to a model equation that attempts accurately to estimate the actual data. A parameterized description can yield insight into the different trends seen in the observed data. More importantly, we can then look at the residuals of the model prediction (as compared to the observed data) to see how "good" the model is and if there are any additional parameters that should be added to the model equation, e.g., learning or fatigue effects.

6.2.3.1 Model Description

To investigate the differences in performance for each of the cases, the model will ideally describe the data with a single parameter representing each case. Thus, case-to-case comparisons can be made by a comparison of the describing parameters. There are many possible ways to describe the differences in the observed case data. We chose to use the averaged percent difference in measured internal force for each case as compared to Case 1. The percent difference is computed based on each subject's *mean* internal force for each of the non-control cases and the subject's mean internal force for Case 1:

$$\phi_{(i,j)} = \frac{f_{int(i,j)} - f_{int(1,j)}}{f_{int(1,j)}} \tag{6.2}$$

where $f_{int(i,j)}$ is the mean measured internal force for case i (i = 1,...,7) and subject j (j = 1,...,11) and $\phi_{(i,j)}$ is the percent difference for cases 2-7 compared to Case 1 for each subject ($\phi_{(1,j)}$ will always equal zero). The percent difference values of all the subjects for a given case are then averaged to form an single parameter that describes the given case in a meaningful way:

$$\overline{\phi_{(i)}} = \sum_{j=1}^{N} \frac{\phi_{(i,j)}}{N} \tag{6.3}$$

where N is the total number of subjects (eleven in our experiment). The percent difference for each subject was specifically chosen to help minimize the effects of subject-to-subject variability in the model parameters. The six averaged percent difference parameters can then be multiplied by each subject's Case 1 mean internal force to formulate an estimate of observed data. Thus, our parameter based model equation is:

$$f_{int,estimate(i,j)} = (1 + \overline{\phi_{(i)}}) \cdot f_{int(1,j)} \tag{6.4}$$

for $i = 1,...,7$.

To compare the cases, we analyze the mean percent difference (our model parameter $\overline{\phi_{(i)}}$) and the standard deviation of $\phi_{(i,j)}$ for all subjects for each case. From Figure 6-9, the averaged percent difference in internal force for Case 6 is roughly -15% with two standard deviations on either side of the mean that does not include zero. If we assume the data

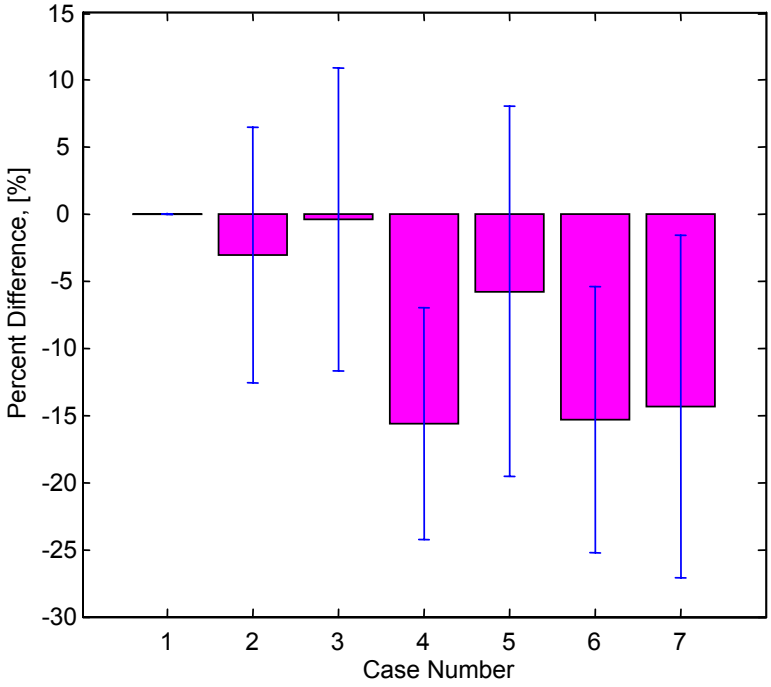

Figure 6-9. Average of each subject's percent difference in mean internal force for each case compared to Case 1. Cases 4, 6, and 7 have an average percent reduction in the mean internal force of approximately 15% as compared to Case 1. Given the value of the means and the standard deviation for these cases, the means are significantly lower than Case 1.

are normally distributed, the empirical rule that 95% of the data falls within two standard deviations of the mean tells us that the Case 6 mean is significantly lower than zero, thus percent difference in force is significantly different than Case 1 (this also holds for Cases 4 and 7, which matches our earlier statistical results). This implies that, on average, subjects used less force to complete Case 6 than Case 1 and supports our hypothesis based on the previously defined performance metrics for delicate object handling.

6.2.3.2 Model Analysis

For our given model, we must ask: How well does the model describe the data? To best answer this question we can perform an analysis of the model residuals. The model residuals are based on the difference between the observed data for each subject and case, $f_{int(i,j)}$, and the estimated value for the mean internal force for each subject and case computed using eq. 6.4, $f_{int,estimate(i,j)}$:

$$residual_{(i,j)} = f_{int(i,j)} - f_{int,estimate(i,j)} \tag{6.5}$$

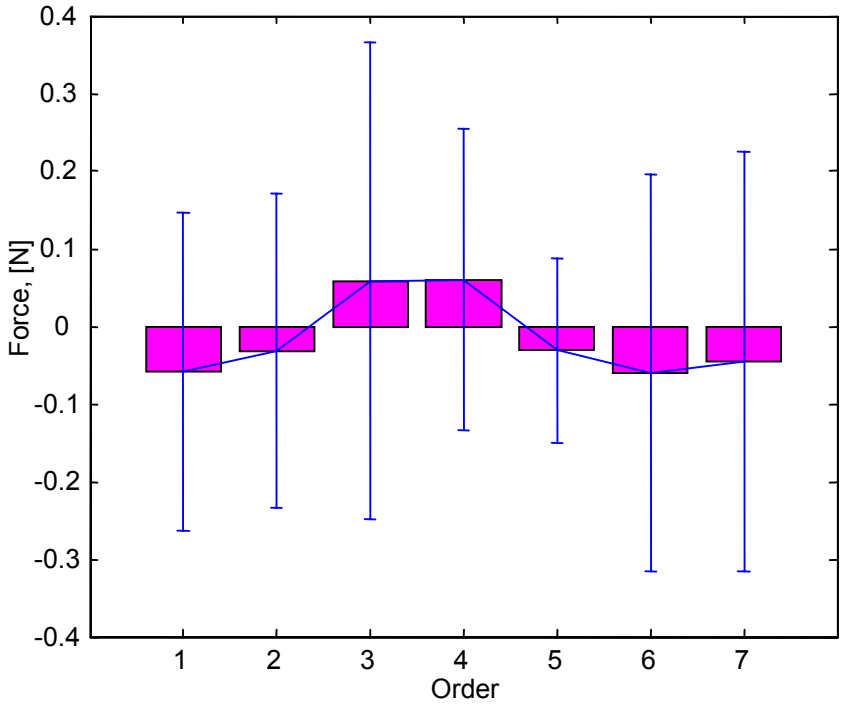

Figure 6-10. A plot of the model residuals for all subjects based on the order of task completion. For example, the residuals from all cases that occurred first are grouped together and the mean (shown with two standard deviation errors bars) is shown for Order 1. All cases that occurred second are plotted for Order 2, and so on. Clearly, the standard deviation is much larger than any of the means, indicating that there is not a significant effect due to learning or fatigue.

If all the residuals equal zero, then the model perfectly describes the data. For this model, the mean of the all the residuals is -0.01 N with a standard deviation of 0.24 N, both fairly small values indicating a good model. However, investigating potential sources of trends in the residual data can improve the model and create a better description of the observed data.

To determine if learning or fatigue plays a significant role in the observed data, we can group the residuals according to task order. Suppose the means of the residuals for all cases that occurred first were significantly higher than zero, then we could state that effects due to learning are present and are quantified by the given mean. Figure 6-10 shows a plot of the means of the residuals for all cases that occurred first, second, third, and so on. Also shown are error bars of two standard deviations. Clearly, the standard deviations are much larger than the mean residuals values, thus we can confidently state that any learning or fatigue effects were not significant.

6.2.4 Subjective Data Analysis

In addition to having the subjects complete the specified task to test the shared control system, a post-experiment questionnaire was administered to obtain qualitative data on the cases tested. Because the human operator is an integral part of the shared telemanipulation system, the expressed preference of the operator is an important parameter in assessing the overall effectiveness of a given case. The complete questionnaire can be found in Appendix E.

Subjects were asked to rank each case tested based on preference. The ranking was on a scale from negative two to positive two, corresponding to "disliked" and "preferred," respectively, with zero being "indifferent." Figure 6-11 shows the average ranking score for each case, for all subjects, with the errors bars indicating two standard deviations. From the figure, we can see that Cases 6 and 7 were most preferred.

A simple statistical analysis was performed to determine which cases had a mean preference ranking greater than zero. A t-test was used to find the p-value associated with each case (see Table 6-6). The table below shows the results from the hypothesis testing.

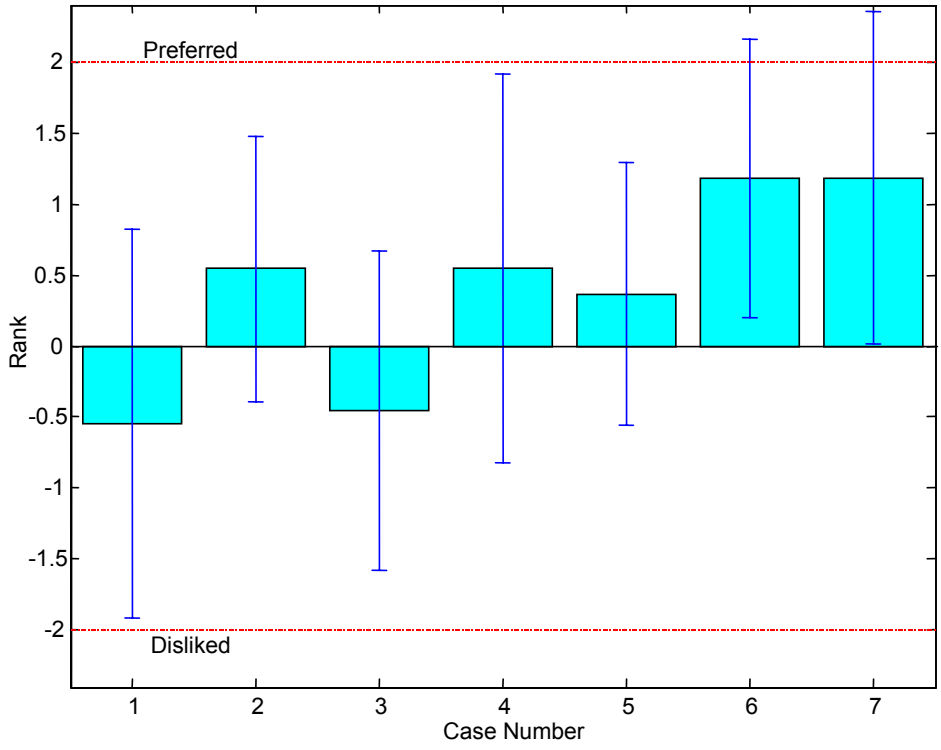

Figure 6-11. Averaged ranking of each case in terms of subjects' expressed preference. Error bars represent two standard deviations

Table 6-6. Case preference average ranking with statistical analysis results.

Case	1	2	3	4	5	6	7
Ave. Ranking	0.55	0.55	-0.45	0.55	0.36	1.18	1.18
p-value	0.89	0.04	0.89	0.11	0.11	0.00	0.00
Statistically significant	NO	YES	NO	NO	NO	YES	YES

At the 95% confidence level (p-value must be less than 0.05), Cases 2, 6, and 7 have a ranking greater than zero, indicating that subjects preferred these cases. While clearly some subjects preferred Case 2, the average ranking is much lower than for Cases 6 and 7.

Subjects were also asked to rank each case in terms of ease or difficulty based on the task goals. Even though this question may seem very similar to the question about case preference, we found that during preliminary testing some subjects realized that a given case might make the task easier but may not be the most preferred. Asking the two questions allows us look into the idea that while some features might be helpful (such as audio alarms), the features could also prove to be annoying to the operator.

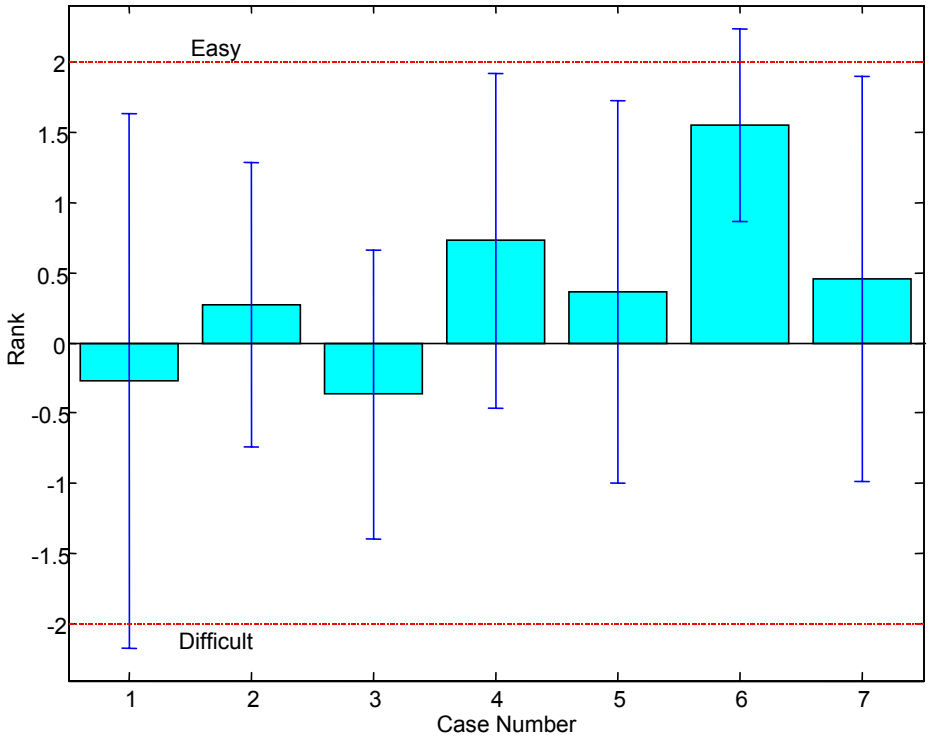

Figure 6-12. Averaged ranking of each case in terms of subjects' expressed ease in terms of completing the task goals. Error bars represent two standard deviations.

Table 6-7. Case ease average ranking with statistical analysis results.

Case	1	2	3	4	5	6	7
Ave. Ranking	-0.23	0.23	-0.36	0.73	0.36	1.55	0.45
p-value	0.68	0.20	0.87	0.04	0.20	0.00	0.16
Statistically significant	NO	NO	NO	YES	NO	YES	NO

The ranking for case ease was on a scale from negative two to positive two corresponding to "easy" and "difficult," respectively, with zero being "no difference." Figure 6-12 shows the average score for each case, for all subjects, with error bars indicating two standard deviations. For the figure, we can see that subjects found Cases 4 and 6 to be the easiest, while Cases 1 and 3 were considered to be the most difficult. Again we can perform a t-test on the data for each case to determine if the ease ranking is greater than zero. The results are shown in Table 6-7.

Subjects were also questioned about the different features of the shared controller. Specifically, operators were asked to rank the effect of the LEDs, audio tones, and robot intervention on task ease in the applicable cases. The LEDs were used in Cases 4, 6, and 7 to indicate that the robot was assuming control over the grasp force regulation. The audio tones were used in Cases 2, 4, and 6 to indicated either excessive force or that the operator was about to drop the object (or command the robot to release the object). Robot intervention could occur in Cases 3-7 if the operator's desired force dropped below a threshold based on the minimum internal force necessary to carry the object. Again, the ranking of the different shared control features was on a scale from negative two to positive two corresponding to "easy" and "difficult," respectively, with zero being "no difference." An additional question was asked regarding the reduced forces fed back during robot intervention in Cases 5, 6, and 7. Subjects were asked if the reduced force feedback helped or hindered task completion. The ranking was on a scale from negative two to positive two, corresponding to "helped" and "hindered," respectively, with zero being "indifferent." The results from these four questions are shown in Figure 6-13. A t-test was performed on the data for each question to determine if the rank is significantly different from zero. Each

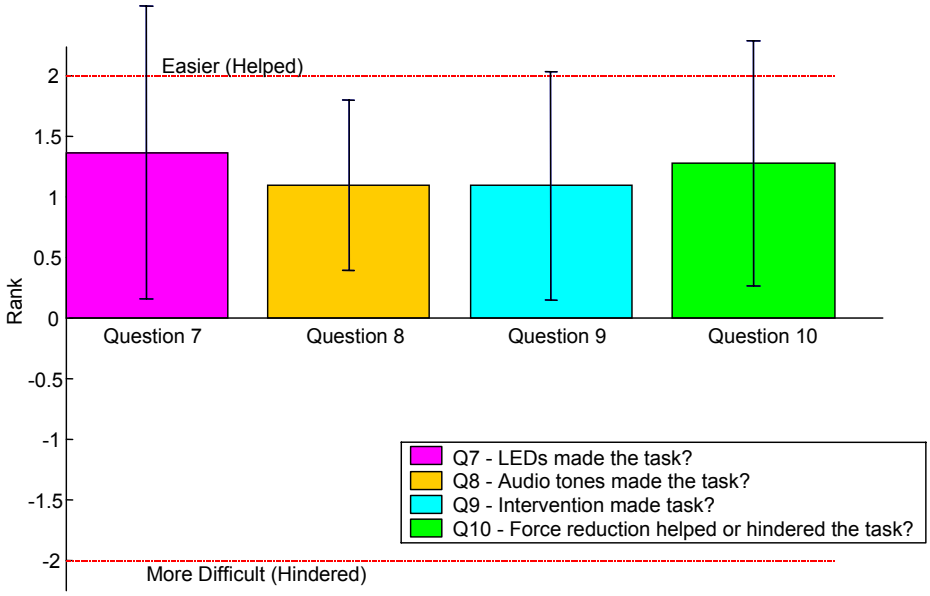

Figure 6-13. Averaged ranking for questions concerning the shared control features and feedback methods. Error bars represent two standard deviations.

question had a *p*-value less than 0.005 indicating that the question results were significantly different from zero. The average rank for each of the features is very high, indicating that operators found the various feedback methods and robot intervention made the task easier.

6.2.4.1 Questionnaire Conclusions

The results from the subject post-experiment questionnaire complement the results found through the objective data analysis. The case that subjects preferred the most and found the easiest was Case 6, which corresponds with the objective data analysis. Also, subjects felt the shared control features made the task easier. Case 6 implemented all of these features. Interestingly, the two cases which subjects disliked the most and found the most difficult to use were the same cases that had the highest applied internal force on the object, i.e., Cases 1 and 3.

6.3 Conclusions

This chapter has covered the details of our experiment to test the addition of a dexterous shared controller to a traditional bilateral telemanipulation system. Our hypothesis was that a shared controlled system would improve an operator's performance during task execu-

tion. The task was designed to test the sharing of control over grasp force regulation between the operator and the remote robot.

For the delicate object handling task, the objective data analysis conclusively shows that the addition of a shared controller reduced the forces applied to the object and decreased the number of object drops. Additionally, the post-experiment questionnaire showed that subjects preferred the shared controller with direct and indirect feedback and found it easier to use than all other cases.

The primary questions that the experiment sought to answer were introduced at the start of this chapter. Based on the objective and subjective data analysis results, we can now answer each of these questions.

Task performance is improved if an operator is warned of a possible failure through indirect feedback. By considering a failure as dropping the object or squeezing the object too hard, the operator was warned with a high frequency and low frequency tone. The indirect feedback effect was isolated when comparing Case 2 to the control case (Case 1). The mean internal force for Case 2 was less than the control case but not by a statistically significant amount. The two audio alarms were not applied separately so the individual effects from the two different alarms can not isolated. However, based on the typical plots of subject data, we believe that providing the operators with the excessive force alarm helps at the start of the trials to reduce the force to a reasonable range. Then, in the latter parts of the trial, the subjects depended on the high frequency alarm to prevent dropping the object and enabled a lower average force than the control case.

We also found that for some of the cases in which the robot could intervene, task performance, in terms of minimizing the internal force and limiting the number of accidental drops, improved. However, the presence and type of direct and indirect feedback had a significant effect on the operator's performance. The cases which informed the operator that the intervention was occurring (Cases 4, 6, and 7) had significantly lower measured internal forces than the control case (Case 1), while the mean internal forces for Cases 3 and 5 did not. This shows that it was important to inform the operator that the robot has assumed control. However, if we also consider the number of failures that occurred, simply

informing the operator that intervention was occurring using LED as a visual indicator was not sufficient for improving task performance.

During intervention, the robot was constantly monitoring the master input to determine the operator's intent, i.e., if the operator desired to release the grasped object. The results show that it was helpful to feed back information that the shared controller was about to trigger a state change in the robot's control, i.e., to release the object. In other words, task performance was improved for cases in which the high frequency audio tone was used to alert the operator that an object release was close to being triggered. This effect is isolated when comparing Case 6 to Case 7, in which Case 7 did not have any audio tones. The number of failures for Case 7 versus Case 6 was much higher. Less directly, we can also compare the mean internal force for Cases 4 and 6 to that of Cases 3 and 5 (cases with intervention but no audio alarms), respectively. The mean internal forces for Cases 4 and 6 were much lower than in Cases 3 and 5. However, it should be noted that in Cases 4 and 6, the excessive force low frequency audio tone was also used. While the alarm effects are not expressly isolated, the operators typically only used the excessive force alarm at the start of the trial (similar to Case 2 trials). Once the operator decreased the desired force enough for the robot to assume control, the operator very rarely triggered the excessive force alarm.

With haptic feedback, the sharing of a force control task was also investigated by the different case features. During robot intervention, different methods for feeding back the robot's measured force to the operators' fingertips were used. Specifically, in Cases 3 and 4, the force fed back to each human finger was based on the magnitude to the robot's finger. During periods of intervention, the robot applied a nearly constant force to the object thus the forces fed back were nearly constant as well. However, in Cases 5, 6, and 7, the forces fed back were based on the operators' grasp size. As an operator relaxed and open his or her grasp, the forces fed back were reduced in proportion. Reducing the feedback force, based on the operator's desired internal force, improved task performance. Comparing Case 3 to Case 5, which isolated this effect, the number of object drops for Case 5 was half that of Case 3. The same trend was also observed when comparing Cases 4 to 6. Additionally, the average internal forces were slightly less in Case 5 than in Case 3.

Finally, based on the objective data and subjective data analyses, the combination of robot intervention, audio alarms, LED indicator, and reduced force feedback in Case 6 led to the best overall performance compared to the bilateral control case. Not only did subjects perform better using Case 6, but the subjects also preferred the case the most and found it easiest to use.

7 Conclusions and Future Work

The main focus of the work presented in this thesis is the development and evaluation of a shared control dexterous telemanipulation system with multi-modal and haptic feedback. This chapter discusses and summarizes the results of the research work described in the previous chapters. The major contributions of this work are reviewed and suggestions for future work are discussed.

7.1 Summary and Results

In support of the development of a human hand-based telemanipulation system, a human-to-robot mapping method was developed (see Chapter 4). The method is based on capturing the intended motions of a virtual object within the operator's hand. Motions of the virtual object are used to compute commanded motions for the robotic hand. In contrast to using a simple planar projection of the operator's fingertip positions, the virtual object method utilizes the three-dimensional fingertip data to define a grasped virtual object between the fingertips, yielding additional information about the intent of the operator's manipulation motions. The virtual object mapping method provides a solution to the problems associated with the kinematic and workspace differences between the human hand and the planar robot hand. The parameters of the virtual object can be independently scaled and modified to account these differences. The virtual object mapping method allows operators to easily grasp, manipulate, and release objects with the slave robot hand using the glove-based interface.

As discussed in Chapter 5, the shared control telemanipulation system incorporates the high level and remote autonomy of supervised systems with the telepresence found in direct control of (and feedback supplied by) master-slave telemanipulation systems. The

incorporation of low level “intelligence” for securely manipulating objects can reduce the demands on an immersive telemanipulation system.

A set of experiments, described in Chapter 6, was designed to determine whether shared control can improve the ability of an operator to remotely handle objects *delicately* and *securely* and to determine what combinations of force, visual, and audio feedback provide the best performance and operator sense of presence.

Several effects and combinations of effects were tested by having operators perform a fragile object handling task using the telemanipulation system for a variety of different cases. Each subject (eleven subjects total) performed a pick-and-place handling task several times for seven different cases. The cases were designed to isolate potential performance differences due to the given effects. The effects included the use of robot intervention to aid the operator in maintaining a delicate grasp on the object and the use of audio alarms to warn an operator of a possible failure or impending state changes in the robot’s control. Visual indicators were used to inform the operator that the robot system was intervening. Additionally, different methods of feeding back forces to the operator during robot intervention were tested.

The performance of each subject during the fragile object handling task was evaluated through an objective data analysis. The average measured internal force applied to the object was the primary metric for the evaluation of a given case. In addition to observing the internal force, the number of failures (i.e., the number of times the object was dropped) was also considered an important indication of task performance. Utilizing results from a statistical analysis of each subject’s measured internal force for each case and the number of failures for each of the given cases, several conclusions can be drawn. We found that the addition of a dexterous shared control framework could improve task performance by improving an operator’s ability to delicately and securely handle an object compared to direct telemanipulation. However, in comparing the performance of all seven cases, we found that it is necessary to:

- inform the operator when the intervention is active (in other words, it is necessary to let the operator know that the robot has assumed control).

- inform the operator of impending state changes (in particular, inform the operator that the robot may release an object in its grasp if the operator's commands continue to diverge from the robot's commands).
- feedback forces to the operator based on the operator's commanded force rather than feeding back the actual forces as measured by the robot during robot intervention.

In addition to evaluating the shared control system based on recorded trial data, a post-experiment questionnaire was given to each subject to obtain each subject's expressed preference and perceived difficulty for each of the cases tested. Based on a statistical analysis of the results from the questionnaire, we found that subjects generally preferred shared control with multiple forms of feedback (following the requirements listed above) and found it easiest to use.

7.2 Review of Contributions

The major contributions of this thesis are summarized here.

- Development of a shared control framework for dexterous telemanipulation. While many parts of this system have been covered in other literature, shared control applied to a system with an unencumbering glove-based interface, fingertip force feedback, and a robot hand with force and tactile sensors has not.
- Investigation of an experimental shared control telemanipulation system. A set of human subject experiments was completed to determine if the addition of a dexterous shared controller to a traditional bilateral system could improve an operator's performance during task execution. The results demonstrate the benefits of shared control and the need to choose carefully the types and methods of direct and indirect feedback.
- Development of a human-to-robot mapping method to support the use of a glove-based interface for intuitive object manipulation with a non-anthropomorphic robot hand. The method captures the intended motions of a virtual object grasped in the operator's hand and uses computed virtual object motions to create commanded

motions for the robot. Kinematic and workspace dissimilarities between the human hand and our planar robot hand are compensated for by independently modifying and scaling relevant virtual object parameters.

7.3 Suggestions for Future Work

There are several interesting directions for future work in the areas of research presented in this thesis. The directions range from direct extensions of the research to applying the fundamental ideas to other disciplines.

One possible avenue of future work is the extension of the virtual object human-to-robot mapping method to other non-anthropomorphic planar robot hands. In general, it is difficult to design and develop fully anthropomorphic robot hands. However, there are many advantages to building a robot hand capable of object manipulation. A compromise to complex anthropomorphic designs are multi-fingered planar hands similar to the one used for the research presented in this thesis (see Chapter 3 for details). While there may be some similarities, depending on the design, differences in kinematics between human and robot must be accounted for. Furthermore, preferred workspace utilization and range-of-motion may also differ significantly. The basic steps of the virtual object mapping method outlined in Chapter 4, Section 4.4.2 on page 65, are extensible to other planar manipulators (e.g., a cartesian prismatic hand). In the case of our planar robot with two, two degree-of-freedom, fingers, the virtual object mapping method gives the operator intuitive control over the motion of the robotic fingers despite kinematic and workspace differences.

Another direct extension of the work would be to perform experiments in shared control telemanipulation for other types of tasks. The fragile object handling experiment described in Chapter 6 was chosen to test several issues associated with sharing control over the internal force of a grasped object during task execution. However, the control framework can support the sharing of control over other aspects of object manipulation. The basic framework supports sharing of control at both high levels and low levels, which could be tested with a more complex manipulation task. For example, in an object assembly task, such as a peg-in-hole insertion task, the robot could modify the impedance of the object based on local sensor information. The cooperative object impedance controller could also support the implementation of a remote center-of-compliance. In terms of shar-

ing control, the operator could maintain position control and the operator's orientation commands could be summed with information task space based commands computed by the robot. Additionally, because the current system lacks the ability to properly display interaction forces with the environment, other methods of feedback could be used to convey contact forces during task execution. High level commands, such as short duration autonomous tasks, could be initiated by the operator or the remote system based on task and sensor feedback.

The experimental results discussed in this thesis clearly demonstrate the benefits associated with shared control in dexterous telemanipulation and need to consider the methods and types of feedback given to the operator. In extensions to shared control, many of the same issues investigated in this thesis will have to revisited. In an assembly tasks, if the master system can not accurately display the quantities computed and measured by the slave robot, what is the most effective method to display this information? In the cases where the slave manipulator is capable of short duration autonomous tasks, what type of feedback (visual, audio, and haptic) is necessary to ensure the operator's sense of presence is not significantly diminished.

Another interesting area for future work is in the investigation of limitations associated with the haptic feedback system. In certain situations, there is the possibility of doing work on the operator in a non-conservative way. For example, suppose an operator is grasping an object with the slave hand and receiving forces via the CyberGrasp system. Assume that the operator is holding an object against a surface in the remote world and maintaining a constant interaction force. Then, as the operator rotates the object about one corner, this does no work against the world. However, based on the method in which external and internal forces are fed back to the operator, there is the possibility that work can be done on the operator in a non-conservative way, leading to potentially unstable haptic interaction. Further investigation could determine if this will pose a serious problem (especially during complex assembly tasks) and if there are practical methods for overcoming it.

The idea of shared control could also be extended to other applications. For example, a drive-by-wire automobile is essentially a teleoperated system where the operator (the driver) is interacting with the environment (the road) through a master-slave system (the

car). With drive-by-wire capability, researchers have developed lane-keeping assistance systems that add computed steering commands to driver commands to help the driver stay on the road [Rossetter 2003]. Similar to shared control for dexterous telemanipulation, many of the same shared control issues will have to be investigated. For example, what is the best way to determine the driver's intent for situations such as lane changing? If the system is also used for obstacle avoidance, will it be necessary to inform the driver that the system is taking control? Also, what will be the most effective feedback modalities?

7.4 Conclusion

This thesis has described an immersive dexterous telemanipulation system with a shared control framework. The main contributions of the work are in the development and investigation of the shared control framework for a dexterous telemanipulation task. The experimental results demonstrated the benefits of shared control and the need to choose carefully the methods and types of feedback.

Bibliography

Aigner, P. and McCarragher, B.J. [2000] "Modeling and constraining human interactions in shared control utilizing a discrete event framework." IEEE Transactions on Systems, Man, and Cybernetic, Volume 30, Issue 3, pages 369-379, 2000.

Ambrose, R.O., Aldridge, H., Askew, R.S., Burridge, R.R., Bluethmann, W., Diftler, M., Lovchik, C., Magruder, D. and Rehnmark, F. [2000] "Robonaut: NASA's space humanoid." IEEE Intelligent Systems and Their Applications, Volume 15, Issue 4, pages 57-62, 2000.

An, K.N., Chao, E.Y., Cooney, W.P., and Linscheid, R.L. [1979] "Normative Model of Human Hand for Biomechanical Analysis," Journal of Biomechanics, Volume 12, pages 775-788, 1979.

Anderson, R.J. and Spong, M.W. [1989] "Bilateral control of teleoperators with time delay." IEEE Transactions on Automatic Control, Volume 34, Issue 5, pages 494-501, 1989.

Arcara, P. and Melchiorri, C. [2002] "Control schemes for teleoperation with time delay: A comparative study." Robotics and Autonomous Systems, Volume 38, pages 49-64, 2002.

Backes, P.G. [1992] "Multi-sensor based impedance control for task execution." Proceedings of the IEEE International Conference on Robotics and Automation, Volume 2, pages 1245-1250, 1992.

Backes, P.G. and Tso, K.S. [1990] "UMI: an interactive supervisory and shared control system for telerobotics." Proceedings of the IEEE International Conference on Robotic and Automation, Volume 2, pages 1096-1101, 1990.

Bennet, D. and Needles, A. [1997] "A new control concept for commercially available telerobotic manipulators." American Nuclear Society 7th Topical Meeting on Robotics and Remote Systems, 1997.

Bicchi, A., Salisbury, J.K. and Brock, D.L. [1993] "Contact sensing from force measurements." International Journal Of Robotics Research, Volume 12, Number 3, pages 249-262, 1993.

Boshra, M. and Zhang, H. [2000] "Localizing a polyhedral object in a robot hand by integrating visual and tactile data." Pattern Recognition, Volume 33, Number 3, pages 483-501, 2000.

Brock, D.L. [1988] “Enhancing the dexterity of a robot hand using controlled slip.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 249-251, 1988.

Brunner, B., Arbter, K. and Hirzinger, G. [1994] “Task directed programming of sensor based robots.” Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems, Volume 2, pages 1080-1087, 1994.

Burdea, G.C. [1996] *Force and Touch Feedback for Virtual Reality.* New York, NY: John Wiley & Sons, Inc., 1996.

Burdea, G.C. and Zhuang, J. [1991] “Dexterous telerobotics with force feedback -- an overview. Part 2: Control and implementation.” Robotica, Volume 9, pages 291-298, 1991.

Buss, M., Hashimoto, H. and Moore, J.B. [1996] “Dextrous hand grasping force optimization.” IEEE Transactions on Robotics and Automation, Volume 12, Issue 3, pages 406-418, 1996.

Buss, M. and Schlegl, T. [2000] “A discrete-continuous control approach to dextrous manipulation.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 276-281, 2000.

Cannon, D. and Thomas, G. [1997] “Virtual Tools for Supervisory and Collaborative Control of Robots.” Presence: Teleoperators and Virtual Environments, Volume 6 Number 1, pages 1-28, 1997.

Cavusoglu, M.C., Sherman, A., and Tendick, F. [2001] “Bilateral controller design for telemanipulation in soft environments.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 1045 -1052, 2001.

Charlebois, M., Gupta, K., Payandeh, S. [1996] “Curvature based shape estimation using tactile sensing.” Proceeding of the IEEE International Conference on Robotics and Automation, Volume 4, pages 3502-3507, 1996.

Colgate, J.E. [1993] “Robust impedance shaping telemanipulation.” IEEE Transactions on Robotics and Automation, Volume 9, Issue 4, pages 374-384, 1993.

Daniel, R.W. and McAree, P.R. [1998] “Fundamental limits for force reflecting teleoperation.” International Journal of Robotics Research, Volume 17, Number 8, pages 811-830, 1998.

Das, H. Zak, H., Kim, W.S., Bejczy, A.K., and Schenker, P.S. [1992] “Operator Performance with Alternate Manual Control Modes in Teleoperation,” Presence: Teleoperators and Virtual Environments, Volume 1, Number 2, pages 201-218, 1992.

Devore, J.L. and Farnum, N.R. [1999] *Applied Statistics for Engineers and Scientists.* Pacific Grove, CA: Duxbury Press, 1999.

Edin, B., Howe, R.D., Westling, G., and Cutkosky, M.R. [1993] "Relaying frictional information to a human operator by physiological mechanisms." IEEE Transactions on Systems, Man, and Cybernetics, Volume 23, Number 2, pages 427-432, 1993.

Eusebi, A. and van der Ham, A.C. [1994] "Experimental verification of bilateral force/position control schemes using the communication network analogy." Proceedings of the 33rd IEEE Conference on Decision and Control, Volume 3, pages 2402-2407, 1994.

Fearing, R.S. [1990] "Tactile sensing mechanisms." International Journal of Robotics Research, Volume 9, Number 3, pages 2-23, 1990.

Fischer, M., van der Smagt, P., and Hirzinger, G. [1998] "Learning techniques in a dataglove based telemanipulation system for the DLR hand." IEEE Proceedings of the International Conference on Robotics and Automation, Volume 2, pages 1603-1608, 1998.

Fite, K.B., Speich, J.E., and Goldfarb, M. [2001] "Transparency and Stability robustness in two-channel bilateral telemanipulation." Journal of Dynamic Systems, Measurement, and Control - Transactions of the ASME, Volume 123, Number 3, pages 400-407, 2001.

Griffin, W.B., Findley, R.P., Turner, M.L., and Cutkosky, M.R. [2000] "Calibration and mapping of a human hand for dexterous telemanipulation." Proceedings of the ASME International Mechanical Engineering Conference and Exposition, Dynamic Systems and Controls, Volume 69, Number 2, pages 1145-1152, 2000.

Goertz, R.C. and Thompson, R.C. [1954] "Electronically controlled manipulator." Nucleonics, pages 46-47, 1954.

Han, L., Trinkle, J.C. and Li, Z. [1999] "Grasp analysis as linear matrix inequality problems." Proceedings of the IEEE International Conference on Robotics and Automation, Volume 2, pages 1261-1268, 1999.

Hannaford, B. [1989] "Stability and performance tradeoffs in bi-lateral telemanipulation." Proceedings of the IEEE International Conference on Robotics and Automation, Volume 3, pages 1764-1767, 1989.

Hannaford, B., Wood, L. McAffee, D.A., and Zak, H. [1991] "Performance evaluation of a six-axis generalized force-reflecting teleoperator." IEEE Transactions on Systems, Man, and Cybernetics, Volume 21, Issue 3, pages 620-633, 1991.

Hashtrudi-Zaad, K. and Salcudean, E. [2002] “Analysis of control architectures for teleoperation systems with impedance/admittance master and slave manipulators.” International Journal of Robotics Research, Volume 20, Number 6, pages 419-445, 2001.

Hayati, S. and Venkataraman, S.T. [1989] “Design and implementation of a robot control system with traded and shared control capability.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 3, pages 1310-1315, 1989.

Hirzinger, G., Butterfass, J., Fischer, M., Grebenstein, M., Hahnle, M., Liu, H., Schaefer, I., and Sporer, N. [2000] “A mechatronics approach to the design of light-weight arms and multifingered hands.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 46-54, 2000.

Hogan, N. [1985a] “Impedance control - An approach to manipulation: Part I. Theory.” Journal of Dynamic Systems, Measurement, and Control - Transactions of the ASME, Volume 107, Number 1, pages 1-7, 1985.

Hogan, N. [1985b] “Impedance control - An approach to manipulation: Part II. Implementation.” Journal of Dynamic Systems, Measurement, and Control - Transactions of the ASME, Volume 107, Number 1, pages 8-16, 1985.

Hong, J. and Tan, X. [1989] “Calibrating a VPL DataGlove for teleoperating the Utah/MIT hand.” IEEE Proceedings of the International Conference on Robotics and Automation, Volume 3, pages 1752 -1757, 1989.

Howe, R.D. [1992] “A force-reflecting teleoperated hand system for the study of tactile sensing in precision manipulation.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 2, pages 1321-1326, 1992.

Howe, R.D. and Cutkosky, M.R. [1993] “Dynamic tactile sensing: perception of fine surface features with stress rate sensing.” IEEE Transactions on Robotics and Automation, Volume 9, Issue 2, pages 140-151, 1993.

Howe, R.D. and Cutkosky M.R. [1996] “Practical force-motion models for sliding manipulation.” International Journal of Robotics Research, Volume 15, Number 6, pages 557-572, 1996.

Howe, R.D. and Kontarinis, D. [1992] “Task performance with a dexterous teleoperated hand system.” Proceedings of SPIE, Telemanipulator Technology, Volume 1833, pages 199-207, 1992.

Hyde, J.M. [1995] *A Phase Management Framework for Event-driven Dextrous Manipulation*. Ph.D. Thesis, Stanford University, Department of Mechanical Engineering, 1995.

Fearing, R.S. [1990] “Tactile sensing mechanisms.” International Journal of Robotics Research, Volume 9, Number 3, pages 3-23, 1990.

Ferrell, W. R. and Sheridan, T. B. [1967] “Supervisory control of remote manipulation.” IEEE Spectrum, October, pages 81-88, 1967.

Kang, S.B. and Ikeuchi, K. [1997] “Toward automatic robot instruction from perception-mapping human grasps to manipulator grasps.” IEEE Transactions on Robotics and Automation, Volume 13, Issue 1, pages 81-95, 1997.

Kao, I. and Cutkosky, M.R. [1993] “Comparison of theoretical and experimental force/motion trajectories for dexterous manipulation with sliding.” International Journal of Robotics Research, Volume 12, Number 6, pages 529-534, 1993.

Kerr, J. and Roth, B. [1986] “Analysis of multifingered hands.” International Journal of Robotics Research, Volume 4, Number 4, pages 3-17, 1986.

Khatib, O. [1987] “Unified approach for motion and force control of robot manipulators: the operational space formulation.” IEEE Journal of Robotics and Automation, Volume 3, Issue 1, pages 43-53, 1987.

Koide, S., Andou, H., Suzuki, G., and Oda, M. [1993] “Semi-autonomous teleoperation control system with layered architecture-An application to space robots.” Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Volume 3, pages 1995-2000, 1993.

Kontarinis, D.A. and Howe, R.D. [1995] “Tactile display of vibratory information in teleoperation and virtual environments.” Presence: Teleoperators and Virtual Environments, Volume 4, Number 4, pages 387-402, 1995.

Kumar, V. and Waldron, K.J. [1989] “Suboptimal algorithms for force distribution in multifingered grippers.” IEEE Transactions of Robotics and Automation, Volume 5, Issue 4, pages 491-498, 1989.

Kyriakopoulos, K.J., Van Riper, J., Zink, A., and Stephanou, H.E. [1997] “Kinematic analysis and position/force control of the Anthrobot dextrous hand.” IEEE Transactions on Systems, Man and Cybernetics, Part B, Volume 27, Issue 1, pages 95-104, 1997.

Lawn, C.A. and Hannaford, B. [1993] “Performance testing of passive communication and control in teleoperation with time delay.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 3, pages 776 -783, 1993.

Lawrence, D.A. [1993] “Stability and transparency in bilateral teleoperation.” IEEE Transactions on Robotics and Automation, Volume 9, Issue 5, pages 624 -637, 1993.

Leveroni S., Shah V., and Salisbury K.J. [1998] “Toward dexterous gaits and hands.” Lecture Notes In Control And Information Sciences, Volume 232, pages 67-78, 1998.

Li, L., Cox, B., Diftler, M., Shelton, S., and Rogers, B. [1996] “Development of a telepresence controlled ambidextrous robot for space applications.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 58-63, 1996.

Liu, Y., Hoover, A., and Walker, I.D. [2001] “Sensor network based workcell for industrial robots.” Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Volume 3, pages 1434-1439, 2001.

Lovchik, C.S. and Diftler, M.A. [1999] “The Robonaut hand: a dexterous robot hand for space.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 2, pages 907-912, 1999.

Maekawa, H., Tanie, K., and Komoriya, K. [1995] “Tactile sensor based manipulation of an unknown object by a multifingered hand with rolling contact.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 743-750, 1995.

Maekawa, H., Tanie, K., Komoriya, K., Kaneko, M., Horiguchi, C. and Sugwara. [1992] “Development of a finger-shaped tactile sensor and its evaluation by active touch.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 2, pages 1327–1334, 1992.

Mason, M.T., and Salisbury, J.K. [1985] *Robot Hands And The Mechanics Of Manipulation*. Cambridge, MA: The MIT Press, 1985.

Massimino, M. and Sheridan, T.B. [1994] “Teleoperator performance with varying force and visual feedback.” Human Factors, Volume 36, Number 1, pages 145-157, 1994.

Montana, D.J. [1988] “The kinematics of contact and grasp.” International Journal of Robotics Research, Volume 7, Number 3, pages 17-32, 1988.

Michelman, P. and Allen, P. [1994] “Shared autonomy in a robot hand teleoperation system.” Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems, Volume 1, pages 253-259, 1994.

Niemeyer, G. [1996] *Using Wave Variables in Time Delayed Force Reecting Teleoperation*. Ph.D. Thesis, Massachusetts Institue of Technology, Department of Aeronautics and Astronautics, 1996.

Niemeyer, G. and Slotine, J.-J.E. [1997] “Using wave variables for system analysis and robot control.” Proceedings of the IEEE International Conference on Robotics and Automation, Volume 2, pages 1619-1625, 1997.

Oda, M., Inaba, N., Takano, Y., Nishida, S., Kayashi, M., and Sugano, Y. [1999] "Onboard local compensation on ETS-W space robot teleoperation." Proceedings of the IEEE/ ASME International Conference on Advanced Intelligent Mechatronics, pages 701-706, 1999.

Okamura, A.M., Smaby, N., and Cutkosky, M.R. [2000] "An overview of dexterous manipulation." Proceedings of the IEEE International Conference on Robotics and Automation, Volume 1, pages 255-262, 2000.

Pao, L. and Speeter, T.H. [1989] "Transformation of human hand positions for robotic hand control." IEEE Proceedings of the International Conference on Robotics and Automation, Volume 3, pages 1758-1763, 1989.

Provancher, W.R. and Cutkosky, M.R., [2002] "Sensing local geometry for dexterous manipulation." Proceedings of the International Symposium on Experimental Robotics, pages 507-516, 2002.

Raju, G.J., Verghese, G.C., and Sheridan, T.B. [1989] "Design issues in 2-port network models of bilateral remote manipulation." Proceedings of the IEEE International Conference on Robotics and Automation, Volume 3, pages 1316 -1321, 1989.

Rohling, R.N., Hollerbach, J.M. and Jacobsen, S.C. [1993] "Optimized fingertip mapping: A general algorithm for robotic hand teleoperation." Presence: Teleoperators and Virtual Environments, Volume 2, Number 3, pages 203-220, 1993.

Rossetter, E.J. [2003] *A Potential Field Framework for Active Vehicle Lanekeeping Assistance*. Ph.D. Thesis, Stanford University, Department of Mechanical Engineering, 2003.

Salisbury, K. [1988] "Issues in human/computer control of dexterous remote hands." IEEE Transactions on Aerospace and Electronic Systems, Volume 24, Issue 5, pages 591-596, 1988.

Schneider, S.A., Cannon, R.H., Jr. [1992] "Object impedance control for cooperative manipulation: theory and experimental results." IEEE Transactions on Robotics and Automation, Volume 8, Issue 3, pages 383-394, 1992.

Sheridan, T.B. [1992a] "Defining our terms." Presence: Teleoperators and Virtual Environments, Volume 1, Number 2, pages 272-274, 1992.

Sheridan, T.B. [1992b] *Telerobotic, Automation, and Human Supervisory Control*. Cambridge, MA: The MIT Press, 1992.

Sherman, A., Cavusoglu, M.C., and Tendick, F. [2000] "Comparison of teleoperator control architectures for palpation task." Proceedings of the ASME International

Mechanical Engineering Conference and Exposition, Dynamic Systems and Controls, Volume 69, Number 2, pages 1261-1268, 2000.

Son, J.S., Cutkosky, M.R., and Howe, R.D. [1995] "Comparison of contact sensor localization abilities during manipulation." Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Volume 2, pages 96-103, 1995.

Speeter, T.H. [1992] "Transforming human hand motion for telemanipulation." Presence: Teleoperators and Virtual Environments, Volume 1, Number 1, pages 63-79, 1992.

Speich, J.E. and Goldfarb, M. [2002] "Implementation of loop-shaping compensators to increase the transparency bandwidth of a scaled telemanipulation system." Proceedings of the IEEE International Conference on Robotics and Automation, Volume 3, pages 2886-2893, 2002.

Tan, H., Pang, X. D., and Durlach, N. [1992] "Manual resolution of length, force, and compliance." ASME-WAM Advances in Robotics, Dynamic Systems and Controls, Volume 49, pages 13-18, 1992.

Turki, L. and Coiffet, P. [1995] "On grasp synthesis and planning of multifingered robot hands for a telemanipulation task." Proceedings of the IEEE International Workshop on Robot and Human Communication, pages 141-146, 1995.

Turner, M.L. [2001] *Programming Dexterous Manipulation by Demonstration*. Ph.D. Thesis, Stanford University, Department of Mechanical Engineering, 2001.

Turner, M.L., Findley, R.P., Griffin, W.B., Cutkosky, M.R., and Gomez, D.H. [2000] "Development and testing of a telemanipulation system with arm and hand motion." Proceedings of the ASME International Mechanical Engineering Conference and Exposition, Dynamic Systems and Controls, Volume 69, Number 2, pages 1057-1063, 2000.

Wagner, C.R., Stylopoulos, N., Howe, R.D. [2002] "The role of force feedback in surgery: analysis of blunt dissection." Proceedings of the IEEE 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, pages 68-74, 2002.

Westling, G. and Johansson, R.S. [1984] "Factors influencing the force control during precision grip." Experimental Brain Research, Volume 53, pages 277-284, 1984.

Yokokohji, Y. and Yoshikawa, T. [1994] "Bilateral control of master-slave manipulators for ideal kinesthetic coupling-formulation and experiment." IEEE Transactions on Robotics and Automation, Volume 10, Issue 5, pages 605-620, 1994.

Yoshikawa, T. and Nagai, K. [1991] "Manipulating and grasping forces in manipulation by multifingered robot hands." IEEE Transactions on Robotics and Automation, Volume 7, Issue 1, pages 67-77, 1991.

Appendix A: Adept Controller Notes

The slave-side system includes a robotic hand and arm, sensors and controller software. The arm is an AdeptOne-MV five axis industrial robot from Adept Technologies (see Figure A-1). The motion of the operator's hand, used to guide the robot arm, is tracked using an unencumbering non-contact tracking system. The tracker is an ultrasonic measurement system, from Logitech, capable of tracking the six degrees-of-freedom of motion of the wrist (see Figure A-2). However, we only use four of the measured degrees-of-freedom, three degrees-of-freedom for position and one degree-of-freedom for orientation (wrist yaw) for control of the industrial robot arm. With this system and the software discussed below, we are capable of achieving a small motion bandwidth of approximately 10 Hz, which is sufficient for tracking human arm motion in our application.

Figure A-1. AdeptOne-MV (SCARA type) robot arm with five degrees-of-freedom. The robot cartesian position and orientation perpendicular to the table are controlled by the operator's wrist position and orientation.

The Adept robot is controlled by the master computer via an Ethernet TCP/IP socket connection. Commands from the master tracking system are interpreted and transformed to the robot workspace and then sent to the robot when requested. One of the major difficulties in setting up the Adept robot as a teleoperated arm was in achieving a suitable bandwidth. While the Adept is a high speed, high accuracy, direct drive SCARA robot, it was designed primarily for industrial operation, such as pick-and-place tasks. Specialized software and custom developed algorithms were necessary for real-time trajectory control over the industrial robot.

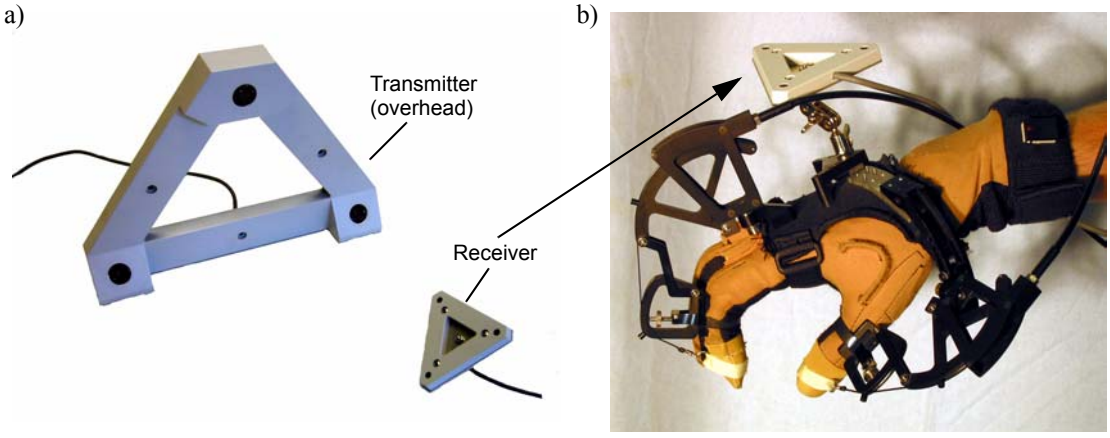

Figure A-2. a) Logitech ultrasonic tracker transmitter and receiver. b) master interface with tracker placed on operator's wrist. Transmitter was placed overhead (above the operator).

A.1 Real-time Trajectory Control of an Adept Robot

The Adept controller software, V+ (version 12.4), provides the front end for programming the robot. The V+ operating system is set to execute multiple program tasks utilizing time slicing with task priorities. Each system cycle runs at 62.5 Hz (every 16 ms) and is divided into sixteen 1ms time slices. Motion commands can be communicated to the trajectory controller once every major system cycle (every 16 ms). However, once a motion command (e.g., MOVE loc.1) is given to the trajectory controller, it cannot be interrupted except in the case of an emergency stop. A single additional motion command can be queued during the first move command to ensure a smooth continuous path trajectory. The queued setpoint is used to compute a new trajectory just before the original deceleration phase was to begin.

While supplying motion commands in this manner creates smooth motion for a string of predetermined setpoints, this is not the case for continuously updated points. The

robot has velocity and acceleration limits set by the programmer and these limits are used by the trajectory controller to create the specified trajectory profile. The robot attempts to reach each point as fast as possible given the programmed limits. To achieve reasonable motion bandwidth, the limits must be set fairly high. However, if the human hand motion is relatively slow the robot could reach the desired setpoint before the next setpoint can be used to create a continuous path, thus giving high acceleration stops and starts and resulting in jittery motion of the arm.

A.1.1 Utilizing the ALTER Command

The ALTER command provides a solution to the problems described above. To access the ALTER command, your Adept system must have the Enhanced Trajectory Control License and running V+ version 12.4 or later.[1]

The ALTER command allows the user to specify a real-time path modification that is applied to the robot path during the next trajectory generation computation. Using this command the programmer is now responsible for creating a smooth trajectory but can supply path modifications that are executed at each major servo cycle (every 16ms). In using this command, the programmer is now effectively sending a velocity command to the robot (relative position changes). However, there are a few caveats:

- This command essentially by-passes all the path planning software. The programmer is now responsible for creating a smooth trajectory profile. Additionally, one could command a very large move, e.g., 50 mm, and the robot will attempt to complete this move within one servo cycle (16 ms). This would result in a commanded velocity on the order of 3 m/s which will cause the robot to shut down.
- The ALTER command is not commonly used in industry. Therefore, the documentation is fairly limited.

1. Another approach to creating smooth trajectories is to by pass the trajectory controller and control the robot at a lower level. However, with this approach, the programmer is by passing the standard safety programs running on the trajectory controller.

- The commanded values for X-Y-Z path modification are straight forward; simply indicate the translation to be performed over the next servo cycle in millimeters. However, the rotational path modification values are more complicated and require extensive matrix calculations (see [Li et al. 2001]). Therefore, only yaw axis path modification commands (in addition to 3 axes of translation) were used.

A.1.2 Software Framework for Trajectory Control

To utilize the ALTER command and the TCP/IP Ethernet communication, it is necessary to use the multi-process capability of the V+ software. This section details the individual program tasks employed.

To control the robot using the ALTER command and desired commands from the Ethernet socket communication requires three separate tasks (or processes). At each servo cycle, a desired change in robot position is added to the robot's current position using the ALTER command. The program with the highest priority, task level 0, runs the ALTER command loop. At task level 1, the TCP/IP ethernet interface is running. The task level 1 program receives the commands from the master computer and pre-processes the data to ensure that the commands are within the workspace limits. Additionally, the velocity of the arm motion is limited for safety of the attached robotic hand. At a slightly lower priority, task level 2, the MOVES HERE command is called once and then the ALTER command can begin (this is a requirement of the ALTER command, see V+ operation manual).

One important aspect of the task level 1 program code is the modification of the data to implement both velocity and workspace limitations. Since the master computer sends the relative position changes to be executed at each servo cycle, the program can simply limit the magnitude of the position change and thus the robot velocity. However, to ensure accurate position tracking between the slave and master, the overflow position values must be stored and executed at a later time. The pseudo code in Table A-1 demonstrates how this is implemented at each servo cycle.

This method of velocity limitation allows for accurate position tracking in a relatively simple manner. In addition to velocity limiting, acceleration limiting was later added to further smooth the motions. Since acceleration limiting can cause a small amount of

The variables:
`des_step` - desired relative position change for the next servo cycle
`stored` - overflow position changes
`delta_max` - the maximum relative position change (max velocity * 16ms)
`step` - value actually sent to ALTER command

Code executed at each major servo cycle:

```
stored = stored + des_step
if stored >= 0 then
     step = min(delta_max, stored)
else
     step = max(-delta_max, stored)
end
stored = stored - step
ALTER (step)
```

Table A-1. Pseudo code for velocity limiting based on commanded changes in robot position.

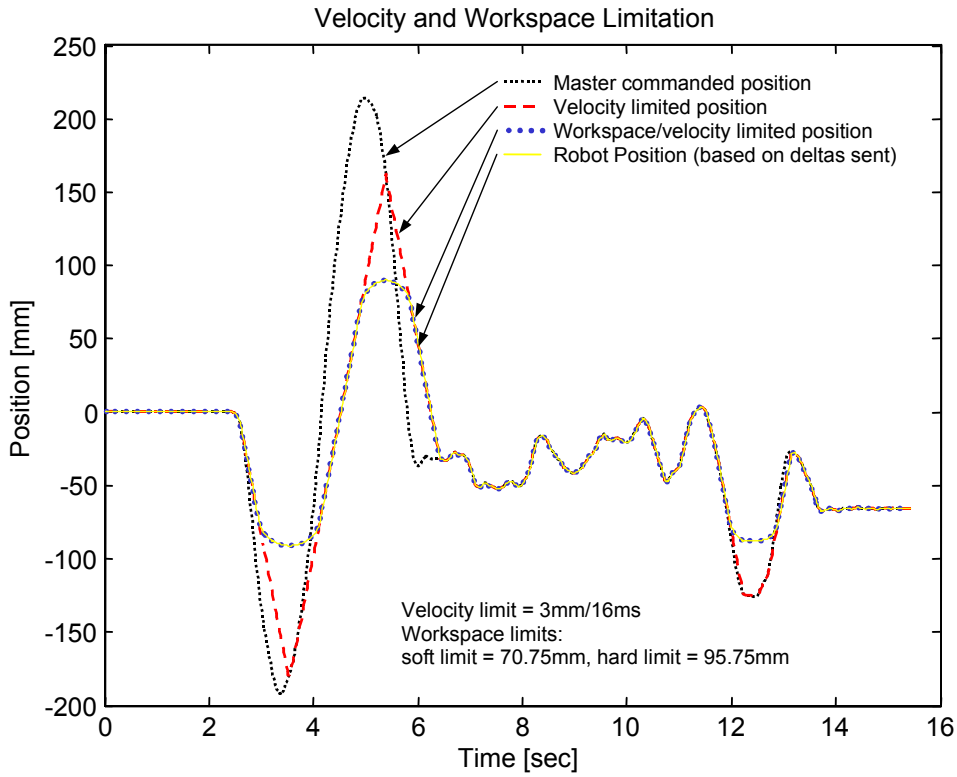

Figure A-3. Effects of velocity and workspace limitation algorithms on tracking data. The master system records data using the ultrasonic hand tracker. At each robot servo cycle, the change in position is sent to the robot. If the velocity of the translation (or rotation) exceeds at pre-set threshold, the rate of position change is limited. Additionally, if the desired position falls outside of the achievable robot workspace, a smoothing function is used to limit the position. The velocity limiting parameter was set conservatively in this trial to illustrate the limiting effect. Notice that for slow motions within the achievable workspace, the robot accurately tracks the position of the hand.

overshoot and errors in position tracking, empirical testing was used to determine the proper parameters for smooth motion given normal human hand motions.

To limit the workspace of the teleoperated robot arm one could simply set the `delta_max` term to zero when the limits are reached. However this would lead to abrupt stops and starts of the robot at the workspace boundary. To ensure smooth motion, a non-linear mapping method is used to modify the desired position near the boundaries. Near the boundaries an asymptotic function based on $\frac{1}{x}$ is used. Figure A-3 shows an example of the velocity and workspace limiting for operator hand motions in a translational axis.

A.2 Problems with Ethernet Based Communication and Velocity Based Control

Even when implementing the ALTER based trajectory control, the motion of the Adept was initially disappointingly jerky. The main source of this problem was traced to unevenly spaced sampling.

The basic communication framework is set up as follows. The Adept runs the TCP/IP communication loop every 16ms (the 62.5 Hz base servo cycle). Each time through the loop, the Adept sends the master computer a character to indicate that it is ready to receive the latest data (i.e., the latest change in relative position or `des_step`). On the master side, a process, dedicated to running the communications, waits for the ping from the Adept and then accesses shared memory to obtain the latest position (gathered in a separate process associated with the tracking system). The relative position change is computed and then sent to the Adept over the socket connection.

Even though the Adept loop is running at exactly 62.5 Hz, the communication time to (and from) the master can vary by a small amount. This implies that the communication program executed on the master system (which sends data when “pinged” by the Adept) is only running at an average of 62.5 Hz; execution timing varies slightly with each loop (approximately 16 ms, $\pm$ 3 ms). Because the master accesses new tracker data based on the Adept ping, the data may not be sampled at evenly spaced intervals.

Suppose you create a smooth sine wave position command. If the sine wave is sampled at regular intervals then each of computed the derivatives will be smooth. However, if the position signal is not sampled at regular intervals (but is then executed at regular intervals), the velocity and, more importantly, the acceleration will not be smooth. Figure A-4

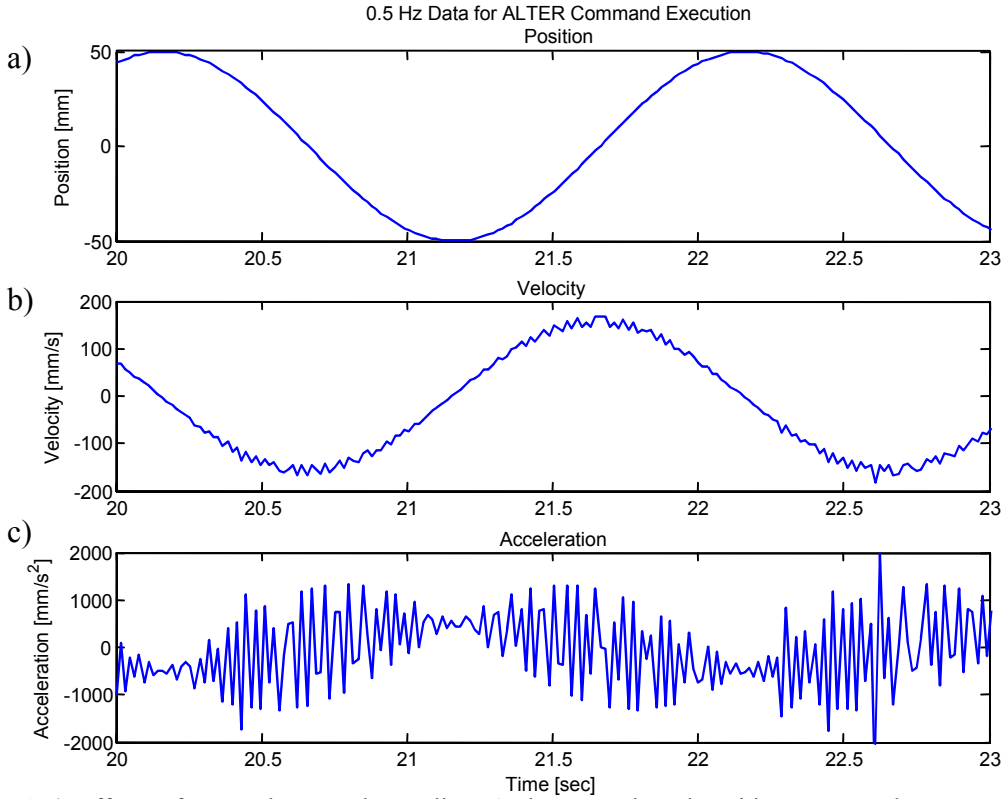

Figure A-4. Effects of unevenly spaced sampling. A sine wave based position command was computed at regular intervals (1 KHz) on the master system. Then based on the communication protocol, the Adept robot requests position changes every 16 ms. Communication delays cause the sampling to occur at slightly irregular intervals. Despite the apparently smooth position data seen by the Adept robot controller (a), the resulting computed acceleration (c) is very noisy. When the commanded sine wave is executed on the robot, the motion is jerky.

was created based on position changes as recorded by the robot. As shown, the computed acceleration is extremely noisy, even though the commanded position data appears smooth.

To solve this problem, a sample-and-hold routine was implemented on the master system. Based on the first few requests from the Adept, the master computer begins to shift the time in which the data are uploaded to shared memory for the communication process. In this way, the master computer samples the tracking data only when the Adept will not be requesting data at about the same time. Therefore, even if the data received at the Adept robot site are not sent exactly every 16 ms, the resulting position commands are based on data sampled exactly every 16 ms. The one drawback to this method is that the data from the tracker are delayed by approximately one-half the sampling period, 8 ms. The implementation of this synchronization results is smooth motion on the Adept robot with a small cost in latency. Figure A-5 again represents a commanded sine wave sent to the Adept

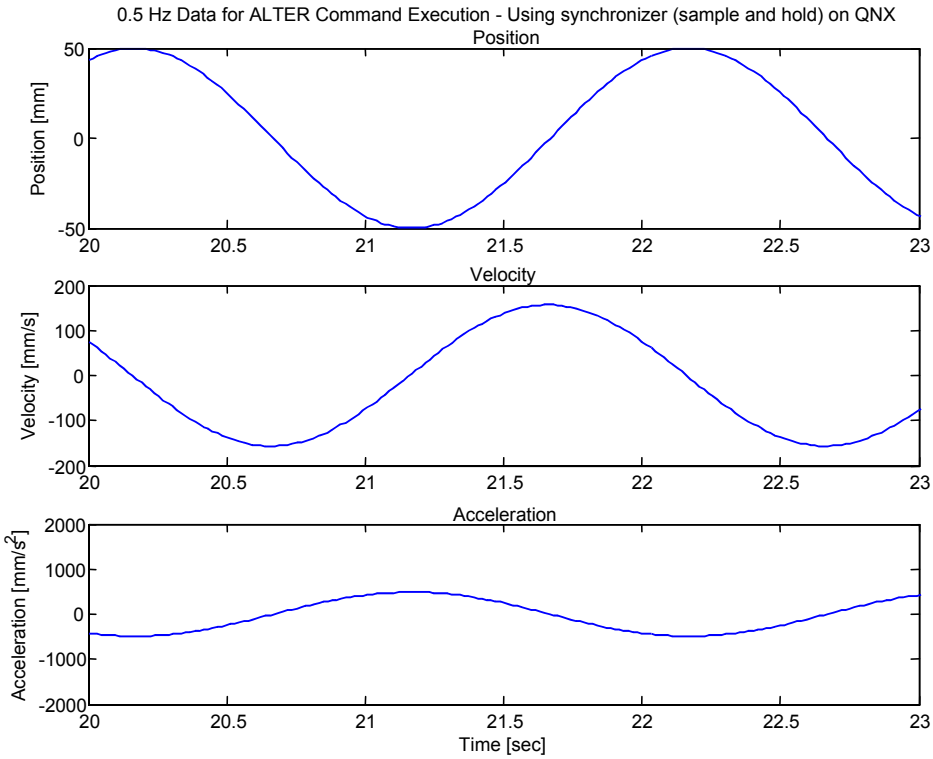

Figure A-5. Effects of sample-and-hold synchronization technique. Notice that the acceleration is now noise free.

robot. However, we can see that the computed acceleration based on the communicated data results in a smooth acceleration profile.

Appendix B: Human-to-Robot Mapping Implementation Details

The following sections detail the implementation of the virtual object mapping method introduced in Chapter 4. First, the parameters of the virtual object are defined in the hand frame. Next, the methods for transforming the virtual object parameters to the robot frame are discussed. The procedures for mapping the parameters to the robot frame are presented. Finally, the methods for modifying the parameters to better match the robot workspace are discussed.

B.1 Computing the Virtual Object Parameters

As discussed in Chapter 4, we must first carefully chose what virtual object information should be mapped from the higher dimensional hand space to the lower dimensional space of the planar non-anthropomorphic robot hand.

In the general case, the object is defined by seven parameters (size plus size parameters to define position and orientation). However, by projecting the object size, midpoint position and orientation onto the hand's assumed plane of manipulation (as in Figure B-1), we reduce the number of parameters to four, which matches the number of degree-of-free-

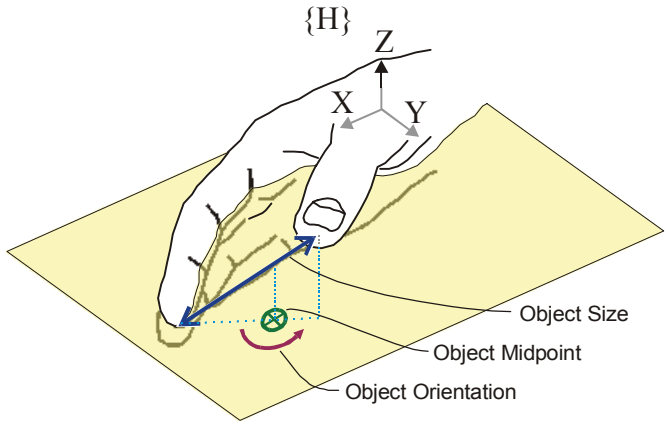

Figure B-1. The virtual object parameters are defined based on the thumb and index fingertip positions. The size of the object is defined by the distance between the fingertips. The object position is defined by the midpoint between the fingertips and projected on the X-Y plane. The object orientation is also defined by the angle of the projected line between the fingertips.

dom in the robot hand. The development of the virtual object parameters from the hand frame data follows.

The size of the virtual object is based on the distance between the fingertips. Thus, if an operator is manipulating a golf ball, the modeled separation should remain the same even as the thumb moves out of the plane of the index finger. We first compute the vector from the thumb tip position to the index tip position by:

$$ {}^{H}\Delta^{k} = {}^{H}\boldsymbol{p}^{k}_{index} - {}^{H}\boldsymbol{p}^{k}_{thumb} \tag{B.1} $$

The vectors ${}^{H}\boldsymbol{p}^{k}_{index}$ and ${}^{H}\boldsymbol{p}^{k}_{thumb}$ are 3×1 vectors representing the three-dimensional fingertip positions in hand frame (denoted by the left superscript *H*) and are computed using forward kinematics based on the calibrated hand model and the glove sensor readings. The right superscript *k* indicates that the quantities are calculated each time the glove sensor values are read at 200 Hz. The magnitude of the vector difference yields the fingertip-to-fingertip distance:

$$ {}^{H}d^{\,k} = \left\| {}^{H}\Delta^{k} \right\| \tag{B.2} $$

where ${}^{H}d^{\,k}$ represents the size of the virtual object.

The position of the virtual object in the hand frame is first calculated by finding the midpoint between the thumb and index fingertips. However, based on observations during method development, natural human grasping motion tends to be asymmetric with respect to the thumb and index finger motions. Small object rolling motions are dominated by index finger motion. Also, as an operator opens his/her grasp from a comfortable pinch-point, the index finger tends to move further than the thumb from the initial starting point. If these motions are not accounted for by the mapping, the midpoint of the virtual object will shift significantly. Additionally, as grasping motions are scaled to fit the robot's workspace, asymmetric motions will tend to limit the achievable positions of the thumb because of the index finger's larger motions. In an attempt to cancel out these asymmetric motions when mapping to the symmetric robot, the object midpoint is redefined as follows:

$$ {}^{H}\boldsymbol{p}'^{k}_{midpoint} = {}^{H}\boldsymbol{p}^{k}_{thumb} + {}^{H}\Delta^{k} \cdot \gamma \tag{B.3} $$

where γ represents an empirical midpoint shift ratio and the prime superscript, ', represents the shifted midpoint. When $\gamma = 0.5$, the object midpoint is the true midpoint between the index and thumb fingertips. If the shift ratio is reduced by a small amount (moving the midpoint towards the thumb), the drift of the virtual object midpoint from the pinch point is reduced when the operator opens and closes his/her grasp or makes a rolling motion.

To accommodate the planar limitations of the robotic hand, the midpoint is projected onto the plane of primary index finger motion. Based on the kinematic model defined for the hand, the vector projection is simply the X and Y components of the midpoint vector, yielding the new midpoint vector:

$$ {}^{H}\boldsymbol{p}'^{k}_{x\text{-}y,\,midpoint} = \left[{}^{H}p'^{k}_{x,\,midpoint} \quad {}^{H}p'^{k}_{y,\,midpoint} \right]^{T} \tag{B.4} $$

The projected motion onto the X-Y plane captures a relevant subset of the virtual object translations. (In fact, it is quite difficult to move a grasped object parallel to the palm-motions along the Z-axis in the hand frame, see Figure B-1).

The orientation of the virtual object is defined by the angle of the planar projection of the line between the thumb and index fingertips. Similar to the midpoint, the plane on which the line is projected is the X-Y plane in the hand space. Rotations about vectors spanning the X-Y plane are not represented by this projection. However, the primary orientation changes during typical object manipulation in the hand frame are captured using the tip-to-tip line projection. We define the orientation using the fingertip-to-fingertip difference vector, ${}^{H}\Delta^{k}$, defined by Equation B.1. Applying the four-quadrant inverse tangent function to the X and Y values of the difference vector yields the projected object orientation:

$$ {}^{H}\phi^{k} = \operatorname{atan2}({}^{H}\Delta^{k}_{y}, {}^{H}\Delta^{k}_{x}) \tag{B.5} $$

Figure B-1 illustrates the virtual object parameters as defined by the thumb and index fingertip positions. The Cartesian thumb and index fingertip positions are reduced to the following planar virtual object parameters (representing four values):

- object size: ${}^{H}d^{k}$
- object midpoint (shifted by γ): ${}^{H}\boldsymbol{p}'^{k}_{x\text{-}y,\,midpoint} = \left[{}^{H}p'^{k}_{x,\,midpoint} \quad {}^{H}p'^{k}_{y,\,midpoint} \right]^{T}$

- object orientation: ${}^{H}\phi^{k}$

Each of these parameters is computed at each sample time, *k*, based on the sampled glove readings applied to the kinematic model.

B.2 Computing Robot Positions

The virtual object mapping method discussed thus far transforms human hand motions into a planar virtual object. For clarity, it is useful to first discuss the method in which the robot fingertip positions are created from the virtual object parameters. With an understanding how the robot motions are created from the virtual object data, it is easier to follow the development of the transformation and scaling methods used to match the workspaces (discussed in the next section). For the following discussion assume that the virtual object parameters have been mapped to the robot hand frame such that hand motions have an obvious correspondence with the robot motions, e.g., if the operator opens his/her grasp, the robot creates a horizontal opposing grasp.

For free-space motions of the robot fingers, the mapped virtual object parameters are used to compute robot fingertip positions. The following planar transformation equations is used:

$$ {}^{R}\boldsymbol{p}_{index}^{k} = \begin{bmatrix} \cos({}^{R}\phi^{k}) & -\sin({}^{R}\phi^{k}) \\ \sin({}^{R}\phi^{k}) & \cos({}^{R}\phi^{k}) \end{bmatrix} \cdot \begin{bmatrix} -\frac{{}^{R}d^{k}}{2} \\ 0 \end{bmatrix} + \begin{bmatrix} {}^{R}p_{x,\,midpoint}^{k} \\ {}^{R}p_{y,\,midpoint}^{k} \end{bmatrix} \tag{B.6} $$

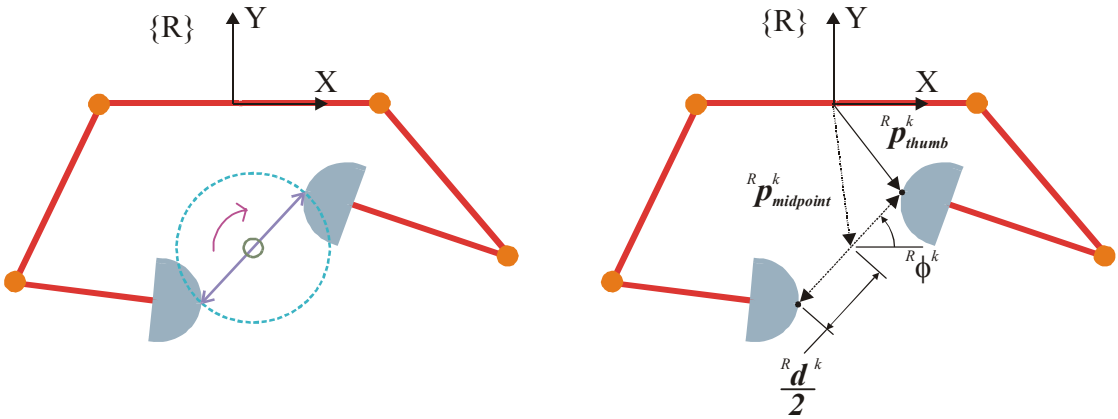

Figure B-2. Computing robot fingertip positions for virtual object parameters for free-space motion.

$$ {}^{R}\mathbf{p}_{thumb}^{k} = \begin{bmatrix} \cos({}^{R}\phi^{k}) & -\sin({}^{R}\phi^{k}) \\ \sin({}^{R}\phi^{k}) & \cos({}^{R}\phi^{k}) \end{bmatrix} \cdot \begin{bmatrix} \frac{{}^{R}d^{k}}{2} \\ 0 \end{bmatrix} + \begin{bmatrix} {}^{R}p_{x,\,midpoint}^{\mathrm{k}} \\ {}^{R}p_{y,\,midpoint}^{k} \end{bmatrix} \tag{B.7} $$

Figure B-2 graphically illustrates the method used to obtain the robot fingertip positions based on the mapped virtual object. It is important to note that the calculation of the robot fingertips is simplified considerably by assuming the robot maintains a point contact with the virtual object (as opposed to assuming rolling contact between the robot fingers and the virtual object). In other words, the virtual object can also be thought of as a virtual "toothpick" of varying length, position, and orientation in the plane. If the mapped virtual object parameters cause a computed fingertip location that falls outside of the workspace, the nearest achievable position (in Cartesian space) is used.

When an actual object is detected between the fingers, the virtual object parameters are used to compute set-points for a cooperative object impedance controller (see Chapter 5 for controller details). At the time the object is detected, the controller creates an object model that is continuously updated by the tactile sensors. The changes in the virtual object position and orientation (computed each servo cycle) are added to the initial model to created desired positions and orientations for the actual object. The virtual object size parameter is used to create a desired internal force applied to the object. The amount of force applied is proportional to the difference between the actual object size and the commanded virtual object size.

It is also important to note that there is a disparity between the rate the glove is sampled (and thus the rate the virtual object parameters are computed) and the rate at which the robot controller is running. The glove is samples at 200 Hz while the robot servo rate is 1 kHz. To account for this difference, the virtual object parameters are smoothed with a low pass filter (7 Hz cutoff frequency).

B.3 Transformation to the Robot Hand Frame

Before the virtual object parameters are used to create robot position or object commands, the parameters must be properly mapped to the robot frame from the hand frame. Our desire is to transform the object information in such a way that the operator can intuitively control

the robotic fingers. Figure B-3 shows the desired correspondence between human hand poses and robot finger configuration.

The transformation of the virtual object parameters is defined such that the comfortable pinch-point of the operator maps roughly to the geometric center of the robot's workspace (Figure B-3, pose D) and natural human grasping motions are mapped to horizontal grasping motions in the robot's workspace (Figure B-3, poses A, B, and C). Ideally, human finger motion towards and away from the palm will also map to vertical motion of the robot fingers (Figure B-3, poses D, E, and F). Because every operator's hand is slightly different, the virtual object transformation variables are generated for each individual operator and

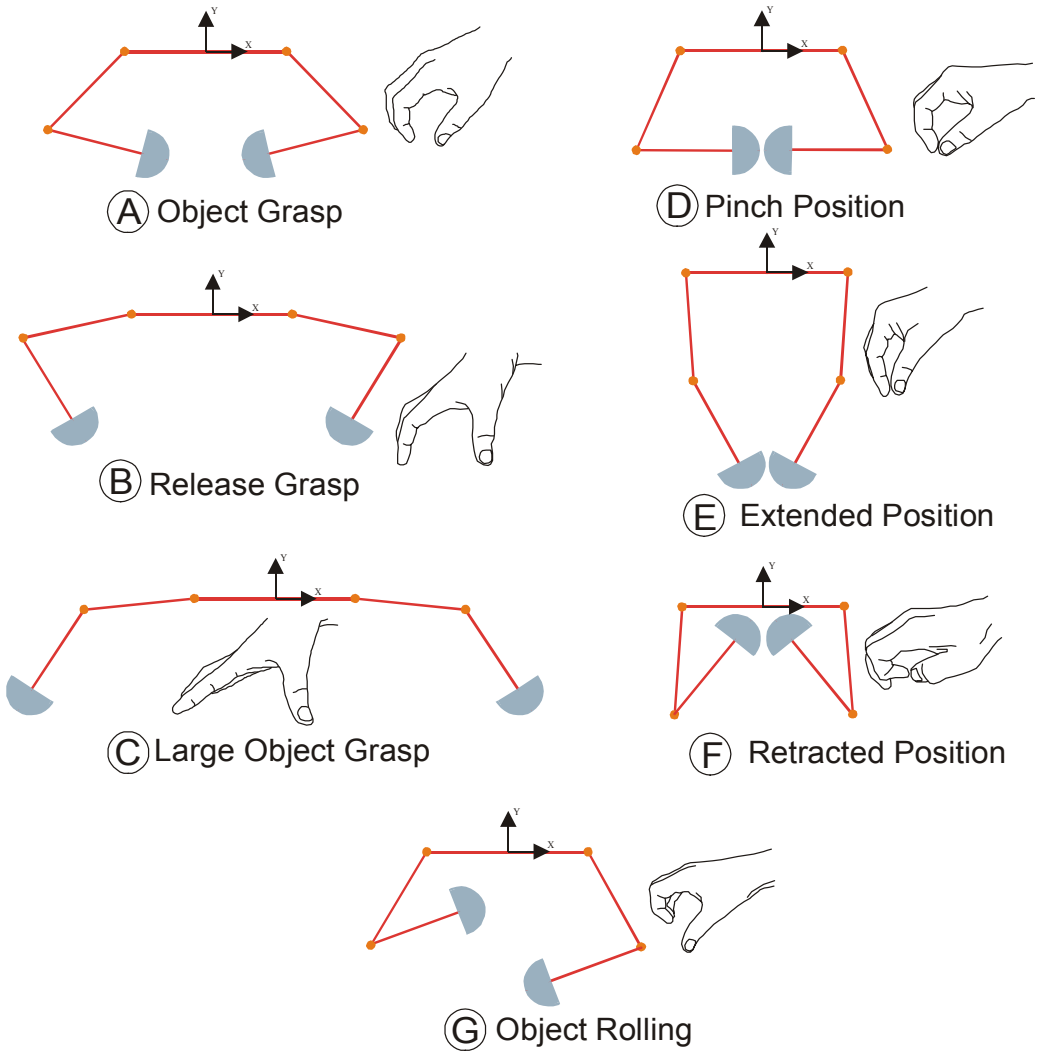

Figure B-3. Desired correspondence between human hand poses and robotic hand configuration. The motion of enlarging one grasp is mapped to an increased separation between the robots fingers along the horizontal (poses A, B, and C). The motion of moving ones fingers towards and away from the palm is mapped to vertical motion of the robot's fingertips (poses D, E, and F).

based on recorded data from a few specific hand motions and poses. Automating the calculation of the transformation variables increases the consistency of the mapping for each subject and prevents tedious trial-and-error variable adjustment.

B.3.1 Mapping the Virtual Object Orientation

The first step in matching the robot motions to the hand motions is to transform the virtual object orientation ${}^{H}\phi^{k}$. To match the horizontal grasping motions in the robot frame to human grasping motions (poses A, B and C in Figure B-3), a simple angular offset is applied to the measured virtual object orientation, ${}^{H}\phi^{k}$. However, determination of the offset based on captured fingertip data is a somewhat complicated task.

To obtain the orientation offset angle, the operator is asked to open and close his/her grasp as in poses A, B and C in Figure B-3 (a natural grasping motion). The thumb and index fingertip positions are used to create a virtual object in the hand frame, as described in Section B.1. The virtual object parameters are then used to compute planar thumb and index fingertip positions in the hand frame, in a manner similar to the method used to calculate the robot's fingertip positions from mapped virtual object data (described in the previous section).

The newly computed tip positions are used to define the necessary orientation offset angle to align the open-and-close grasping motions with horizontal robot grasping motion. Using thumb and index positions created using the same method the robot employs ensures that the offset variable used to compute the mapped virtual object orientation (and thus robot tip positions) produces a consistent mapping. In other words, the computed planar thumb and index fingertip positions can be viewed as un-transformed robot fingertip positions in the hand frame. We are using the tip positions to determine the necessary offset angle to rotate the tip positions such that the open-and-close grasping motions map to the horizontal robot grasping motion. A best-fit line is fitted to the computed planar thumb and index fingertip positions. The orientation of the line defines the angle of the grasping motion in the hand space. With the motion characterized in the hand frame it is a simple operation to compute the offset.

The open-and-close grasp data are recorded for five seconds at 100 Hz. For each glove reading, the thumb and index data are used to compute the virtual object parameters

and then the adjusted (planar) index and thumb fingertip positions are computed as follows (same basic approach outlined for Equations B.6 and B.7):

$$ {}^{H}\boldsymbol{p}^{\,i}_{x\text{-}y,\,index,adj} = \begin{bmatrix} \cos({}^{H}\phi^{i}) & -\sin({}^{H}\phi^{i}) \\ \sin({}^{H}\phi^{i}) & \cos({}^{H}\phi^{i}) \end{bmatrix} \cdot \begin{bmatrix} \frac{{}^{H}d^{\,i}}{2} \\ 0 \end{bmatrix} + \begin{bmatrix} {}^{H}p'^{\,i}_{x,\,midpoint} \\ {}^{H}p'^{\,i}_{y,\,midpoint} \end{bmatrix} \tag{B.8} $$

$$ {}^{H}\boldsymbol{p}^{\,i}_{x\text{-}y,\,thumb,adj} = \begin{bmatrix} \cos({}^{H}\phi^{i}) & -\sin({}^{H}\phi^{i}) \\ \sin({}^{H}\phi^{i}) & \cos({}^{H}\phi^{i}) \end{bmatrix} \cdot \begin{bmatrix} -\frac{{}^{H}d^{\,i}}{2} \\ 0 \end{bmatrix} + \begin{bmatrix} {}^{H}p'^{\,i}_{x,\,midpoint} \\ {}^{H}p'^{\,i}_{y,\,midpoint} \end{bmatrix} \tag{B.9} $$

where the right superscript, i, indicates the i^{th} record from the open-and-close motion data. It is important to note that the adjusted tip positions are only intermediate values computed during the mapping set-up procedure and used to calculate the transformation quantities for each individual operator. Figure B-4a shows a typical planar projection of actual fingertip

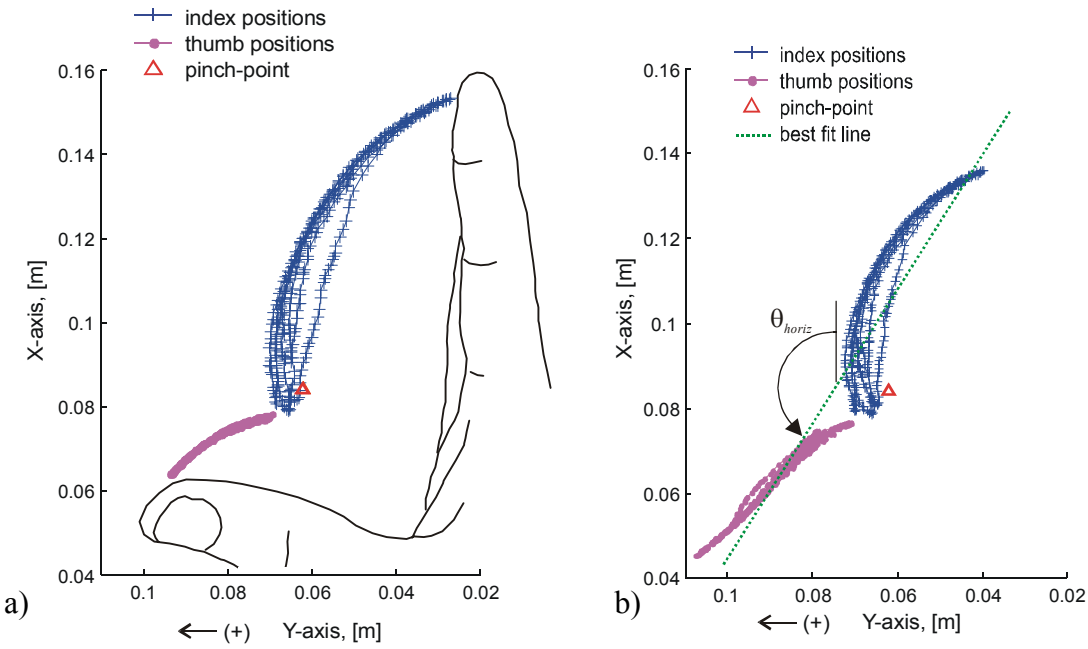

Figure B-4. a) Planar projection of typical thumb and index fingertip data for an open and close grasping motion. Note: the X-Y frame is rotated for clarity. b) The same thumb and index fingertip data modified as in Equations B.8 and B.9, where the fingertip data are used to create a virtual object and then the virtual object data are used to recreate planar fingertip positions. A best fit line is placed through the data and the angle θ_{horiz} is used to determine the offset for the virtual object orientation. The graph also illustrates the effect the shifted midpoint has on making the data more symmetric with respect to the pinch-point location.

positions used to compute the virtual object parameters. Figure B-4b shows the planar fingertip positions created using the virtual object parameters.

A best-fit line is computed using the adjusted planar thumb and index fingertip positions. The best-fit line algorithm is based on a total least squares method. The orientation of this line in the hand frame is defined as θ_{horiz} and is measured counter-clockwise from the X-axis (right-hand rule), see Figure B-4b.

When mapping the virtual object orientation to the robot frame, we desire a level opposing grasp of the robot fingers when the virtual object orientation in the hand frame is coincident with the orientation of the open-and-close grasp line (i.e., when ${}^{H}\phi^{k} = \theta_{horiz}$). In other words, when the operator opens his/her grasp, the robot finger's should move apart along a line parallel to the X-axis in the robot frame. (Remember that the index finger corresponds to the robot's left finger and the thumb corresponds to the robot's right finger.)

In the robot frame, the virtual object orientation is measured counter-clockwise from the X-axis (about the Z-axis, using the right hand rule). Thus, a level opposing grasp in the robot frame is defined as a zero degree orientation of the mapped virtual object. The mapped virtual object orientation is computed as follows:

$$ {}^{R}\phi^{k} = {}^{H}\phi^{k} - \theta_{horiz} \tag{B.10}$$

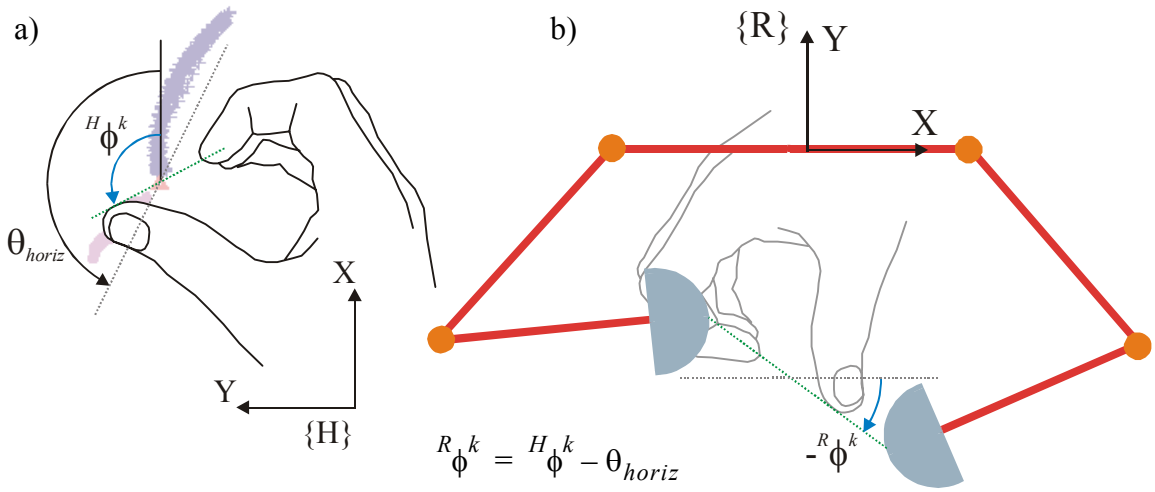

Figure B-5. The virtual object orientation is mapped from the hand frame to the robot frame such that natural open-and-close grasp motions (at and angle of θ_{horiz}) map to a horizontal grasp motion in the robot frame. Thus, A clockwise virtual object rotation in the hand frame (about the Z-axis) maps to a clockwise virtual object rotation (about the Z-axis) in the robot frame.

Figure B-5a shows a typical hand rolling motion causing clockwise rotation of the virtual object in the hand frame (about the Z-axis). When the object motion is transformed to the robot frame using Equation B.10, the mapped virtual object undergoes an equal clockwise rotation (about the Z-axis), resulting in a negative object orientation angle (Figure B-5b).

B.3.2 Rotating the Virtual Object Midpoint

The next step in transforming the virtual object data to the robot frame is to map the virtual object midpoint vector, ${}^{H}\boldsymbol{p}'^{k}_{x\text{-}y,\,midpoint}$. A standard planar frame transformation is used:

$$ {}^{R}\boldsymbol{p} = {}^{H}_{R}\underline{R} \cdot {}^{H}\boldsymbol{p} + {}^{R}\boldsymbol{p}_{o} \tag{B.11} $$

where ${}^{H}_{R}\underline{R}$ is a 2×2 rotation matrix from the hand frame to the robot frame and ${}^{R}\boldsymbol{p}_{o}$ is a offset vector specified in the robot frame.

To simplify the calculation of the transformation offset, the midpoint data is computed relative to the natural pinch-point location. The pinch-point location is obtained by asking the operator to place his/her thumb and index fingertips together in a comfortable pose (as in pose D, Figure B-3). One hundred samples of the fingertip positions are recorded over one second and used to create an average pinch-point position. The average pinch point position is defined as follows:

$$ {}^{H}\boldsymbol{p}_{pinch\text{-}point} = \frac{1}{N} \cdot \sum_{i}^{N} \frac{{}^{H}\boldsymbol{p}^{i}_{index} + {}^{H}\boldsymbol{p}^{i}_{thumb}}{2} \tag{B.12} $$

for N = 100. Only the X-Y projection of the pinch-point data, ${}^{H}\boldsymbol{p}_{x\text{-}y,\,pinch\text{-}point}$, is used (based on the same reasons the midpoint is projected onto the X-Y plan). The relative midpoint vector (with respect to the pinch-point) is computed as follows:

$$ {}^{H^{rel}}\boldsymbol{p}'^{k}_{x\text{-}y,\,midpoint} = {}^{H}\boldsymbol{p}'^{k}_{x\text{-}y,\,midpoint} - {}^{H}\boldsymbol{p}_{x\text{-}y,\,pinch\text{-}point} \tag{B.13} $$

where H^{rel} represents a translated hand frame with an origin at the average pinch-point location. Forming a relative vector allows us to place the pinch-point in the robot frame by simply adding the desired pinch-point location within the robot's workspace. Also, as we will show, performing the rotation about the H^{rel} origin allows us to easily apply other

affine transformation techniques to skew and scale the data in order to improve the mapping.

With the midpoint data expressed relative to the pinch-point location, we next determine the angle necessary to rotate the data from the hand frame to the robot frame. Using the previously calculated "horizontal angle," θ_{horiz}, we define the rotation angle such that object motion along the open-and-close grasp line maps to horizontal virtual object motions in the robot frame. As a result, any X-Y motions of the virtual object perpendicular to the grasp motion will map to vertical motion in the robot frame (ideally ensuring the motion correspondence shown in poses D, E, and F of Figure B-3).

At first glance, using the "horizontal" angle to define the necessary rotation of the midpoint data may appear odd because the most useful midpoint motion is primarily along the motions into and away from the palm. When mapped to the robot hand frame, this motion will allow operators to move the object up and down within the grasp. Thus, as we will discuss fully in section B.4, Workspace Matching, it is necessary to apply a high gain to the motions along the Y-axis to improve the workspace matching. However, it is not necessary to have a large gain on the X-axis motion. Because the midpoint shift due to the operator's natural open-and-close grasp motion can not be completely removed using the linear adjustment in Equation B.3, the midpoint will slightly drift. If the midpoint drift motion is not mapped to the horizontal in the robot's workspace (i.e., $\theta_{rot} \neq \theta_{horiz}$, instead "vertical" midpoint motions are mapped to the vertical axis), the mapped virtual object tends to move dramatically up and down as the operator opens and closes his/her grasp (more details to follow). Thus, we define the rotation of the midpoint based on θ_{horiz}.

If we define a vector, ${}^{H}\boldsymbol{h}$, in the X-Y plane of the hand frame to point in the direction defined by θ_{horiz} (see Figure B-5a), the mapped vector in the robot frame should align with the X-axis to ensure object motions along and parallel to ${}^{H}\boldsymbol{h}$ map to the horizontal (see Figure B-5b). In the hand frame, this vector will generally point towards the thumb. Thus, when mapped to the robot frame, the vector should point towards the right finger (in the positive X-axis direction). Therefore, we define the frame transformation rotation angle as:

$$\theta_{rot} = \theta_{horiz} \tag{B.14}$$

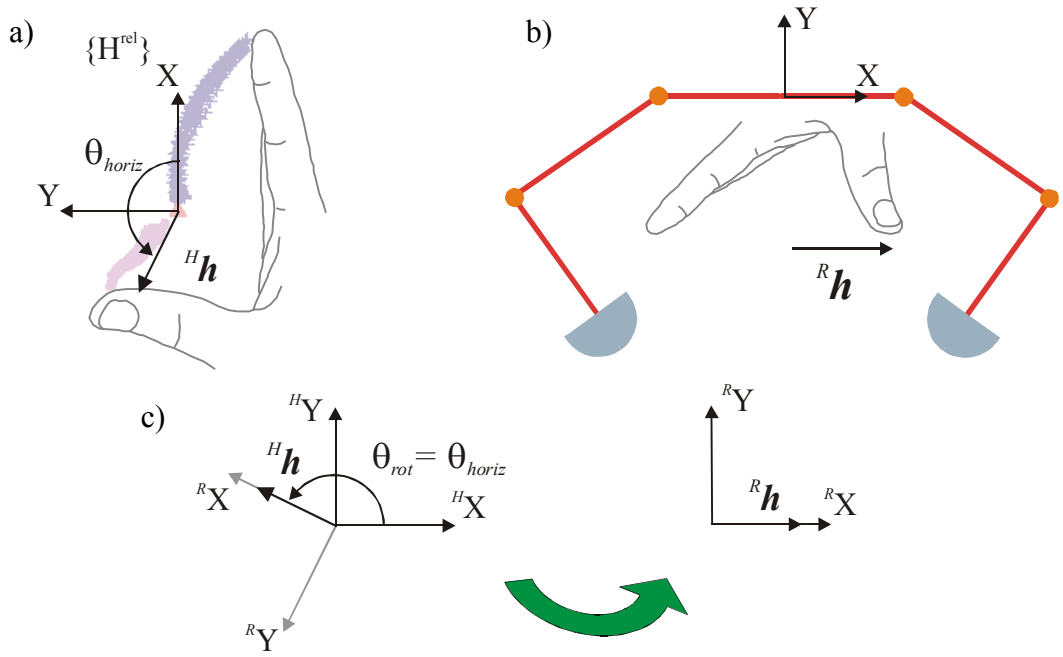

Figure B-5. Determination of the standard planar transformation rotation angle. Motions of the virtual object midpoint are mapped using the calculated angle, θ_{horiz}, mapping midpoint motions along ${}^{H}\boldsymbol{h}$ to horizontal motions in the robot frame. Motions perpendicular to ${}^{H}\boldsymbol{h}$ map to vertical motions in the robot frame.

To map the relative midpoint vector to the robot frame, the following transformation was used:

$$ {}^{R}\boldsymbol{p}^{\,k}_{midpoint} = \begin{bmatrix} \cos(\theta_{rot}) & \sin(\theta_{rot}) \\ -\sin(\theta_{rot}) & \cos(\theta_{rot}) \end{bmatrix} \cdot {}^{H^{rel}}\boldsymbol{p}'^{\,k}_{x\text{-}y,\,midpoint} \qquad \text{(B.15)} $$

B.3.3 Translating the Virtual Object Midpoint

With the virtual object midpoint vector properly rotated, we define the offset in the robot frame such that the natural pinch-point maps to the approximate center of the robot's workspace, as in Figure B-3 pose D. The offset determines the location where the robot's fingertips will come together when the operator brings his/her fingertips together. Because the differences between thumb and index motion have been accounted for using the virtual object parameters, the desired pinch-point location is placed along the Y-axis (along the line of symmetry). The height of the pinch-point in the workspace was specifically chosen to give operators a good manipulation range of motion. If the point is placed too low or too high in the workspace, the robot's manipulation range is limited. Thus, the desired pinch-

point location is placed in the approximate geometric center of the robot's workspace. We define our transformation offset in the robot frame as follows:

$$ {}^{R}\boldsymbol{p}_{pinch\text{-}point,des} = {}^{R}\boldsymbol{p}_{o} = \begin{bmatrix} 0 \\ -0.1 \end{bmatrix} \tag{B.16} $$

B.3.4 Skewing the Virtual Object Midpoint

Equations B.15 (and B.16) rotate and translate the human hand data to the robot hand plane attempting to provide and intuitive correlation between human finger motions and robot finger motions. However, in applying this transformation as defined, we noticed that vertical motions created by moving the thumb and index into and away from the palm (poses D, E, and F) were not always perpendicular to the grasping motions (poses A, B, and C in Figure B-3). Therefore, in addition to rotating the data into the robot workspace, a skew transformation was added to orthogonalize the these two motions.[1] To determine the skew transformation parameter, additional fingertip data were recorded for each operator. The operator was asked to move his/her fingers into and away from the palm while keeping their fingertips together (as in poses D, E, and F of Figure B-3). Fingertip data were recorded for five seconds at 100 Hz.

The recorded midpoint data were projected into the X-Y plane. A best fit line was computed using the midpoint as calculated by Equation B.3, with a shift ratio of 0.5 (the true midpoint). The orientation of this line in the hand space defines the "vertical angle," θ_{vert}, and measured from the X-axis using the right hand rule. Figure B-6 shows the X-Y planar projection of typical midpoint data recorded and the best fit line describing the "vertical" motion.

1. An obvious question to ask is: Why not simply rotate the midpoint data in the hand frame such that motions towards-and-away from the palm match vertical motions in the robot frame (as in poses D, E, and F of Figure B-3) *and then* offset the virtual object orientation angle to ensure the open-and-close grasp motions map to horizontal fingertip motions in the robot frame? Again, the inability to completely cancel the midpoint shifting would result in large motions of the mapped virtual object when the vertical gain is increase to match workspaces. Thus, the final mapping requires a skew transformation to orthogonalize the "horizontal" and the "vertical" motions. However, one could rotate to align the vertical motions, then apply a skew transform to prevent midpoint shift from adding to the objects vertical motion, resulting in a nearly identical mapping.

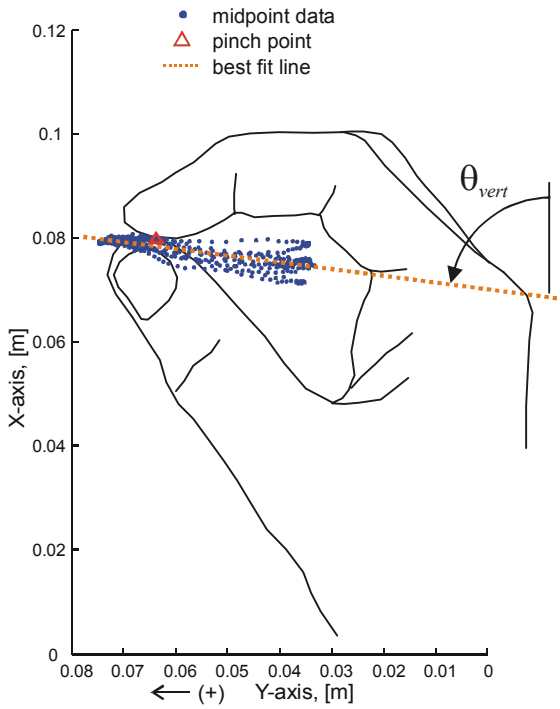

Figure B-6. Planar projection of typical midpoint data as the operator moves thumb and index fingertips together into and away from the palm. A best fit line is placed through the data and the "vertical" angle is used for the skew transformation to orthogonalize the "horizontal" and "vertical" hand motions.

A skew transformation matrix is used to orthogonalize the "horizontal" and "vertical" motions. An important property of the skew transform is that all lines parallel to a defined fixed line (the X-axis) are mapped to themselves[2]. However, the points along these lines are shifted by an amount proportional to the distance from the chosen fixed line[3]. Since the midpoint data is expressed relative to the pinch-point we chose the X-axis as the fixed line. Using these properties, the skew transformation allows us to shift all the data so that the "vertical" midpoint motion is now perpendicular to any horizontal motions as mapped in the robot hand space. Choosing the X-axis as the fixed line, the general form of the skew transformation is given by:

2. See http://www.quantdec.com/GIS/affine.htm for additional information.

3. Because of the skew transform properties, it is important that the midpoint data we desired to orthogonalize is relative to the intersection of the "horizontal" and "vertical" lines. The intersection of these lines in the hand space is approximated by the measured pinch-point location. Thus, the midpoint vector is expressed relative to this location, computed by Equation B.13, before applying the skew transform. If the skew transform was applied after the midpoint vectors were rotated and offset, an additional offset would be required to the desired pinch-point correspondence as in pose D, Figure B-3.

$$\begin{bmatrix} 1 & n \\ 0 & 1 \end{bmatrix} \tag{B.17}$$

where n is some number that defines the amount a point is shifted as the distance of the point from the X-axis increases.

As shown in Figure B-7, we define a vector ${}^{H}\boldsymbol{v}$ in the X-Y plane of the hand frame to point in the direction defined by θ_{vert} (see Figure B-7a). When this vector is rotated to the robot frame, we can see that it is not perpendicular to the vector ${}^{R}\boldsymbol{h}$ pointed along the horizontal. To bring the vector $\boldsymbol{v}$ in line with the negative Y-axis, n is defined as follows:

$$n = \frac{-1}{\tan(\theta_{skew})} \tag{B.18}$$

where,

$$\theta_{skew} = \theta_{vert} - \theta_{vert} \tag{B.19}$$

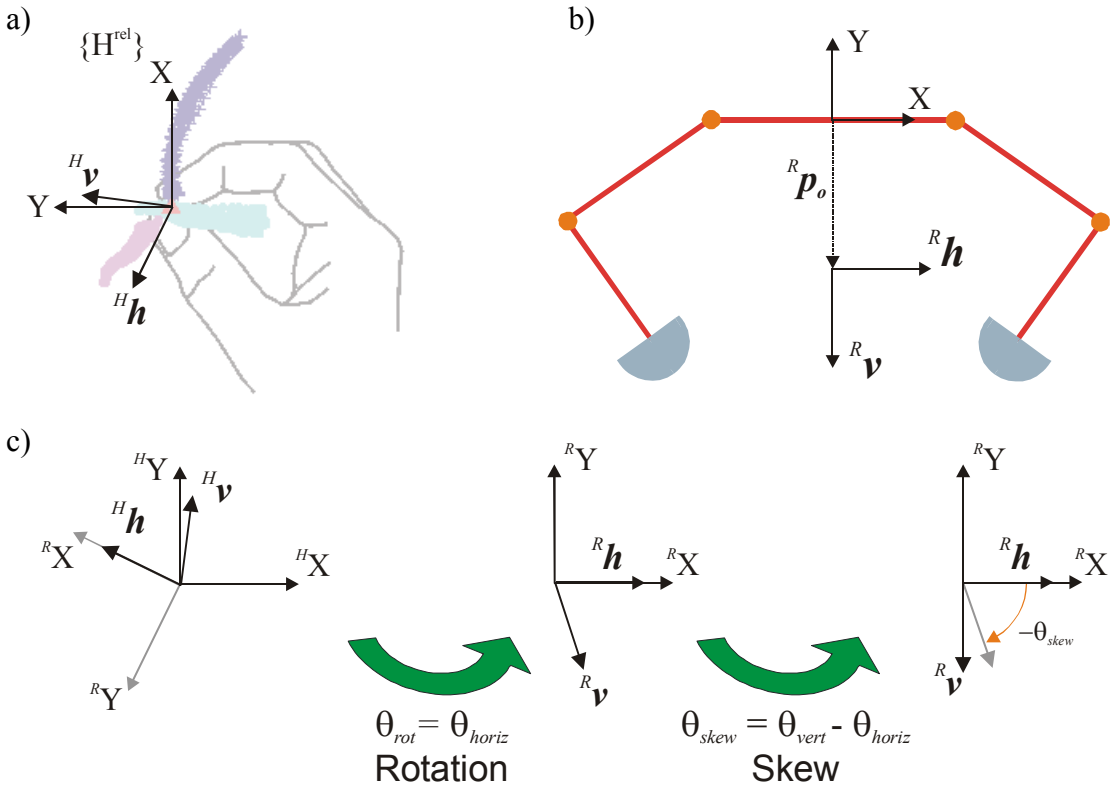

Figure B-7. The general procedure for mapping the fingertip positions from the hand frame to the robot frame. The virtual object position data is rotated to the robot hand frame and then a skew transformation is applied to orthogonalize midpoint motions during grasping with motions into and away from the palm.

With the rotation angle, skew matrix, and offset vector defined, the virtual object midpoint vector is transformed to the robot frame using the following equation:

$$ {}^{R}\boldsymbol{p}_{midpoint}^{\,k} = \begin{bmatrix} 1 & n \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta_{rot}) & \sin(\theta_{rot}) \\ -\sin(\theta_{rot}) & \cos(\theta_{rot}) \end{bmatrix} \cdot {}^{H^{rel}}\boldsymbol{p}'^{\,k}_{x\text{-}y,\,midpoint} + {}^{R}\boldsymbol{p}_{pinch\text{-}point,des} \quad \text{(B.20)} $$

which operates on the k^{th} sample virtual object midpoint vector relative to the pinch-point in the hand frame, ${}^{H^{rel}}\boldsymbol{p}'^{\,k}_{x\text{-}y,\,midpoint}$.

B.4 Workspace Matching

The virtual object size, orientation, and midpoint position are now represented in the robot hand frame. The transformation method discussed thus far addresses the kinematic differences between the human hand and the robot hand. In particular, the natural hand motions are modified to account for the symmetric and non-anthropomorphic design of the robotic hand. However, the robotic hand is physically larger than the human hand allowing for a much greater workspace. To better utilize the workspace of the robot hand, the virtual object parameters are scaled. To achieve the desired correspondence in poses as shown in Figure B-3, the virtual object size and virtual object Y-axis midpoint data are scaled.

B.4.1 Virtual Object Size Scaling

The size of the virtual object is scaled such that the maximum span of the human hand pose matches the maximum grasp size the robot can achieve (pose C, Figure B-3). Initially a simple linear scaling was applied to the virtual object size using the maximum span achieved during the open-and-close grasp motions (recorded for computing θ_{horiz}). The object size was scaled as follows:

$$ {}^{R}d^{\,k} = \left(\frac{{}^{R}d_{max}}{{}^{H}d_{max}} \right) \cdot {}^{H}d^{\,k} \quad \text{(B.21)} $$

With ${}^{R}d_{max}$ set to 0.04 m (a slightly conservative span for the robot), and ${}^{H}d_{max}$ on the order of 0.01, the linear gain was approximately four for most operators. Based on typical human fingertip velocities during free-space motion (i.e., not during object manipulation), an object distance gain of four tends to cause very fast motions of the robot fingers.

Also, operators rarely used the large grasp pose. However, if the gain was decreased (thus preventing an operator from creating the large grasp pose with the robot) operators often became frustrated in attempting to open the robot to a large grasp. To address these problems, a quadratic gain function is used:

$$ {}^{R}d^{\,k} = c_1 \cdot ({}^{H}d^{\,k})^2 + c_0 \cdot {}^{H}d^{\,k} \tag{B.22} $$

The quadratic gain function is beneficial because it decreases the robot tip velocities in the range near typical object sizes but still allows an operator to achieve a large grasp. Of course, the disadvantage is that the robot tip velocities are much higher in the large grasp poses, but as mentioned, this pose was not commonly used, especially during object manipulation.

The coefficients of the Equation B.22 are determined for each individual operator to match his/her maximum span with the robot's large grasp size pose. However, because the glove calibration was not perfect, an operator could not necessarily bring his/her modeled fingertips together[4]. And since virtual object size is increased by the mapping, the robots fingertips would be separated by a visible amount even if the operator physically placed his/her thumb and index fingertips together. The results tended to be disturbing for the operator. Therefore, the operator's minimum achievable distance was mapped to a zero sized virtual object in the robot frame.

Utilizing the data that recorded an operator's "vertical" motions with his/her fingertips together, an average fingertip separation was computed, ${}^{H}d_{min}$. The operator's largest span, ${}^{H}d_{max}$, was recorded during the open-and-close grasp motions. The operator's minimum and maximum sizes were used to compute the quadratic coefficients, such that:

$$ {}^{R}d_{min} = 0 = c_1 \cdot ({}^{H}d_{min})^2 + c_0 \cdot {}^{H}d_{min} \tag{B.23} $$

$$ {}^{R}d_{max} = c_1 \cdot ({}^{H}d_{max})^2 + c_0 \cdot {}^{H}d_{max} \tag{B.24} $$

4. The RMS tip-to-tip error (based on the recorded calibration pose data) was on the order of 4 millimeters.

Because ${}^{H}d_{min}$ is based on an average, operator's could achieve virtual object sizes slightly smaller than this value. To prevent negative object sizes in the robot hand frame, the mapped virtual object size, computed using Equation B.22, is limited to positive values.

B.4.2 Virtual Object Midpoint Scaling

The last step in the virtual object mapping is to scale the motions of the virtual object midpoint such than an operator can utilize the full workspace of the robot. Through preliminary testing of the mapping method, we found it necessary only to scale the Y-axis (vertical) motion of the virtual object midpoint in the robot frame. A unity gain was applied to the midpoint in the X-axis (along the horizontal). Not scaling the X-axis midpoint motions did not significantly affect the operator's ability to achieve the desired pose correspondence shown in Figure B-3 because of the object size scaling. Intentional motions of the virtual object in the direction of $\pm{}^{H}\boldsymbol{h}$ (i.e., along the open-and-close grasp line which is mapped to the horizontal, as shown in Figure B-7) are uncommon during normal object manipulation. Additionally, any midpoint shifting not cancelled out by Equation B.3 is not exacerbated with a unity gain.

Scaling of the mapped virtual motion of the midpoint was necessary to achieve the correspondence shown in poses E and F of Figure B-3. In order to maintain the mapping of the operator's pinch-point to the center of the robot's workspace (pose D), the vertical midpoint gain was applied relative to the desired pinch-point location.

The Y-axis maximum and minimum (un-scaled) mapped midpoint locations are scaled to the Y-axis maximum and minimum achievable robot positions. Since the scaling occurs relative to the desired pinch-point locations it is possible to use different scaling functions for the upper and lower portions of the workspace. During preliminary testing of the mapping method, a quadratic gain for both regions produced the best mapping results. Using a quadratic gain tends to center the operator's motions about the desired pinch-point location and thus in the center of the robot's workspace. The centering effect of the quadratic gain is especially useful during rolling motions (pose G in Figure B-3) because it was found to be difficult to maintain a fixed rotation axis when virtually rolling an object (even with midpoint shifting). Operators could still reach the workspace limits if desired.

To scale the Y-axis midpoint value, the relative vector to the point from the pinch-point was first calculated:

$$ {}^{R_{rel}}\boldsymbol{p}^{k}_{midpoint} = {}^{R}\boldsymbol{p}^{k}_{midpoint} - {}^{R}\boldsymbol{p}_{pinch\text{-}point,des} \tag{B.25} $$

Next, the scaled relative Y-axis midpoint values are computed as follows:

$$ {}^{R_{rel}}p^{k}_{y,\,midpoint} = k_{+y} \cdot ({}^{R_{rel}}p^{k}_{y,\,midpoint})^{2} \qquad \text{if } {}^{R_{rel}}p^{k}_{y,\,midpoint} \geq 0 \tag{B.26} $$

$$ {}^{R_{rel}}p^{k}_{y,\,midpoint} = -k_{-y} \cdot (\left|{}^{R_{rel}}p^{k}_{y,\,midpoint}\right|)^{2} \qquad \text{if } {}^{R_{rel}}p^{k}_{y,\,midpoint} < 0 \tag{B.27} $$

The coefficients are computed for each individual operator using the data recorded from the "vertical" motion (recorded for the determination of θ_{vert}). The coefficients are calculated as follows:

$$ k_{+y} = \frac{{}^{R_{rel}}y_{max,\,ws\ limit}}{max({}^{R_{rel}}p^{i}_{y,\,midpoint})} \tag{B.28} $$

$$ k_{-y} = \frac{{}^{R_{rel}}y_{min,\,ws\ limit}}{min((\left|{}^{R_{rel}}p^{i}_{y,\,midpoint}\right|)^{2})} \tag{B.29} $$

where ${}^{R_{rel}}y_{max,\,ws\ limit}$ and ${}^{R_{rel}}y_{min,\,ws\ limit}$ represents the maximum and minimum vertical distance from the desired pinch-point location that the robot can achieve within the work-space. Because the workspace boundary actually varies with X-axis location, the Y-axis limits are set slightly conservatively.

Once the relative Y-axis midpoint values have been scaled, the midpoint vector is added back completing the mapping of the virtual object parameters to the robot's work-space.

B.5 Method Summary

The virtual object mapping method steps can be summed up as follows:

- Prior to a telemanipulation session, an operator's hand model is calibrated using the method outlined in [Turner 2001]. Once calibrated, fingertip position data are recorded for three separate hand poses and motions: the natural pinch-point pose (fingertips together), open-and-close grasping motions, and motions towards and away from the palm with the fingertips together. The collected data (as well as pre-defined robot parameters) are used to computed the necessary transformation and scaling parameters (θ_{horiz}, θ_{vert}, c_0, c_1, k_{+y}, k_{-y}).
- During a telemanipulation session, the sampled human fingertip position data are used to computed the planar virtual object parameters (size, orientation, midpoint) in the hand frame.
- The virtual object orientation is offset such that the natural open-and-close grasp motion in the hand frame maps to horizontal motion in the robot frame.
- The virtual object midpoint vector is computed relative to the natural pinch-point position and then rotated into the robot hand frame. A skew matrix is applied in addition to the rotation transform to orthogonalize grasping motions with motions towards and away from the palm.
- The rotated relative midpoint vector is translated to the desired pinch-point position in the robot's workspace.
- The virtual object size is scaled to roughly match the maximum human hand span with that of the robot.
- The virtual object midpoint location is scaled (relative to the desired pinch-point) to better match the workspace of the robot.

The mapped and scaled virtual object parameters are used to compute the desired robot fingertip locations for free-space motions or are used to create desired motions of (and forces applied to) a grasped object.

Appendix C: Robot Joint Space and Operational Space Properties

To utilize the operational space formulation for control, we must first fully describe our robot manipulator's joint space properties. The robot manipulator used for the experimental investigation discussed in this thesis is a two-fingered robot hand with two degrees-of-freedom per finger. Each link is connected to the ground or the previous link through a revolute joint (see Figure C-1a and b). It is easiest to first develop the joint space dynamics for the robot hand and then transform the equations into the operational space formulation.

C.1 Link Properties

Each finger is constructed of two main aluminum links. The second link is driven by a four-bar mechanism connected to ground through a pulley and coupler link. This design significantly reduces the inertia of the first link, as compared to placing a motor at the end of the first link.

The link properties (length, mass, center-of-mass location, and polar moment of inertia) were determined empirically during assembly of the robot hand. The parameters for the two fingers are nearly identical. For simplicity, the subscript i will be used when prop-

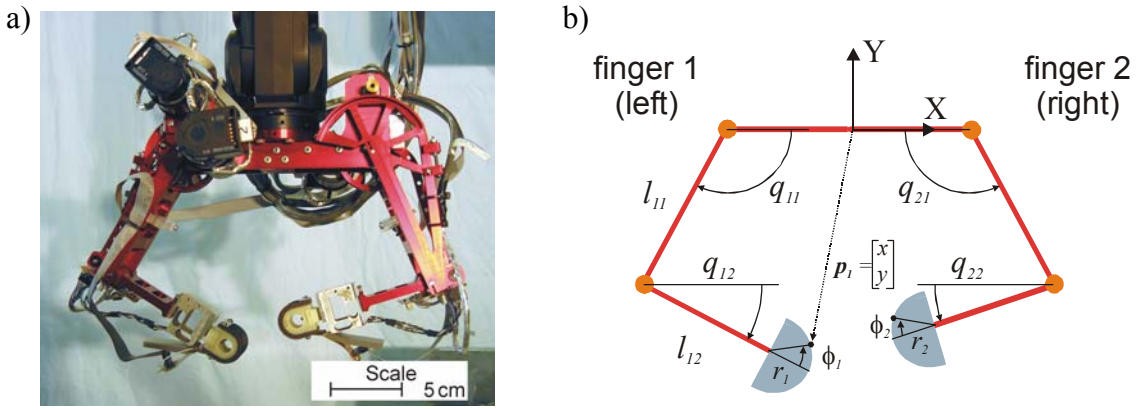

Figure C-1. a) DEXTER, a custom designed two-fingered dexterous robotic hand with two-degrees of freedom per finger. b) Kinematic model of robotic hand. Notice the second link is specified with respect to ground. Additionally, the angles specifying the configuration of the second finger (right), are measured with respect to a z-axis pointing out of the page (opposite of the left finger). This convention allows for a more intuitive specification of the robot angles, i.e., identical angles for the left and right fingers result in a symmetric pose. Also shown is teh contact location angle at the fingertip.

erty values are listed the same for both fingers, otherwise the subscript 1 or 2 will be used to denote the i^{th} finger.

C.1.1 Link Length

The base lengths of each of the links are as follows:

$$l_{base,i} = \begin{bmatrix} 0.1016 \\ 0.0508 \end{bmatrix} \text{m} \tag{C.1}$$

where the j^{th} component represents the length of the j^{th} link. Note that the $l_{base,\,i,\,2}$ value shown is the length of the base of link 2. However, to accurately compute the position of the robot and the Jacobian, the length of the sensor and fingertip must be added. For ease of computation, the link length specified for the second link, l_2, is to the center of the fingertip:

$$l_i = \begin{bmatrix} 0.1016 \\ 0.0996 \end{bmatrix} \text{m} \tag{C.2}$$

The additional length due to the contact location at the fingertip is added in when computing the forward kinematics. If the finger is not in contact, the contact angle, ϕ_i, is assumed to be zero. The radius of the fingertip is $r_i = 0.013\,\text{m}$.

C.1.2 Link Mass

Similar to the link length, there is a base mass measurement associated with the links (not including the sensor and fingertip assemblies). The mass of each link is as follows:

$$m_{base,i} = \begin{bmatrix} 0.11185 \\ 0.04061 \end{bmatrix} \text{kg} \tag{C.3}$$

where $m_{base,i,1}$ is the mass of the upper link and attached link pulley and $m_{base,i,2}$ is the mass of the lower link assembly. To accurately model the total mass of the system, the force sensor and fingertip assembly mass must be included in the mass of the system and results in the following mass parameters for the two links:

$$m_i = \begin{bmatrix} 0.11185 \\ 0.09291 \end{bmatrix} \text{ kg} \tag{C.4}$$

C.1.3 Link Center-of-Mass

The location of the center-of-mass for the first link is measured from the base joint to the second joint. The location of the center-of-mass of the second link is measured from second joint along the symmetric axis of the second link. It is assumed that the center-of-mass for each link lies along these lines, yielding a single value for each link. The location of the center-of-mass of each of the links (without the sensor and fingertip assemblies) is as follows:

$$x_{base,com,i} = \begin{bmatrix} 0.0363 \\ 0.0146 \end{bmatrix} \text{m} \tag{C.5}$$

To include the sensor and fingertip assemblies in the center-of-mass calculation, the second link center-of-mass location must be computed as follows:

$$x_{com,i,2} = \frac{x_{base,com,i,2} \cdot m_{base,i,2} + x_{tip,com} \cdot m_{tip}}{m_{base,i,2} + m_{tip}} \tag{C.6}$$

where $x_{tip,com}$ is measured from the second joint along the second link and m_{tip} is the mass of the sensor and fingertip assembly. Based on the sensor and fingertip assembly properties, the link center-of-mass location is:

$$x_{com,i} = \begin{bmatrix} 0.0363 \\ 0.0474 \end{bmatrix} \text{m} \tag{C.7}$$

C.1.4 Link Inertia

The polar moment of inertia of each link is based on several different measured parameters. The calculation of the inertia for the first link is based on the empirically measured inertia of the link, the inertia of the motor, the inertia of the pulley attached to the motor, and the measured transmission ratio of the cable-capstan drive:

$$j_{i,1} = j_{base,i,1} + \eta_{i,1}^{2} \cdot j_{motorassem,i,1} \tag{C.8}$$

where $j_{base,i,1}$ is of the polar moment of inertia of the first link assembly for the i^{th} fingertip, $\eta_{i,1}$ is the transmission or gear ratio (the radius of the motor pulley over the radius of the link pulley), and $j_{motorassem,i,1}$ is the polar moment of inertia of the motor and motor pulley.

The inertia of the second link includes the same factors, as well as, a lumped parameter model estimate of the inertia of the sensor and fingertips (based on distance to the center-of-mass and mass), given by:

$$j_{i,2} = j_{base,i,2} + \eta_{i,2}^{2} \cdot j_{motorassem,i,2} + m_{tip} \cdot x_{tip,com}^{2} \tag{C.9}$$

Given the base link inertias:

$$j_{base,i} = \begin{bmatrix} 3.411 \times 10^{-4} \\ 6.107 \times 10^{-5} \end{bmatrix} \text{kg} \cdot \text{m}^2, \tag{C.10}$$

the motor assembly inertias:

$$j_{motorassem,i} = \begin{bmatrix} 6.62 \times 10^{-6} \\ 6.62 \times 10^{-6} \end{bmatrix} \text{kg} \cdot \text{m}^2, \tag{C.11}$$

the transmission ratios:

$$\eta_1 = \begin{bmatrix} 0.119195 \\ 0.126955 \end{bmatrix} \quad \eta_2 = \begin{bmatrix} 0.121136 \\ 0.125629 \end{bmatrix}, \tag{C.12}$$

and the tip properties, the polar moment of inertia for the links is as follows:

$$j_1 = \begin{bmatrix} 8.07 \times 10^{-4} \\ 7.11 \times 10^{-4} \end{bmatrix} \text{kg} \cdot \text{m}^2 \quad j_2 = \begin{bmatrix} 8.07 \times 10^{-4} \\ 7.11 \times 10^{-4} \end{bmatrix} \text{kg} \cdot \text{m}^2 \tag{C.13}$$

C.2 Forward Kinematics

The position of the robot is based on encoder measurements and the measured contact location. The forward kinematics used to compute the location at the fingertip contact point is as follows:

$$\boldsymbol{p}_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} -0.5 \cdot w_b + l_{1,1} \cdot \cos(q_{1,1}) + l_{1,2} \cdot \cos(q_{1,2}) + r_1 \cdot \cos(\phi_1 + q_{1,2}) \\ l_{1,1} \cdot \sin(q_{1,1}) + l_{1,2} \cdot \sin(q_{1,2}) + r_1 \cdot \sin(\phi_1 + q_{1,2}) \end{bmatrix}$$
$$\boldsymbol{p}_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0.5 \cdot w_b - l_{2,1} \cdot \cos(q_{2,1}) - l_{2,2} \cdot \cos(q_{2,2}) - r_2 \cdot \cos(\phi_2 + q_{2,2}) \\ l_{2,1} \cdot \sin(q_{2,1}) + l_{2,2} \cdot \sin(q_{2,2}) + r_2 \cdot \sin(\phi_2 + q_{2,2}) \end{bmatrix} \quad \text{(C.14)}$$

where w_b is the distance between the base joints of each finger measured along the x-axis. The position is computed with respect to the frame attached to the robot centered between the two base joints, see Figure C-1b.

Utilizing the forward kinematics, we can compute a Jacobian for each finger. The Jacobian of the left finger (finger 1) is:

$$\begin{aligned} \underline{J}_1 &= \begin{bmatrix} j_{11} & j_{12} \\ j_{21} & j_{22} \end{bmatrix} \\ j_{11} &= -l_{1,1} \cdot \sin(q_{1,1}) \\ j_{12} &= -l_{1,2} \cdot \sin(q_{1,2}) + -r_1 \cdot \sin(\phi_1 + q_{1,2}) \\ j_{21} &= l_{1,1} \cdot \cos(q_{1,1}) \\ j_{22} &= l_{1,2} \cdot \cos(q_{1,2}) + r_1 \cdot \cos(\phi_1 + q_{1,2}) \end{aligned} \quad \text{(C.15)}$$

and the Jacobian of the right finger (finger 2) is:

$$\begin{aligned} \underline{J}_2 &= \begin{bmatrix} j_{11} & j_{12} \\ j_{21} & j_{22} \end{bmatrix} \\ j_{11} &= -l_{2,1} \cdot \sin(q_{2,1}) \\ j_{12} &= l_{2,2} \cdot \sin(q_{2,2}) + r_2 \cdot \sin(\phi_2 + q_{2,2}) \\ j_{21} &= l_{2,1} \cdot \cos(q_{2,1}) \\ j_{22} &= l_{2,2} \cdot \cos(q_{2,2}) + r_2 \cdot \cos(\phi_2 + q_{2,2}) \end{aligned} \quad \text{(C.16)}$$

C.3 Joint Space Dynamics

Our robot manipulator is a multi-link, non-linear, and coupled dynamic system. By modeling the dynamics of the manipulator, a control law can provide feed-forward compensation

for finger inertia, Coriolis and centrifugal forces, and gravitational force. The joint space dynamic model is of the form:

$$A(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{v}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \tau \tag{C.17}$$

where $A(\boldsymbol{q})$ is the configuration dependent mass matrix (joint space kinetic energy matrix), $\boldsymbol{v}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is a vector of centrifugal and Coriolis forces of the form $\boldsymbol{v}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \underline{C}(\boldsymbol{q}) + \underline{B}(\boldsymbol{q}) \cdot \left[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}\right]$, $\boldsymbol{g}(\boldsymbol{q})$ represents the gravity terms, and τ is a vector of joint torques. For our system, the joint positions for the i^{th} finger are specified by $\boldsymbol{q}_i = \left[q_{i,1}\ q_{i,2}\right]$.

The matrices can be computed using a Lagrangian formulation to develop the equations of motions [Khatib 1987]. The mass matrix for the left finger our robot hand is:

$$\begin{aligned}
A_1(\boldsymbol{q}) &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \\
a_{11} &= j_{1,1} + m_{1,2} \cdot l_{1,1}^2 \\
a_{12} &= m_{1,2} \cdot l_{1,1} \cdot x_{com,1,2} \cdot \cos(q_{1,1} - q_{1,2}) \\
a_{21} &= a_{12} \\
a_{22} &= j_{1,2}
\end{aligned} \tag{C.18}$$

and for the right finger:

$$\begin{aligned}
A_2(\boldsymbol{q}) &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \\
a_{11} &= j_{2,1} + m_{2,2} \cdot l_{2,1}^2 \\
a_{12} &= m_{2,2} \cdot l_{2,1} \cdot x_{com,2,2} \cdot \cos(q_{2,1} - q_{2,2}) \\
a_{21} &= a_{12} \\
a_{22} &= j_{2,2}
\end{aligned} \tag{C.19}$$

The formulation of $\boldsymbol{v}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is based on the $\underline{B}$ and $\underline{C}$ matrices (see [Khatib 1987]). The $\underline{B}$ matrix for the left finger is:

$$\underline{B}_1(\boldsymbol{q}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{C.20}$$

and for the right finger:

$$\underline{B}_2(\boldsymbol{q}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{C.21}$$

The $\underline{C}$ matrix for the left finger is:

$$\begin{aligned} \underline{C}_1(\boldsymbol{q}) &= \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \\ c_{11} &= 0 \\ c_{12} &= m_{1,2} \cdot l_{1,1} \cdot x_{com,1,2} \cdot \sin(q_{1,1} - q_{1,2}) \\ c_{21} &= m_{1,2} \cdot l_{1,1} \cdot x_{com,1,2} \cdot \sin(q_{1,2} - q_{1,1}) \\ c_{22} &= 0 \end{aligned} \tag{C.22}$$

and for the right finger:

$$\begin{aligned} \underline{C}_2(\boldsymbol{q}) &= \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \\ c_{11} &= 0 \\ c_{12} &= m_{2,2} \cdot l_{2,1} \cdot x_{com,2,2} \cdot \sin(q_{2,1} - q_{2,2}) \\ c_{21} &= m_{2,2} \cdot l_{2,1} \cdot x_{com,2,2} \cdot \sin(q_{2,2} - q_{2,1}) \\ c_{22} &= 0 \end{aligned} \tag{C.23}$$

The vector product of joint velocities is:

$$\left[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}\right] = \left[\dot{q}_{i,1} \cdot \dot{q}_{i,2}\right] \tag{C.24}$$

The last step in developing the equations of motion is to specify the gravity compensation vector:

$$\boldsymbol{g}_1 = \begin{bmatrix} m_{1,1} \cdot x_{com,1,1} \cdot \cos(q_{1,1}) + m_{1,2} \cdot l_{1,1} \cdot \cos(q_{1,1}) \\ m_{1,2} \cdot x_{com,1,2} \cdot \cos(q_{1,2}) \end{bmatrix} \cdot g \tag{C.25}$$

$$\boldsymbol{g}_2 = \begin{bmatrix} m_{2,1} \cdot x_{com,2,1} \cdot \cos(q_{2,1}) + m_{2,2} \cdot l_{2,1} \cdot \cos(q_{2,1}) \\ m_{2,2} \cdot x_{com,2,2} \cdot \cos(q_{2,2}) \end{bmatrix} \cdot g \tag{C.26}$$

where $g = 9.81\ \mathrm{m/s^2}$. Additionally, a term was added to $g_{i,1}$ to estimate the gravity term due to the four-bar coupler link.

C.4 Operational Space Dynamics

As discussed in Chapter 5, the robot fingers were controlled in operational space. Based on [Khatib 1987], we can map the joint space dynamics to operational space for a given point of interest, the operational point. For our manipulator, the operational point is the point of contact at the fingertip (if the fingertip is not in contact then $\phi_i = 0$ is used).

The relationship between joint space and operational space dynamics is formed by using the Lagrangian formalism and equating the quadratic form of kinetic energy. The relationships between the joint space equation of motion components and operational space equation of motion components is as follows.

The operational space configuration dependent mass matrix or kinetic energy matrix is computed by:

$$\Lambda(\boldsymbol{x}) = \underline{J}^{-T}(\boldsymbol{q}) \cdot \underline{A}(\boldsymbol{q}) \cdot \underline{J}^{-1}(\boldsymbol{q}) \tag{C.27}$$

The centrifugal and Corilois operational space vector is:

$$\mu(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \left[\underline{J}^{-T}(\boldsymbol{q}) \cdot \underline{B}_{op}(\boldsymbol{q}) - (\Lambda(\boldsymbol{q}) \cdot \underline{H}_{op}(\boldsymbol{q})) \right] \cdot \left[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}} \right] \tag{C.28}$$

where:

$$\underline{B}_{op}(\boldsymbol{q}) \cdot \left[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}} \right] = \underline{B}(\boldsymbol{q}) \tag{C.29}$$

and:

$$\underline{H}_{op}(\boldsymbol{q}) \cdot \left[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}\right] = \dot{\underline{\mathrm{J}}}(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} \tag{C.30}$$

And lastly, the gravity vector is computed by:

$$\boldsymbol{p}(\boldsymbol{x}) = \underline{\mathrm{J}}^{-T}(\boldsymbol{q}) \cdot \boldsymbol{g}(\boldsymbol{q}) \tag{C.31}$$

Finally, the end effector equations of motion in operational space can be written as:

$$\Lambda(\boldsymbol{x}) \cdot \ddot{\boldsymbol{x}} + \mu(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \boldsymbol{p}(\boldsymbol{x}) = \boldsymbol{F}. \tag{C.32}$$

Appendix D: Developing the Grasp Transformation

Given our two fingered robot manipulator and an object within the robot's grasp, we can develop the transformations necessary to compute fingertip forces from desired forces applied to the body. The transformation can also be used to compute the velocity of the body based on the velocity of the robot's fingertips. This appendix will describe the development of the grasp transformation matrix for a planar object in a two fingered grasp. For a more general treatment of the problem, see [Mason and Salisbury 1985].

D.1 Grasp Transformation Matrix

The grasp transformation matrix for a given object is based on the number and the type of contacts. The type of contact depends on the geometries of the object and finger and determines the nature of the contact constraint. Common contact types include: point contact, line contact, and planar contact, each can exist with or without friction. Other contact constraints, such as rolling, place kinematic constraints on the motion of the object and finger.

For our planar system, the contact type between the finger and the object is assumed to be a point contact with friction.[1] The contact force with respect to the surface of the object can be resolved into a normal force and tangential force, as in Figure D-1. In order

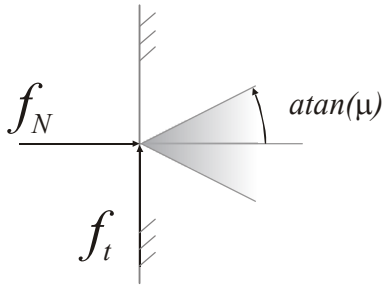

Figure D-1. Point contact with friction. The contact force can be resolved into normal and tangential forces.

1. Both object and finger have depth with the plane, thus the contact type is technically a line contact with friction. However, if we assume that the center-of-gravity of the object lies with in the center plane of the robot fingertips, a point contact model with friction can sufficiently describe the contact conditions.

for the robot fingertips to maintain contact with the object and prevent slippage, the resultant contact force must lie within the friction cone, or:

$$f_N \geq \frac{|f_t|}{\mu} \tag{D.1}$$

where μ represents the coefficient of friction between the fingertip and object surfaces.

D.1.1 The Wrench Matrix

In general, with each particular type of contact, there is associated a set of lines about which generalized forces or *wrenches* may be exerted. We can formulate a matrix of the set of wrenches exerted on a given object:

$$\underline{W} = \left[\boldsymbol{w}_1 \ \ldots \ \boldsymbol{w}_n\right] \tag{D.2}$$

The general form of the wrench matrix, $\underline{W}$, is a $6 \times n$ matrix where each wrench, $\boldsymbol{w}_i$, is represented in screw coordinates [Mason and Salisbury 1985]. To apply an arbitrary net wrench, $\boldsymbol{w}$, to an object, we must find, $\boldsymbol{c}$, a vector of contact wrench intensities, that satisfies:

$$\underline{W} \cdot \boldsymbol{c} = \boldsymbol{w} \tag{D.3}$$

The first step is to develop the wrench matrix. Figure D-2 illustrates a two-fingered grasp of an object with both contacts modeled as point contacts with friction. (It is assumed that the ratio of the normal and tangential forces obey Equation D.1 so that contact is maintained). The force at each contact can be resolved into forces along the axes of a coordinate

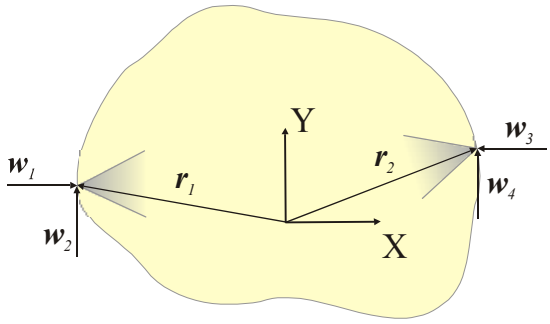

Figure D-2. Grasped object with point contacts. Each point contact has friction between the finger (not shown) and the object surface.

frame embedded in the object. Each wrench, or external force, can be represented in terms of screw coordinates with respect to a reference frame in the object. In the planar case, the wrench vectors can be easily developed through a simple force balance.

First, an arbitrary wrench applied to the object can be represented with a 3×1 vector of planar forces and a moment about the axis perpendicular to the plane:

$$\boldsymbol{w} = \begin{bmatrix} f_x \\ f_y \\ m_z \end{bmatrix} \tag{D.4}$$

and is defined with respect to the reference frame embedded in the object. Second, the vector, $\boldsymbol{c}$, represents the wrench intensities (or magnitudes of the forces) of each of the contact wrenches, $\boldsymbol{w}_i$. We can then express Equation D.3 as:

$$\begin{bmatrix} f_x \\ f_y \\ m_z \end{bmatrix} = \underline{W} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} \tag{D.5}$$

Lastly, the wrench matrix, $\underline{W}$, now represents a 3×4 matrix that can be developed based on a force balance of forces applied at the contact to forces applied to the body. By inspection (see Figure D-2), the wrench matrix follows:

$$\underline{W} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ -r_{1y} & r_{1x} & r_{2y} & r_{2x} \end{bmatrix} \tag{D.6}$$

where r_{ix} and r_{iy} represent the x and y components of the vector from the object frame to the i^{th} contact.

It is also important to recognize that for a given disturbance wrench, $\boldsymbol{w}$, the contact intensity vector will not be unique. If we are to solve for the contact wrench intensities based on a given disturbance wrench, the solution would be of the form:

$$\boldsymbol{c} = \boldsymbol{c}_p + \lambda \cdot \boldsymbol{c}_h \tag{D.7}$$

where $\boldsymbol{c}_p$ is a particular solution to Equation D.3 and $\boldsymbol{c}_h$ is the homogeneous solution. The scalar, λ, represents the magnitude of the *internal force* applied to the object.

For example, referring to Figure D-2, assume $\boldsymbol{r}_1 = \begin{bmatrix} -1 & 0 \end{bmatrix}^T$ and $\boldsymbol{r}_2 = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$, then we can express Equation D.5 as:

$$\begin{bmatrix} f_x \\ f_y \\ m_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} \tag{D.8}$$

Also, suppose that the desired force on the object is $\boldsymbol{w} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$ (a force along the y-axis of 1 unit). If we solve Equation D.3 for the contact wrench intensities, we find:

$$\boldsymbol{c} = \begin{bmatrix} 0 \\ 0.5 \\ 0 \\ 0.5 \end{bmatrix} + \lambda \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{D.9}$$

As we can see, we are free to specify any arbitrary value for the internal force, λ, and the internal force value will not affect the desired force applied to the object.[2]

D.1.2 Constructing the Grasp Transformation Matrix

The wrench matrix and the homogeneous solution (the null space of $\underline{W}$) can be used to formulate the grasp transformation matrix. The grasp transformation is constructed by augmenting the wrench matrix with the homogeneous solution. For the planar case, this results in the following matrix:

$$\underline{G}^{-T} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ -r_{1y} & r_{1x} & r_{2y} & r_{2x} \\ r_{12x} & r_{12y} & r_{12x} & -r_{12y} \end{bmatrix} \tag{D.10}$$

2. We assume that the contact intensities, $\boldsymbol{c}$, are of an appropriate magnitude relative to each other such that contact with the object is maintained.

where r_{12x} and r_{12y} represent the x and y components a unit vector from the left contact (number 1) to the right contact (number 2) defined by:

$$\boldsymbol{r}_{12} = \frac{\boldsymbol{r}_2 - \boldsymbol{r}_1}{|\boldsymbol{r}_2 - \boldsymbol{r}_1|} \tag{D.11}$$

Assuming the homogeneous solutions concatenated to the wrench matrix spans the null space of the wrench matrix, the grasp transformation matrix $\underline{G}^{-T}$ will be square and invertible. For the planar case, we can now define a vector of external and internal forces and moments on the object:

$$\boldsymbol{F}_{obj} = \begin{bmatrix} f_x \\ f_y \\ m_z \\ f_{int} \end{bmatrix} \tag{D.12}$$

where λ is replaced with f_{int}. If we define a vector of fingertip forces magnitudes, $\boldsymbol{F}_{tip}$, in which magnitudes are for forces occurring along the wrench axes at each contact (which is equivalent to the wrench intensity vector $\boldsymbol{c}$)

$$\boldsymbol{F}_{tip} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix}, \tag{D.13}$$

then:

$$\boldsymbol{F}_{obj} = \underline{G}^{-T} \cdot \boldsymbol{F}_{tip} \tag{D.14}$$

and

$$\boldsymbol{F}_{tip} = \underline{G}^{T} \cdot \boldsymbol{F}_{obj}. \tag{D.15}$$

D.1.3 Obtaining Object and Fingertip Velocities

Similar to the force-velocity duality of the Jacobian, the grasp transformation matrix can be used to defined a transformation between the object velocity and the velocity of the contact points. First, we must define a vector to represent the twist intensities per unit time occurring along the wrench axes at each contact. A twist intensity is a screw system representation of motion, for more details see [Mason and Salisbury 1985]. The vector of twist intensities is as follows:

$$\boldsymbol{V} = \begin{bmatrix} d_1 & \dots & d_n \end{bmatrix}^T \tag{D.16}$$

where n is the number of contact wrenches. In our simplified planar case, the twist intensity vector, $\boldsymbol{V}$, can be thought of as the velocity of the contact locations along the axes defined by the reference frame embedded in the object:

$$\dot{\boldsymbol{x}}_{tip} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix} \tag{D.17}$$

We can also define a vector to represent the object's linear and angular velocity. The vector also includes the objects's (virtual) velocities resulting from the deformations of the object. In the planar case, the first three components represent the planar velocity and the angular velocity (about an axis perpendicular to the plane) of the object. An additional component is added to represent the virtual velocity from the deformation along the line between the two contact points:

$$\dot{\boldsymbol{x}}_{obj} = \begin{bmatrix} v_x \\ v_y \\ \omega \\ \gamma \end{bmatrix} \tag{D.18}$$

Where the object velocities are measured with respect to a frame coincident to the object frame but fix in a world frame.

If we equate the power input with the power output (see [Mason and Salisbury 1985]), it follows that:

$$\dot{\boldsymbol{x}}_{tip} = G^{-1} \cdot \dot{\boldsymbol{x}}_{obj} \tag{D.19}$$

and, by inversion:

$$\dot{\boldsymbol{x}}_{obj} = G \cdot \dot{\boldsymbol{x}}_{tip}. \tag{D.20}$$

D.2 Computing the Internal Force

Using the developed grasp transformation to compute the internal force based on measured fingertip forces (contact intensities) will lead to an incorrect calculation of internal force under certain circumstances. In cases where the fingertip forces can contribute to an internal force and cause an acceleration of the object, the grasp transform as specified in Equation D.10 must be modified.

As advocated by Yoshikawa and Nagai [1991], the internal force can be computed based on choosing the internal force from the minimum of the forces projected along the line of contact. First we can define the *left* internal force as:

$$f_{int,left} = \boldsymbol{F}_1^{\,T} \cdot \boldsymbol{r}_{12} \tag{D.21}$$

and the *right* internal force as

$$f_{int,right} = \boldsymbol{F}_2^{\,T} \cdot -\boldsymbol{r}_{12} \tag{D.22}$$

where $\boldsymbol{F}_i$ represents the contact force, as expressed in the object frame, of the i^{th} finger contact (i.e., $\boldsymbol{F}_1 = \begin{bmatrix} f_1 & f_2 \end{bmatrix}^T$ and $\boldsymbol{F}_2 = \begin{bmatrix} -f_3 & f_4 \end{bmatrix}^T$). The internal force is computed as follows:

$$f_{int} = min(f_{int,left}, f_{int,right}) \tag{D.23}$$

This method of computing the internal force can be incorporated into the grasp transformation formulation so that commanded forces on the object result in proper forces at the robot's fingertips and vise-versa. Therefore, if the minimum of $(f_{int,left}, f_{int,right})$ is based on $\boldsymbol{F}_1$ then the grasp transformation is formulated as follows:

$$G^{-T}_{left} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ -r_{1y} & r_{1x} & r_{2y} & r_{2x} \\ r_{12x} & r_{12y} & 0 & 0 \end{bmatrix} \tag{D.24}$$

and if the minimum is based on $\boldsymbol{F}_2$, the grasp transformation is:

$$G^{-T}_{right} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ -r_{1y} & r_{1x} & r_{2y} & r_{2x} \\ 0 & 0 & r_{12x} & -r_{12y} \end{bmatrix} \tag{D.25}$$

The proper grasp matrix should be used when computing the necessary fingertip forces required to apply the desired external and internal forces to the grasped object.

Appendix E: Post Experiment Subject Questionnaire

Your case order (first to last): ___ ___ ___ ___ ___ ___

1. Did you prefer one case more than others? If so, in what way?

2. Did you dislike one case more than others? If so, in what way?

3. Please rate each case from Preferred to Disliked (in terms of completing the task).

	Preferred				Disliked
Case 1	☐	☐	☐	☐	☐
Case 2	☐	☐	☐	☐	☐
Case 3	☐	☐	☐	☐	☐
Case 4	☐	☐	☐	☐	☐
Case 5	☐	☐	☐	☐	☐
Case 6	☐	☐	☐	☐	☐
Case 7	☐	☐	☐	☐	☐

4. Was one case more difficult to use than the others? If so, in what way?

5. Was one case easier to use than the others? If so, in what way?

6. Please rate each case from Easy to Difficult (in terms of completing the task).

	Easy				Difficult
Case 1	☐	☐	☐	☐	☐
Case 2	☐	☐	☐	☐	☐
Case 3	☐	☐	☐	☐	☐
Case 4	☐	☐	☐	☐	☐
Case 5	☐	☐	☐	☐	☐
Case 6	☐	☐	☐	☐	☐
Case 7	☐	☐	☐	☐	☐

7. In the applicable cases, did you use the fingertip LEDs while completing the task?

In these cases, did you find the task easier or more difficult?

Easier		No Difference		More Difficult
☐	☐	☐	☐	☐

Comments:

8. In the applicable cases, did you use the audio tones while completing the task?

   In these cases, did you find the task easier or more difficult?

Easier		No Difference		More Difficult
☐	☐	☐	☐	☐

   Comments:

9. In the cases where you were told the robot was assisting, did you find the task easier or more difficult?

Easier		No Difference		More Difficult
☐	☐	☐	☐	☐

   Comments:

10. In the applicable cases, did you notice the *target window force reduction*?

    Did it help or hinder you ability to complete the task?

Helped		No Difference		Hindered
☐	☐	☐	☐	☐

    Comments:

11. Did you feel there was a change in the relative degree of difficulty of the **control case after using cases with intervention and/or alarms** (*i.e.*, case 1 was not your first case)? Please rank this relative degree of difficulty.

More Difficult		No Difference		Less Difficult
☐	☐	☐	☐	☐

    Comments:

12. General comments: