The Thesis Committee for Jack Lyle Thompson
Certifies that this is the approved version of the following thesis:

Redesigning the Human-Robot Interface: Intuitive Teleoperation of Anthropomorphic Robots

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Sheldon M. Landsberger

Co-supervisor:

Mitchell W. Pryor

Redesigning the Human-Robot Interface:
Intuitive Teleoperation of Anthropomorphic Robots

by

Jack Lyle Thompson, B.S.; B.E.

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin
December 2014

Acknowledgements

I would like to thank my supervisors, Dr. Mitch Pryor and Professor Sheldon Landsberger, for their guidance, support and fine sense of humor; the Department of Energy-Office of Nuclear Energy for providing me with a Nuclear Energy University Program Graduate Fellowship; Mr. Doug Kautz and Dr. Brad Storey of Los Alamos National Laboratory for their ongoing support of the Nuclear Robotics program, and for the use of their robots; and Dr. Steve Biegalski and Ms. Dana Judson of the Nuclear Engineering Teaching Laboratory as well as Ms. Jenna Payne of the NEUP Integration Office at Idaho National Laboratory for their administrative and logistical assistance.

Many thanks and much love to my parents.

This material is based upon work supported under a Department of Energy Nuclear Energy University Program Graduate Fellowship.

Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the Department of Energy-Office of Nuclear Energy.

Abstract

Redesigning the Human-Robot Interface: Intuitive Teleoperation of Anthropomorphic Robots

Jack Lyle Thompson, M.S.E.

The University of Texas at Austin, 2014

Supervisor: Sheldon M. Landsberger

Co-supervisor: Mitchell W. Pryor

A novel interface for robotic teleoperation was developed to enable accurate and highly efficient teleoperation of the Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) system and other robotic systems. In order to achieve a revolutionary increase in operator productivity, the bilateral/master-slave approach must give way to shared autonomy and unilateral control; autonomy must be employed where possible, and appropriate sensory feedback only where autonomy is impossible; and today's low-information/high feedback model must be replaced by one that emphasizes feedforward precision and minimal corrective feedback. This is emphasized for task spaces outside of the traditional anthropomorphic scale such as mobile manipulation (i.e. large task spaces) and high precision tasks (i.e. very small task spaces). The system is demonstrated using an anthropomorphically dimensioned industrial manipulator working in task spaces from one meter to less than one millimeter, in both simulation and hardware. This thesis discusses the design requirements and philosophy of this interface, provides

a summary of prototype teleoperation hardware, simulation environment, test-bed hardware, and experimental results.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Motivation

> "*Most accidents are attributed to human error, but in almost all cases the human error was the direct result of poor design. The principles that guide a quality, human-centered design are not relevant just to a more pleasurable life – they can save lives.*" –Don Norman, in *The Design of Everyday Things* [1]

Design is of critical importance in a nuclear facility. Buildings and building services must be designed to survive natural disasters as well as physical attack by hostile forces without releasing radioactivity into the environment, while also preventing the possibility of a criticality. Every day, hundreds of glovebox workers stream into PF-4 in TA-55 at Los Alamos National Laboratory. The nature of their work means their health and safety is continually at risk from exposure to:

- Gamma and neutron radiation, which pass through shielding
- Skin, and in some cases, internal contamination from glove breaches
- Repetitive strain injury due to glovebox design
- Ergonomic injury due to glovebox design

In addition, the gloves used by glovebox workers:

- Inhibit worker dexterity and productivity
- Cause repetitive strain injury
- Decay rapidly when exposed to alpha particles
- Are frequently breached during routine operations

1.2 Remote Operations

To mitigate health risks to radiation workers, equipment and processes are designed so that worker radiation dose is kept As Low As Reasonably Achievable (the principle of ALARA). Worker dose can be reduced in three ways: by increasing distance, increasing shielding, or by reducing the time of exposure. Typically, the only viable solution is to place additional distance between workers and radioactive material. Remote manipulators were first developed in the 1940's for this very purpose [2].

However, remote manipulators for glovebox work tend to be purely mechanical, and require manual actuation by human operators in order to move them about the workspace and to actuate their grippers. This significantly limits the payload operators can lift, limits the dexterity of the operator in performing tasks, and limits the workspace volume. It also leads to repetitive strain injuries, even in systems with electrically assisted gripper actuation. These manipulators are also extremely expensive.

Industrial robots, on the other hand, are low cost, compact, lightweight and robust, with Mean Time Between Failures (MTBFs) on the order of tens of thousands of hours of continuous operation [3]. However, most industrial robots are far from ideal for the task of robotic telemanipulation. Industrial robots are mechanically non-compliant (stiff and non-backdrivable), making them unsuitable for contact handling without external sensing and control. They are typically configured for position rather than velocity control, making it impractical to return force feedback to the operator. The control rate of an industrial robot is typically no more than 60Hz, which among other factors limits the velocity at which a robot can respond to commands or to a dynamic environment. Finally, almost all human-robot interfaces are poorly equipped to perform spatially oriented and/or teleoperation tasks in real time.

The best of these interfaces, the master-slave (bilateral) telemanipulation interface, is extremely expensive, ergonomically problematic, and lacks the sensor and actuator bandwidth to provide true haptic feedback to an operator. These incomplete haptic signals lead to increased operator cognitive load, especially when performing fine work.

In the glovebox, where there are only a few small windows in order to maximize radiation shielding, visibility is poor. The corrosive environment means that practically everything inside the box is made of stainless steel; consequently almost everything in a glovebox is the same color, and so there is minimal visual contrast. Objects frequently occlude one another, and may not even be visible. Even under ideal circumstances, visual feedback requires considerable cognitive effort, but the conditions of the glovebox environment undermine the already poor visual feedback achievable through a small window, making haptic feedback necessary for task completion.

Figure 1: Glovebox Operations [4]

1.3 TASK COMPLETION

Efficient task completion requires a suitably efficient interface, but providing one at low cost is difficult, and any such interface must be thoughtfully designed to prevent ergonomic injuries. This all but eliminates existing pointing devices, which are both inefficient for spatially oriented tasks, and lead to repetitive strain injury with overuse. Natural User Interfaces (NUIs) are promising [5], and postural gesture recognition has been applied to the problem of robotic control [6], [7] and humanoid teleoperation [8]. Hand gesture recognition has also been used for robotic control with some success [9], though at best it replaces a pointing device. It, too, can be ergonomically unsafe; while the hands can adopt many different poses, and produce a diverse gestural vocabulary, most hand poses require the use of secondary muscles that tire and injure easily, such as the palmar and dorsal interossei [10]. While voice recognition has become much more accurate over time, and technologies such as throat- and bone conduction microphones as well as active noise cancellation improve performance even further, the small but significant false positive rate means voice recognition is still poorly suited for use in a noisy environment such as a manufacturing facility where ambient speech, process noise and alarms are all sources of confusion [11].

All of these interfaces hinder the operator in completing spatially oriented contact handling tasks, as they introduce additional layers of abstraction between the operator's plan and the execution. From the point of view of the operator, the best interface is no interface at all. After all, to perform a task with one's hands, one uses one's hands. Therefore, **we hypothesize that for a spatially oriented task such as contact handling, the ideal control input for a teleoperator is the (complete) state of the operator's hand, and the ideal end-effector is a replica of the operator's hand.**

Instrumented gloves provide pose data but typically require additional hardware to track their position in space, and complete systems such as those used in motion capture for the movie industry can be extremely expensive. However, recent developments in optical hand tracking now permit the optical tracking of position and pose, without additional sensors, via depth imagery [12]. This has opened up the possibility of a robotic telemanipulation system that could meet hitherto conflicting requirements for efficient task completion, low cost and zero ergonomic injury.

However, it's extremely difficult to provide useful haptic feedback to an operator when tracking their ungloved hands with a camera. A number of technologies have been proposed, from subsonic acoustic waves [13] to direct stimulation of the peripheral motor nerves [14]; the latter could also eliminate the need for a camera. However, even if these technologies turn out to be commercially viable, much more development is required before they can be adopted on any more than an experimental basis.

On closer inspection **the basic assumption of bilateral telemanipulation – that haptic feedback must be provided to the operator – is based on a flawed premise.** Human operators do not need force feedback signals any more than airline pilots need to stick a finger out the window to measure their airspeed. What is necessary is instrumentation, relevant to an appropriate control strategy, and for the system to rely not on the human operator's neural circuitry to respond to force signals, but on software.

In the Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) system, operated by The University of Texas at Austin's Nuclear Robotics Group, the combination of a world model populated by depth imagery, a high bandwidth programmable logic controller (PLC) based industrial robot controller, a commercial off-the-shelf (COTS) force/torque sensor and an intelligent COTS force-closing gripper come together to create a system capable of safely handling contact tasks in software, sharing control with a remote operator. The purpose of this research is to develop a hands-free teleoperation interface. To move the end effector, the operator need only move their hand in space. To control the sensitivity of the motion, the operator need only turn a dial or operate a foot pedal.

By providing visual feedback to the operator, and force/torque feedback to the robot control software, we can leverage the benefits of unilateral and bilateral teleoperation, and free up the operator to "concentrate the greater portion of his mental attention on the work that is to be done rather than on how to control the manipulator" [15].

Figure 2 illustrates the architecture of the proposed system. The individual components are presented in this thesis. Notable features include optical tracking of operator hand pose, and separate control loops for high level (visual/user) and low level (haptic/robot) feedback control.

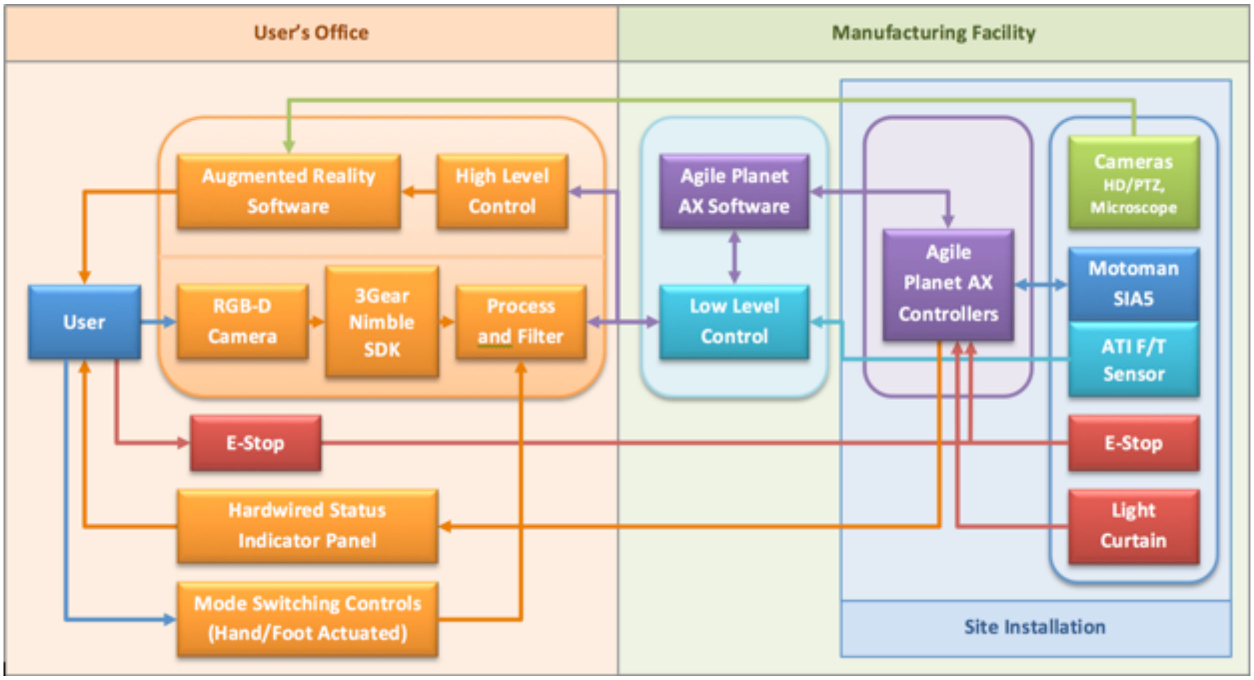

Figure 2: Block Diagram of the Proposed Teleoperation System

1.4 Scope and Objective of Research

The purpose of this research is to design a low cost, robust, responsive, efficient, user-centered interface for robotic teleoperation.

The complete implementation of such a system is beyond the scope of this work. As such, this research will focus on the developing the requirements for such a system, the development of a prototype to showcase this system's core functionality by focusing on motion at extremely small scales, and demonstration of the validity of this approach by experiment.

1.5 Building on Previous Work

This research builds upon the work of many others at the University of Texas and elsewhere.

Robot control in software and hardware was performed using technology from Agile Planet, now a part of Yaskawa Motoman. Agile Planet's Kinematix and AX software libraries evolved from earlier work performed in the University of Texas' Robotics Research Group [16].

Optical hand position and pose tracking was performed using the PrimeSense Carmine depth camera, and with the Nimble SDK from 3Gear Systems, which evolved from camera based hand tracking research by Robert Wang and others at Massachusetts Institute of Technology [12], [17].

1.6 ORGANIZATION OF THE REPORT

The remainder of the report is organized as follows:

- **Chapter Two** is a review of teleoperation, robotic telemanipulation, human factors and user interface design.
- **Chapter Three** is an overview of the key technologies required for this work, from quaternions to the Robot Operating System.
- **Chapter Four** describes the proposed system, namely the hardware and software required to perform optical hand pose and position tracking from depth imagery, as well as the hardware and high and low level control software required to use this data for robotic telemanipulation.
- **Chapter Five** summarizes a demonstration of the proposed system, and provides experimental results.
- **Chapter Six** discusses this research, and highlights possibilities for future work.

Chapter 2: Background

The first part of this chapter will focus solely on the aspects of teleoperation and telerobotics pertinent to this research, while the second part will provide an overview of key aspects of human factors and user interface design. A substantive survey of telerobotics can be found in [18, Ch. 31]. Additional material on earlier developments can be found in Sheridan's seminal paper [19], as well as its 1995 update [20]. Further material on bilateral teleoperation can be found in [21]. The second part of this chapter will discuss relevant physiological and cognitive human factors, and outline the basic principles of user interface design that have informed the choices made in this research.

2.1 TELEOPERATORS

The development of modern teleoperator technology was driven by the need to process spent nuclear fuel behind radiation shielding during the Manhattan Project [22], although people have been performing work at a distance for thousands of years using all kinds of hand tools, from chopsticks to fire tongs [23].

The first modern teleoperators developed by Goertz et al. at Argonne National Laboratory were electrically actuated, with operators adjusting the position of each axis via open-loop push-button control [18, Ch. 31], [24]. These were unilateral manipulators, meaning that no information about the forces on the end-effector was transmitted back to the operator. The awkward controls and lack of feedback meant they were "slow and somewhat awkward to operate" [18, Ch. 31] and were quickly supplanted by mechanically passive tape- and cable-driven bilateral manipulators.

Bilateral manipulators (also known as master-slave manipulators) are so named for their transmission of forces between operator and end-effector, as well as between end-effector and operator. The operator controls "a 'master' arm, with articulation resembling a man's except that the elbow telescopes instead of bending. A 'slave' arm duplicates motion in all degrees of freedom, and feeds back to the master any resistance it encounters" [25].

Goertz noted in 1964 that with regards to the handling of radioactive materials in hot cells, “the speed of performing work with the master-slave [or bilateral] manipulator system is about 10% the speed of doing the same type of work with the hands. The rate of performing work with the unilateral electric types of manipulator is about 10 to 100 times less than this. That is, they can be operated to do work at about 0.1% to 1% the speed of the human hands...” [15]

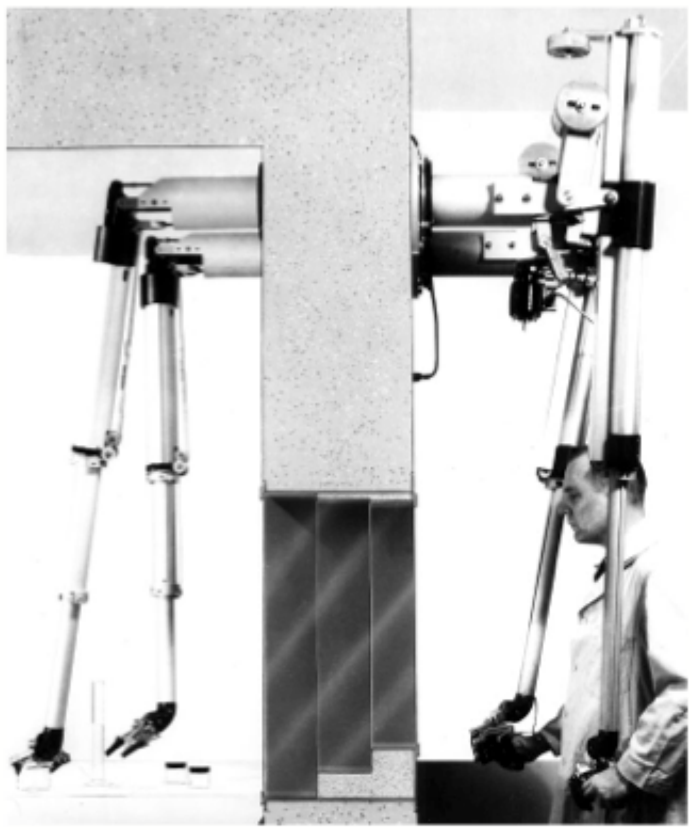

Figure 3: Mechanical Bilateral (Master-Slave) Manipulator [26]

Goertz went on to explain “The reason that the master-slave manipulator is as good as it is, we believe, is primarily due to the natural movements of the handle tongs, force feedback to give ‘feel’, and the high reversibility of master to slave and slave to master. We have found that a manipulator operator having several months of experience can concentrate the greater portion of his mental attention on the work that is to be done rather than on how to control the manipulator.” [15]

While the “natural movements” of the master-slave manipulator allow workers to accomplish tasks more quickly and with less cognitive load, operation of a mechanical pantograph relies solely on the operator’s physical strength. Strictly mechanical teleoperators are, therefore, of limited utility in many environments e.g. extremely high radiation fields, where the thickness of shielding is so great that tape/cable flexure as well as link inertia become severe impediments to operation [27].

2.2 SERVOMANIPULATORS

Ever since the development of the master-slave manipulator, engineers have tried to integrate electrical and hydraulic systems into parts of the linkage with varying degrees of complexity and success. On the low end, simple alterations such as push button electric controls to open and close a gripper allow operators to work with heavier loads, but at the cost of a loss of haptic feedback. On the high end, complete mechanical decoupling was achieved between fully instrumented and electrically actuated master and slave units under feedback control as early as 1954 [28].

What may seem like a trivial control problem – to simply make the slave do whatever the master does – turns out to be anything but trivial when the loop is closed and the master controls reflect the state of the slave. The finite mechanical bandwidth of sensors and actuators introduce lag into the system, as do signal processing and propagation delays in feedforward and feedback. Under lag, the closed-loop dynamics between master and slave can easily become unstable.

The control strategy most commonly adopted is position-force [29], in which the master controller transmits the position of the master to the slave, and the slave transmits forces and velocities back to the operator via the master. The reason for this asymmetry is twofold. First, position-position control leads to oscillation that must be damped either by the operator’s muscles causing fatigue (*in extremis,* a loss of control), or by introducing significant damping into the system leading to sluggish or “spongy” controls. Second, while position control lends itself to accuracy, velocity control leads to smoother motion; because human motion perception is relative, it's relatively less important for the master to accurately reflect the absolute position of the slave.

Figure 4: Teleoperated Servomanipulator at the Joint European Torus Fusion Reactor [30]

Position-force control is nonetheless problematic. The forces reflected to the master are proportional to the master-slave mass ratio. With a 40 kg slave and a 1 kg master, when the slave makes contact with its environment the master's instantaneous recoil velocity will be 79 times that of the slave, pulling the slave (and quite possibly the operator) along with it [31]. Unless this recoil is severely damped, contact can't be maintained and contact tasks can't be performed. This damping, however, makes the teleoperator feel sluggish and unresponsive, and prevents the operator from achieving high accuracy.

The operators of servomanipulators view their workspace through closed-circuit television. While it's typically much more comfortable to look at television screens than to peer through a small window, fixed monocular cameras aren't nearly as sensitive as the eyes to variations in light and color, and their resolution leaves much to be desired. They also make it impossible to perceive depth except through contextual cues; an operator at the glovebox window can perceive depth directly either by binocular (stereo) vision, or by moving their head to obtain depth information via parallax. The use of multiple camera angles, pan-tilt-zoom (PTZ) cameras, etc. can help to improve operator situational awareness, however no form of telepresence has yet come close to actual presence in terms of the quantity, quality and timeliness of sensory feedback available to the operator.

2.3 SUPERVISORY AND SHARED CONTROL

In the early 1960's Sheridan and Ferrell, motivated by the space race, studied the effects of time delay on teleoperation [32]. While it was shown that tasks could be accomplished with the open-loop move-and-wait approach, it requires trading a great deal of speed for accuracy and stability. Further, the round-trip time delay of radio communications between Earth-based ground stations and spacecraft on orbit meant that closed-loop stability could only be ensured by a (suitably complex) controller onboard the spacecraft. The addition of this remote control loop was the first true supervisory control, in the sense that "a human operator [could now] act as a supervisor, intermittently communicating to a [subordinate]… while the subordinate telerobot executes the task based on the information received from the human operator plus its own sensing and intelligence" [19].

Allowing the remote controller to autonomously ensure its own stability naturally led to a desire to expand the scope of this autonomy to include other programmed behaviors, in which a remote system could act on high level behavioral commands such as "weld the door panel". The development of computerized control systems capable of running nontrivial programs fueled this notion, however as it became clear that the problem of artificial intelligence was fiendishly difficult, the notion of supervisory control at a very high level gave way to more realistic notions, and a more pragmatic partitioning of responsibilities between human and robot. The notion of shared control reflects the reality that humans are better at some tasks, and robots at others, so that at different times the human operator and robot ought to be able to intelligently allocate responsibilities between themselves. For example, on detecting an anomalous condition for which there is no suitable preprogrammed response, a robot may defer control to the human operator; similarly, on completing a particularly complex operation, a human operator may give control to the autonomous system, similar to the way a surgeon may hand over the (relatively simple) procedure of closing a patient to an assistant.

Today, the vast majority of robotic telemanipulation systems in service have no autonomous capabilities beyond gravity compensation. They are stabilized on the basis of passivity, which can't be guaranteed in the presence of an energy source such as a human operator, another telemanipulator, a power tool or a robot. Even a valve leaking compressed air could render such a system unsafe. This significantly constrains the application of teleoperator technology.

2.4 NETWORKED TELEROBOTS

While early servomanipulators relied on analog communications, digital communications are preferable over all but the shortest links. The use of digital controllers and digital communication links naturally led to experiments in increasing the separation between master and slave. Roughly speaking, as the distance increases, the stability and latency of such a system are increasingly dominated by the properties of the communication link.

Radio communication links are subject to interference from a wide variety of natural and manmade sources, including lightning, solar activity, vehicle ignition systems, multipath effects, not to mention collisions that occur when multiple radios try to transmit simultaneously on the same channel. Wired and fiber optic communication links are less subject to interference, but are much more expensive, requiring the installation of cable which is typically buried underground. Further, wired and fiber optic links aren't particularly useful for spacecraft and submarine communications. Today, communication links are typically not dedicated point-to-point links between isolated systems, but share the infrastructure of the network of networks better known as the Internet [33].

The Transmission Control Protocol/Internet Protocol (TCP/IP) is responsible for the delivery of the majority of data over the Internet. Data is broken up into small pieces or "packets" that may travel any number of ways from their source to destination, as the shortest path is constantly changing. As such, packets arrive at their destination in no particular order. Routers along the way may even discard some packets; an Automatic Repeat reQuest (ARQ) is generated after these lost packets are deemed missing. Under heavy network loads, the ARQ messages may even be lost. Thus, no guarantees can be made about how long it might take before a transmission is successfully received, which means that no guarantees can be made about the stability of a teleoperator under pure feedback control over the Internet [18, Ch. 32].

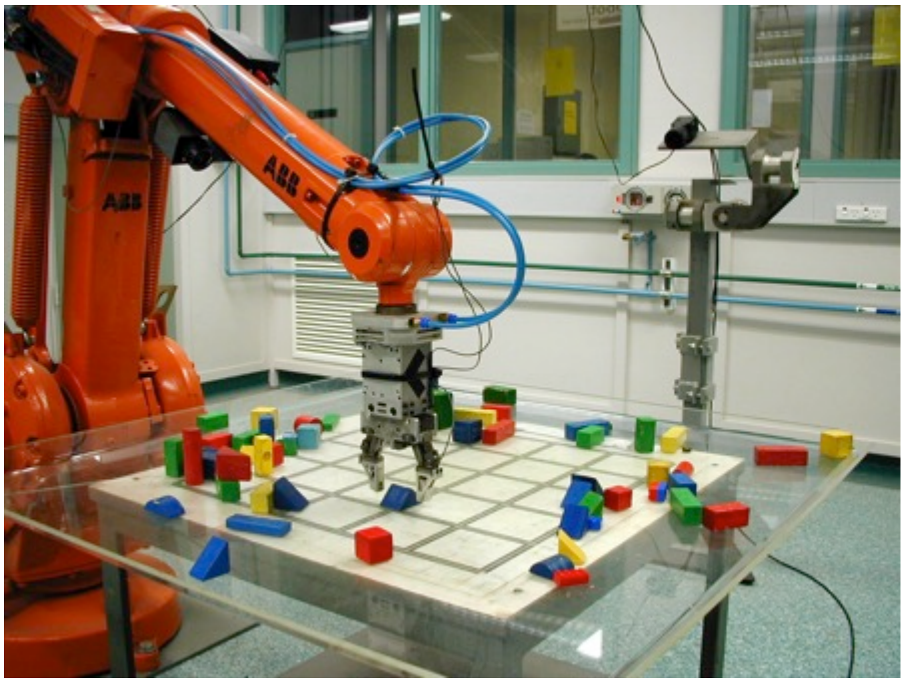

Figure 5: UWA Telerobot, the First Industrial Manipulator on the Internet [34]

As a result, practically all networked telerobots are under supervisory control, defined by Sheridan as "one or more human operators… intermittently programming and continually receiving information from a computer that itself closes an autonomous control loop through artificial effectors and sensors to the controlled process or task environment" [35]. In other words, a local computer ensures safe operation and stability while responding to remote commands to the greatest extent possible. This model is particularly relevant to safe and high reliability operations not due to any desire to connect telerobots to the Internet, but because Ethernet networks are ubiquitous, and because a loss of communications will inevitably occur. In fact, between the lack of security built in to robotic systems, the likelihood of network penetration, and the consequences of kinetic cyberattacks, it would be advisable to reconsider any decision to connect a networked telerobot to the Internet for the foreseeable future.

2.5 HUMAN FACTORS

The discipline of human factors and ergonomics is concerned with the design of products and systems that match the physical and cognitive capabilities of the people who use them. Human operators are central to teleoperation, and so a rational teleoperator design ought to begin not with a study of the tasks to be performed, but with a study of the characteristic physiological and cognitive abilities, strengths and weaknesses of the operators themselves.

Human physiological limits determine the work that can be done, and the rate at which work can be done, without injury. Just as in glovebox work, ergonomic injuries are not uncommon among operators, mostly due to repetitive strain injury [36]. These injuries are costly both in terms of lost time and medical rehabilitation, and are demoralizing to workers. Employers have a moral obligation, and in most jurisdictions a legal obligation, to provide workers with safe work and a safe workplace. While it's possible to address these issues by mandating worker adaptation (e.g. by prescribing strengthening exercises), or work adaptation (e.g. by mandating daily use limits), these constrain productivity and can give rise to resentment and even ridicule. The most efficient way to address issues of occupational health and safety is not by treating process as dogma and requiring workers to adapt to their tools, but by coming up with new tools and processes that accomplish the same goals (not necessarily via the same set of tasks) without the possibility of operator injury.

2.6 PERCEPTION AND PLANNING

Human operators perform tasks under feedback control of the musculature by the nervous system. The degree to which feedback is required is a function of the difficulty/complexity of the task and the operator's level of expertise. Indeed, in idiomatic English we often say that a simple task can be done "with your eyes closed", and someone of great skill can perform a task "with their eyes closed", i.e. in the absence of visual feedback. More often than not this isn't just a figure of speech, but is literally correct. We expect that an expert will perform familiar tasks with much greater accuracy and at greater speed than a neophyte, who must proceed more slowly and carefully, frequently checking their progress. In engineering terms, we observe that as familiarity develops, and a task becomes "second nature", there is a shift from perception-centered and reactive control to planning-centered and proactive control, i.e. from pure feedback control to accurate feedforward control with minimal corrective feedback.

Perception and planning are customarily performed in the brain, which communicates with its sensors and effectors via two sets of nerves. The efferent nerves extend from the spinal cord and control the muscles (effectors), and the afferent nerves return information from several types of specialized mechanoreceptors (sensors) to the spinal cord and brain by a separate system. While the loss of efferent nervous function results in paralysis, there are rare cases where individuals lose afferent nervous function but retain efferent nervous function. These deafferented individuals lose their sense of touch, but are still able to control their effectors, albeit with vastly reduced dexterity.

One such deafferented individual, G.O., was studied in [37], and found that as a result of his condition "his hands were relatively useless to him in daily life. He was unable to grasp a pen and write, to fasten his shirt buttons or to hold a cup in one hand. Part of his difficulty lay in the absence of any automatic reflex correction in his voluntary movements, and also to an inability to sustain constant levels of muscle contraction without visual feedback over periods of more than one or two seconds. He was also unable to maintain long sequences of simple motor programmes without vision." This description is highly reminiscent of the challenges faced by operators of unilateral telemanipulators, who must work without haptic feedback.

At first glance, it might seem that this implies the sense of touch is critical to dexterous manipulation, however an equally valid interpretation is that G.O.'s difficulty stemmed not from his inability to perform closed-loop feedback control, but his inability to perform open-loop control of his efferent nerves without saturating his effectors. In other words, G.O.'s difficulty arose from his inability to issue fine-grained commands to his muscles more than from his inability to respond to sensorimotor feedback. The fact that he was capable of any activity shows that force feedback is not entirely necessary. This point is corroborated by the two million surgeries that have been successfully performed using the *da Vinci* robotic surgery system without force feedback from the surgical instruments [38]. Surgeons rely instead on visual feedback, their proprioception, and their understanding of the mechanical properties of various kinds of tissue. In other words, they rely on a combination of situational awareness and experience. The same can be said of drone pilots, who command aircraft on the other side of the world using only visual feedback and limited instrumentation. All that these surgeons and pilots have at their disposal that G.O. did not is a certain delicacy.

2.7 Sensory Bandwidth and Complexity

Much effort has been placed into providing operators with "transparent" controls, i.e. accurate and timely force feedback. While humans can perceive tactile vibrations at up to 1 kHz, even 2 kHz turns out to be inadequate bandwidth to accurately represent interactions with stiff systems [39], although the energy component of force feedback is effectively limited to frequencies below 30 Hz [31]. To achieve stable and realistic haptic rendering, sampling must be performed at or above the Nyquist rate[1], thus we require a sensor bandwidth of more than 4 kHz, and correspondingly high performance actuators. At the time of writing, achieving complete transparency by mechanical means is not possible, nor will it be for the foreseeable future. Of course, tasks can be completed without transparency, using far less sensor and

[1] The Nyquist rate is the minimum sampling rate required to avoid aliasing, i.e. to be able to fully reconstruct a signal from samples, and is defined as twice the bandwidth of the signal. While it's possible to reconstruct a signal with fewer samples using compressed sensing techniques, they're beyond the scope of this work.

actuator bandwidth, but the cost is placing additional cognitive and physical load on the operator, who has to deal with phenomena such as "buzzing" and "spongy" or sluggish controls that confuse the senses, impede accuracy, and induce fatigue.

Feedback need not necessarily be haptic. Even under ideal conditions, the ordinary human senses are poor at resolving phenomena at unfamiliar scales, i.e. the very large/close/fast and the very small/distant/slow. As such, instruments of various kinds have been used for more than two thousand years to extend the range and sensitivity of the senses. For the most part, instruments comprise a sensor and visual indicator, though today it's just as likely that a sensor has an electronic rather than a mechanical/visual interface. However, the best way of communicating sensor data to operators is usually by means of visual feedback.

While visual feedback may seem more complex than haptic feedback, and require more processing, human operators respond just as quickly to visual as to haptic stimuli. In the case of visual stimuli, reaction time is improved when stimuli are present in the central (as opposed to peripheral) vision [40], just as reaction time is improved when touching the fingertips rather than the arm (due to the greater number of nerves in the fingertips, as well as to the greater number of neurons in the brain devoted to processing their signals). It's also been shown that reaction time to light and sound (to which we respond most quickly) is equivalent if the intensity of the visual stimulus is sufficiently high. Two psychophysical laws help us understand the importance of these results. The Weber-Fechner law shows that reaction time is proportional to the logarithm of the intensity of stimulus [41], and Hick's law shows that reaction time is proportional to the logarithm of the number of (similar) choices [42]. In other words, there's a logarithmic relationship between the complexity of/difficulty in discerning stimuli and the time it takes to respond to them.

Additionally, Fitts showed in [43] a logarithmic relationship between the difficulty/complexity of a task and the time to complete it, leading to what would come to be known as Fitts' Law. Sheridan and Ferrell extended Fitts' work in [32] by experimenting with a "minimal" two degree of freedom (DOF) manipulator plus grasp in the presence of time delay, and demonstrated a similar logarithmic relationship between time delay and task completion time.

Faster	Slower
Moderate Arousal	**Low/High Arousal**
Young	**Old**
Moderate Intensity	**Low/Painful Intensity**
High Sensitivity/Low Spurious	**Low Sensitivity/High Spurious**
Slight Anxiety	**High Anxiety/Carefree**
Central Vision	**Peripheral Vision**
No Distractions	**Distractions**
Warned in Advance	**Surprised**
Well Rested	**Fatigued**
Trained/Experienced	**Untrained/Inexperienced**
Recently Practiced	**Rusty**

Table 1: Factors Affecting Reaction Time [44]

Synthesizing these results, we have that complex tasks take longer to execute, and complex/poorly discernible stimuli take longer to respond to. These influence one another as the operator closes the loop, as an operator executing more complex tasks will spend more time perceiving, processing and responding to feedback, which in turn adds time delay to each task, which in turn means the operator will spend more perceiving, processing and responding to feedback, etc. and so we expect and observe that task completion times inflate exponentially with complexity.

We conclude that adding complexity to a task (where "complexity" is loosely defined as that which increases the physical or cognitive load on the operator) can be expected to cause an exponential increase in task completion time. This complexity may be either direct, as in the physical difficulty and number of operations required to execute commands, or indirect, as in a decrease in the quality, quantity or timeliness of feedback leading to slower reaction times and therefore a slower rate of work.

However, simply reducing complexity is not necessarily a means to boost productivity. Reconfiguring work to fit the assembly line model, where workers perform simple and repetitive tasks, reduces complexity but also reduces flexibility and increases the likelihood of ergonomic injury. To reduce complexity while retaining flexibility we must look beyond the limits of direct control, toward autonomy.

2.8 COMMAND RATE AND ABSTRACTION

The rate at which work can be performed is obviously proportional to the rate at which operators can issue commands. A higher command rate/more bandwidth means operators can issue more commands per unit time, or more complex commands in equal time. What Goertz referred to as the "natural movements" of the master tongs in a master-slave manipulator means that operators can control all six axes simultaneously. By comparison to the Goertz's early unilateral manipulators and many of today's EOD robots, which are controlled by moving axes in joint space one at a time, six-axis controls lead to a much greater than six-fold increase in operator productivity. A similar (but not quite as dramatic) boost in productivity is also seen when comparing the rate of work achievable with a six-axis master to that achievable with e.g. dual joystick controls. The reason for the nonlinear relationship is that working in any space other than Cartesian space dramatically increases operator cognitive load, as the mapping to and from the new task space and Cartesian space is unfamiliar and often unpredictable.

The nature of neural networks and the nature of learning are such that we start with the familiar when learning to interact with the unfamiliar. For example, when learning a foreign language, beginners habitually translate everything to and from their native tongue. However, this bidirectional translation is extremely hard work, and significantly hampers learning as the student spends most of their brainpower translating instead of learning. Similarly, when operators take a goal (i.e. motion to a given position and orientation) they plan a series of motions, mapping each part of the motion to the commands that they must execute, and then they execute those commands in order to provide the appropriate inputs to the interface and controller, all the while ensuring that they don't violate any system constraints and applying corrective feedback. Operators that must spend most of their mental energy translating these Cartesian motion plans into joint space will find it more difficult and time consuming to perform work, as well as more difficult and time consuming to master the controls. Over time, an experienced operator may learn to bypass at least some planning in Cartesian space and work directly in joint space, or at least take shortcuts based on their intuition. However, building this degree of efficiency requires thousands of hours of training and/or low productivity operations, which are clearly not desirable.

This cognitive mapping (from Cartesian to joint space) is an example of what we call a **layer of abstraction** through which all information must pass. Layers of abstraction are mappings; while it's possible to define filters and constraints as mappings, we list them separately for the sake of linguistic clarity.

Were it possible to produce a perfectly transparent set of controls, there would be no discernible layers of abstraction between the operator and teleoperator. Realistic controls, however, require the operator to contend with multiple constraints, filters and mappings at any given time. An operator must filter out (ignore) any controller "buzzing", accommodate damping by adjusting their own movements, plan the execution of the task, map the task to a number of Cartesian motion plans, map the Cartesian motion plans to the control inputs, perceive their environment (where sensory integration time is a function of the discernibility and intensity of the phenomena in question), process the sensory signals and determine how to respond to them, etc. If there's appreciable delay, the operator will also likely wait for confirmation between every single command. Of course, an operator with finite cognitive capacity can't juggle all of these at once, and so must continually switch their attention between planning and perception; as working memory is finite, this requires time spent storing and retrieving states, just as in computing, where this is referred to as **context switching**. The more layers of abstraction, the more complex the system, the more time the operator spends context switching, and the less time the operator spends performing useful work.

Primate studies have shown that the motor cortex encodes primitive motion plans which encompass multiple muscle groups, e.g. chewing, reaching, hand-to-mouth (feeding), etc. [45]. Thus, we conclude that motion planning is not an unconstrained search through configuration space, but a blending of primitives. To accomplish this, we use essentially the *same* parts of the brain to plan our motions as we do to execute them. Given a goal, an operator will generate a task plan, but must then decide how to specify those commands to the teleoperator, then create additional sub-plans to perform the motions that execute the commands. We conclude that even a single layer of abstraction will reduce operator productivity as operators switch back and forth between monitoring, execution, planning and replanning of tasks at different levels of abstraction. Training can be effective, but a simpler and cheaper alternative is to eliminate the abstraction altogether. Each human brain is highly adapted to use the set of arms and hands to which it's uniquely adapted. In the absence of an inexpensive, robust and safe brain-machine interface, the ideal human-robot interface is one that exploits the innate human capacity for mimicry and mirroring.

We conclude that the best interface is completely transparent to the operator, but not in the conventional sense. The traditional view is that the operator would ordinarily perform work with their hands, which can be best achieved by means of transparent telepresence. We consider that reducing the load on the operator is always desirable, not for the purposes of making the operator's job easier, but for the purposes of allowing them to perform more useful work in a given time, and for this purpose delegation (i.e. autonomy) is better than feedback.

2.9 User Interface Design

Numerous volumes have been published on the subject of user interface design, across the disciplines of psychology, sociology, anthropology, computer science and design. With few exceptions, a researcher (the designer, or possibly an ethnographer) observes the work to be completed. The designer then develops an interface from a palette of existing user interface elements, and devotes the majority of their time to user feedback and observation cycles. If all goes to plan, the end result is a usable, user-centered interface.

It is a little odd that this methodology rarely results in an excellent user interface. A small number of highly experienced designers have identified this as a problem and offered theories as to why this might be. Alan Cooper believes that "… it is our almost universal willingness to accept bad interaction as an unavoidable cost" of technology [46]. In other words, we are so used to user interfaces being awful to work with that it doesn't even register as a possibility that we might be able to do better.

In [46], Cooper calls for what he calls **goal-directed design**, citing the example of a hypothetical office manager who wants her office to run smoothly. She has a wireless router, and a VPN, and offsite backup, and a Microsoft per-seat license. The logical thing to do is to produce user interfaces for each task that allow her to perform required maintenance on all of these systems as quickly and easily as possible. However, the office manager's goal is not to perform routine maintenance. In fact, she doesn't want to do any maintenance. She wants her office to run smoothly, and to know when there's an issue, so she can call for technical support. Providing her with interfaces that allow her to perform routine maintenance is decidedly not logical.

This kind of **task-directed design** is pervasive and usually counterproductive, and although the user interface design community has rallied around the concept of **user-centered design** in recent years, not a lot has changed, since the term user-centered is a slippery one. No mention is made of who the user is, or of what the user is supposed to accomplish. Cooper suggests that designers go out of their way to create personas and use cases, and to purposefully create niche rather than mass-market products. By designing for one very specific person, with the specific goal they have in mind, there are few if any compromises, and the end result is a great product. Despite this sage advice, the vast majority of user interfaces continue to cater to everyone and serve no one. One notable exception to this is Dropbox, which started in an already saturated market, with a product whose main "innovation" was to do one thing extremely well.

2.10 USER INTERFACE TYPES

In computing, there are presently four user interface paradigms, and each has strengths and weaknesses for particular types of users performing particular tasks.

- Command language
- Tactile manipulation (the desktop metaphor)
- Menus and forms
- Natural User Interfaces (NUIs)

Expert users (who have a broad understanding of the capabilities of computer systems and a correspondingly large vocabulary of commands) tend to favor command language interfaces, as they are flexible, extensible, and allow for task completion with high efficiency given prior knowledge of the process. Conversely, beginners are often more productive using menus and forms, which offer users a limited palette of options, organized around the requirements of the process, presented in an appropriate context, often with explanations.

For example, completing a simple task such as counting the words in a document is not possible in a menu driven user interface if a word counting command has not been explicitly programmed into the user interface. On the command line, however, this can be accomplished in a number of different ways. One might use a program built for this purpose, or connect together a number of smaller, more specialized programs to achieve the same result, e.g. by turning the document into an ordered list of unique words, and then passing it on to another program that counts the number of lines in the list.

Natural User Interfaces (NUIs) are designed to be effectively invisible to the operator, with the goal of creating "an experience… that ultimately leads to the user feeling like the pitcher atop the mound: completely comfortable, expert, and masterful—a virtuoso of the user experience… from the very beginning, for complete novices, and to carry this feeling through as the users become experts… at minimal cost in learning time and effort." [5] The classic example of the NUI is the touch/multi-touch interface used in smartphones and tablets, where the input device is a fingertip, and most UI elements are controlled with touches and swipes. Designers learned from their work on the earliest PDAs that a handwriting-based interface requiring a stylus was insufficiently robust (in the case of the Apple Newton) or overly complex (in the case of the Palm Pilot). Thus a broad gestural vocabulary was eschewed in favor of a contextual on-screen keyboard when complex information needs to be entered.

NUIs come in many forms, and no doubt many more have yet to be invented. No matter the context, it must be remembered that NUIs are just a tool in the toolbox to be adopted when it makes sense to do so, and that "the underlying driver is the reduction in time and effort that users incur in adopting new ways of interacting with machines." [5] For example, gestures and natural language interfaces are useful when issuing simple commands but relatively useless when trying to record large quantities of complex data.

2.11 THREE DIMENSIONAL COMPUTING

Computing and its metaphors are firmly stuck in two dimensions. By contrast, we live in a space which we can describe (in a classical, non-relativistic way) as $\mathbb{R}^3 \times SO(3)$. Achieving motion along six degrees of freedom requires at least six axes [47], and issuing these commands using repurposed human-computer interfaces designed for very different tasks is very challenging.

Over the past several decades almost every imaginable interface has been used to provide commands to robots: hardwired consoles covered in buttons and switches that run on relay logic, pushbutton keyboards, virtual (projected) keyboards, roll/pitch joysticks with numerous buttons, roll/pitch/yaw joysticks with numerous buttons, traditional (2D) mice with two or three buttons that used a ball and two rotary encoders, traditional (2D) mice with two buttons and a button/scroll wheel that use high resolution optical sensors and digital image correlation algorithms, 3D mice, 6D mice, gyroscopes, accelerometers, magnetometers, instrumented gloves, passive gloves with optically tracked fiducial markers, fingertip tracking from depth imagery, gestures recognized as skeletal motions from depth imagery, and so on.

The majority of these input devices operate in only two dimensions, and require multiple operations in order to issue a command in $\mathbb{R}^3 \times SO(3)$. Of those few that operate in more than two dimensions, there has historically been a tradeoff between cost and accuracy, workspace volume, degree of constrained motion, portability, comfort and usability. Systems such as VICON provide high sensing rates and accuracy but are expensive and rely on calibrated fixed cameras. "Prosumer" grade 3D pointing devices such as the 3Dconnexion SpaceNavigator are inexpensive but can't be used for absolute positioning. The best designs can be found in consumer grade 3D pointing devices designed for playing video games, as these peripherals must be suitable for mass production, relatively inexpensive, physically robust and ergonomically sound due to the fact that users often play games for many hours at a time without taking breaks, which repetitive strain injury is not uncommon. The Nintendo Wiimote is inexpensive, intuitive, and quite accurate. However, it communicates with the Bluetooth wireless protocol, making it unsuitable for use in a nuclear facility. Like any instrumented glove or wireless hand-held peripheral, it can also be inconvenient, as it's almost impossible to do anything else without first taking off the wrist strap.

2.12 SITUATIONAL AWARENESS

Unilateral joint space control is still used today for many teleoperated mobile manipulators (e.g. the Telerob tEODor, Foster-Miller Talon, Allen-Vanguard Digital Vanguard, etc.) and other Explosive Ordnance Disposal (EOD) robots. Professional operators report that the primary difficulty in controlling these robots in the field is the lack of situational awareness. Robots are fitted with a number of cameras and a microphone, however the operator must simultaneously navigate through unfamiliar territory, drive the robot, scan for threats, interact with objects in the environment via the joint-space controlled manipulator, maintain a mental model of the configuration of the robot, etc. all via relatively poor quality visual feedback. As a result, situational awareness is quite poor, and the cognitive load on the operator is extremely high. It's highly desirable to automate as many of these tasks as possible, however there is a strong desire to keep EOD robots as brutally simple as possible simply because they tend to be blown up.

Computers	Humans
Fast	Slow
Error free	Error prone
Deterministic	Irrational
Apathetic	Emotional
Literal	Inferential
Sequential	Random
Predictable	Unpredictable
Amoral	Ethical
Stupid	Intelligent

Table 2: Strengths and Weaknesses of Humans and Computers [46]

The glovebox environment similarly presents great challenges to the development of appropriate sensory feedback. Cameras are generally unsuitable for use in high radiation fields due to the spurious activation of pixel detectors and the introduction of opaque defects into lenses by ionizing radiation. Placing cameras on a glovebox teleoperator with the goal of being close to the task unfortunately exposes them to high levels of radiation, as intensity is proportional to the inverse-squared distance. Cameras used inside the glovebox must therefore be hardened, and can be expected to have relatively short service lives. Of course, it's likely preferable to install additional, more complex cameras that sit outside the glovebox and peer through existing windows, however gloveboxes tend to be very crowded, and visual occlusion can be expected to be a significant problem. Borescope or flexible fiberscope ports to the exterior of the glovebox would expose relatively inexpensive and easily replaceable optical elements to radiation, and would allow operators to change the visual magnification and orientation of their point of view to achieve the highest intensity of visual stimulus for the task at hand in order to make the task as easy to see as possible.

2.13 THE PROBLEM OF TELEOPERATION IS PRODUCTIVITY

The goal of teleoperation is to perform tasks that would otherwise be done with the hands. The goal of pure teleoperation is to provide a transparent experience to the operator, which incidentally can't offer any performance gain beyond the rate of doing work with the hands. By contrast, pure autonomy, supervisory control and shared control offer the possibility of increasing operator productivity beyond the rate of doing work with the hands. Task-focused design, which has been extensively employed in the past, has led to better teleoperators and tools. However, recognizing that the true goal of teleoperation is operator productivity, health and safety, **our main goal is to design a teleoperation interface that maximizes operator productivity, health and safety**.

2.14 SUMMARY

The goal of teleoperation is to perform tasks that would otherwise be done with the hands. The first teleoperators, mechanical pantographs, offered high mechanical bandwidth but poor payload capacity. Servomanipulators offered significantly higher payload capacities but proved difficult to control, and even today's technology is insufficient to provide operators with transparent controls and a sense of presence. A historical emphasis on developing ground-up pure automation solutions rather than sharing control between operators and automated systems has met with limited success, and a tendency toward task-focused design has improved teleoperator technology only incrementally.

The selection of appropriate instrumentation and provision of appropriate feedback requires consideration of many factors, especially human factors. Psychophysical considerations show the desirability of minimizing physical and cognitive loads on the operator, as well as lag. We seek to do so by minimizing context switching, reducing or eliminating layers of abstraction by favoring autonomy over feedback.

Our intent is to leverage the benefits of unilateral and bilateral teleoperation, freeing up the operator to "concentrate the greater portion of his mental attention on the work that is to be done rather than on how to control the manipulator" [14] by developing a new interface for user-centered robotic teleoperation that maximizes operator productivity, health and safety.

Chapter 3: Techniques and Technologies

The purpose of this work is to design a low cost, robust, responsive, efficient, user-centered interface for robotic teleoperation. To do so while avoiding many of the issues identified in Chapter 2, a significant number of technologies (some newly available) and techniques from the literature were evaluated. This section will describe the techniques and technologies on which this work depends.

- RGB-D Cameras: a new class of low cost integrated RGB and depth cameras have become available in the last few years and are used to sense the operator
- Quaternions: the kinematic abstraction of choice due to its efficient and non-singular representation of orientations
- Hand Tracking Software: designed for gestural user interfaces, this software uses data from an RGB-D camera to track the position and orientation of various parts of an operator's hands, enabling touchless/hands-free control of computer systems
- Robot Operating System (ROS) and support packages: a meta-operating system and framework for efficient interprocess messaging, with a rich ecosystem of software packages for navigation, path planning, visualization, etc.
- Robot Hardware: the manipulators, grippers and sensors that comprise the Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) system
- Workspace Cameras: additional camera hardware employed to provide optimal visual feedback for differing task scales, including a low-latency, low-profile gripper-mounted video camera as well as a portable video microscope for particularly fine work
- Evaluating and Comparing Teleoperation Methods: a discussion of metrics that are applicable to comparison of teleoperation techniques.

3.1 RGB-D CAMERAS

The hand tracking software package used in this work (3Gear Systems' Nimble SDK) requires depth sensor data from an RGB-D[2] or time-of-flight camera. In this work, a PrimeSense Carmine 1.09 RGB-D camera was used [48]. PrimeSense cameras use a technique that the company calls *Light Coding* [48]. A near-IR laser in the camera is used to illuminate the scene with a predefined pattern of dots. On a perfectly flat surface, this pattern is reflected without distortion; in a complex 3D scene, the points converge or diverge depending on the orientation and curvature of the surfaces from which they're reflected. This warped reflection is observed with a standard CMOS sensor, and the point locations are calculated on an Application Specific Integrated Circuit (ASIC) that's built into the camera, which alleviates much of the load on the Universal Serial Bus (USB) as well as the host computer's central processing unit (CPU). It should be noted that consumer RGB-D cameras are only suitable for indoor applications, as sunlight has a considerable infrared component.

Early versions of 3Gear Systems' hand tracking SDK were designed to work with a pair of Microsoft Kinect RGB-D cameras via the OpenNI framework. The Kinect was based on PrimeSense's first generation reference design, and its low 320x240 depth image resolution and minimum sensor range of 80 cm necessitated the mounting of the cameras on a bulky frame that was both inconvenient to work around and also significantly constrained the workspace. When PrimeSense released its second-generation designs with 640x480 depth image resolution, with nearly an order of magnitude better depth resolution, 3Gear Systems immediately switched to a single camera setup using the PrimeSense Carmine. This proved less expensive, required only one USB port, eliminated the problem of combining images, and

[2] RGB-D = Red, Green, Blue and Depth channels

with a minimum sensor range of just 35 cm, the PrimeSense Carmine provided users with a larger workspace volume.

PRODUCTS SPECIFICATIONS

PROPERTY		CARMINE 1.08	CARMINE 1.09	UNIT
Operating Temperature		5 – 40	10 – 40	[°C]
Data Interface		USB 2.0 / 3.0	USB 2.0 / 3.0	
Operation Range		0.8 – 3.5	0.35 – 1.4	[m]
Field of View (Horizontal, Vertical, Diagonal)		57.5, 45, 69 (H,V,D)	57.5, 45, 69 (H,V,D)	[deg]
Depth Image Size		640 x 480 (VGA)	640 x 480 (VGA)	[pixel x pixel]
Spatial x/y Resolution (2-Sigma Values)	@2m	3.4		[mm]
	@0.5m		0.9	[mm]
Depth Resolution (2-Sigma Values)	@2m	1.2		[cm]
	@0.5m		0.1	[cm]
Maximal Frames-per-Second Rate		60	60	
Color Image CMOS	@ 30 FPS	640 x 480 (VGA)	640 x 480 (VGA)	[pixel x pixel]
Built-in Microphones		2	2	
Data Format		16	16	[bit]
External Digital Audio Inputs		4	4	[inputs]
Dimensions: Width x Height x Depth		18 x 2.5 x 3.5	18 x 2.5 x 3.5	[cm]
Power Supply		USB	USB	
Maximal Power Consumption		2.25	2.25	[Watt]

Figure 6: PrimeSense Carmine Specifications [48]

Apple bought PrimeSense in late 2013 and has since shut down all of PrimeSense's public activities, from hardware sales to the distribution of their open-source depth camera drivers and middleware. Currently it's still possible to buy Asus Xtion Pro cameras, which are more or less equivalent to the PrimeSense Carmine 1.08, with a minimum sensor distance of 80 cm, and so are not ideal for this work. At the time of writing, 3Gear Systems is working with pmdtec, a German company, that is developing miniature time-of-flight cameras that may prove useful in developing future interfaces for robotic teleoperation.

3.2 QUATERNIONS

The 3Gear Systems Nimble SDK makes use of quaternions in its pose messages. This section describes the basic properties and operations of quaternions and discusses their relevance to kinematics.

Specifying the orientation of a rigid body can be done in several ways. Euler angles (such as the roll-pitch-yaw system of aeronautics) are very common, as are rotation matrices. However, Euler angle representations have singularities that lead to problems such as distortions in path calculations and gimbal lock in physical systems. Rotation matrices are inefficient in their representation, using nine parameters to encode what can be expressed in three; due to this redundancy their basis vectors tend to drift and so lose their orthonormality, which must be corrected by renormalization and reorthogonalization operations.

Hamilton developed quaternions in the mid-19th century as an extension to the complex numbers. They are succinctly expressed in the following equation:

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1$$

Quaternions are typically written as

$$q = (q_0, q_1, q_2, q_3) = q_0 + q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k} \ \in \mathbb{R}^4$$

where q_0 is a scalar, and $q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k}$ defines a vector. Thus, a quaternion uniquely expresses an angle about an axis of rotation, with no singularities, in only four parameters.

Quaternion multiplication is straightforward; given two quaternions p and q, we have

$$pq = p_0q_0 - p \cdot q + p_0q + q_0p + p \times q$$

All non-zero quaternions have a well-defined inverse,

$$q^{-1} = \frac{q^*}{\|q\|^2}$$

where the conjugate is defined as

$$q^* = q_0 - q_1\boldsymbol{i} - q_2\boldsymbol{j} - q_3\boldsymbol{k}$$

and the Euclidean norm is defined as

$$\|q\|^2 = qq^*$$

To rotate a vector, we pre- and post-multiply the vector by a unit quaternion and its inverse, with

$$u' = quq^{-1}$$

rotating u while preserving its norm.

Due to their lack of singularities, quaternions are also prized for their usefulness in interpolating between two orientations with the SLERP algorithm [49] as such interpolations can be numerically unstable in Euler angle and rotation matrix representations. This makes them particularly useful for path planning.

Since quaternions are non-commutative, care must be taken to preserve the order of the operands [50]. Similar care must be taken to ensure that the components of both quaternions are specified in the same order, as the scalar component can be placed at the beginning (w, x, y, z) or end (x, y, z, w) of a 4-tuple.

3.3 CAMERA CALIBRATION

In this work, the user's hand pose[3] is obtained from the Nimble SDK data as a seven-element vector comprising a Cartesian translation and a quaternion rotation. These are defined in relation to a coordinate space defined during camera calibration. The calibration process begins with placement of a black-and-white chessboard pattern on the work surface in front of the user, between the user and the computer screen, such that the center of the chessboard pattern is the origin of the workspace [51].

The calibration software takes an image of the scene with the camera, detects the corner points where the squares meet, and compares their positions and orientations to those of the selected calibration pattern. Using their relative positions and orientations in the image, the software uses a closed-form calculation to find an initial guess for the camera pose, which is refined using the Levenberg-Marquardt nonlinear

[3] A pose is a position and an orientation, and is typically comprised of three position elements and a quaternion.

optimization algorithm, yielding a maximum likelihood estimate of the camera matrix as well as the distortion coefficients that characterize the lens [52]. The workspace is then bounded from below by the work surface, from above by the minimum range of the depth sensor (0.35 m), and on the sides by the boundary of the (conic) field of view of the sensor.

Hand pose measurements, then, are given relative to this coordinate frame, in which "the x-axis points right; the y-axis points up; and the z-axis points away from the screen. Units are in millimeters and the origin is at the center of the checkerboard… used during calibration." The quaternion specifies a "rotational frame tracking the palm portion of the hand. The frame's x-axis is parallel with the vector extending from the hand to the forearm. The frame's y-axis points up from the back of the hand." [51]

Most manipulation tasks involve the operator's hand pointing away from the body, the robotic equivalent of which being, by convention, the z-axis pointing out of the wrist joint of an industrial manipulator. Thus orientations provided by the Nimble SDK in the hand tracking coordinate frame require rotation to a more prudent frame for teleoperation. Since our goal is for the end-effector to track the operator's hand, it was determined that transforming coordinates from the Nimble SDK's frame to the end-effector frame would be more suitable than the manipulator base frame for this work.

3.4 HAND TRACKING SOFTWARE

The hand tracking software used in this research is 3Gear Systems' Nimble SDK. As part of his PhD research at MIT, 3Gear Systems co-founder Robert Wang developed a glove-based optical hand tracking system in which users wore a brightly colored patchwork glove, which made the problem of optical pose estimation much simpler. Gloves, however, are problematic. In [53] Wang rightfully states that "[users] may be reluctant to put on a glove when switching from a 2D task such as menu navigation to a 3D task such as object assembly. Wearing a glove may also become uncomfortable during long work sessions." Wang doesn't mention that user reluctance to wear his gloves might have something to do with aesthetics.

Figure 7: Color Glove Used in Early Hand Tracking Research at MIT [17]

In the color glove work, an image captured by an RGB camera was denoised, segmented with a mixture-of-Gaussians color classifier, and then downscaled to produce a 40x40 pixel “tiny image” [17], [54]. This tiny image was compared to a database of 100,000 entries based on 18,000 finger configurations captured with a Cyberglove II motion capture system, which returned a blend of the *k*-nearest neighbor configurations. This pose estimate was further refined using inverse kinematics calculations to counteract differences between the downsampled estimate and the original image, and using temporal smoothing to reduce the effects of jitter [17]. In [12], Wang et al. changed to a silhouette-based technique, still using RGB cameras, but segmenting out the hands using background subtraction and skin-tone detection.

By contrast, the Nimble SDK makes no use of the RGB image, a fact easily proven by covering the RGB sensor, thus it performs its segmentation and pose estimation based on depth data. Details of the algorithms used in the latest version of the Nimble SDK remain proprietary, however the fact that the Nimble SDK generates databases for each hand indicate that the same basic technique of segmentation, downscaling and database lookup is still used. Having used multiple versions of the software over a period of more than two years, qualitatively speaking, it seems that recent versions have tightened constraints on the inverse kinematics to exclude more invalid hand poses.

A pose message is issued for every tracked frame, and consists of a basic message describing the position and orientation of the left and right wrists, followed by hand pose information for the left and right hands, then pose confidences for the left and right hands, and finally finger joint positions for the left and right hands [51]. A complete pose message contains 332 parameters.

Basic messages are composed of a message type string followed by the xyz position, the $xyzw$ quaternion, and an integer “click count” (where “clicks” are pinch gestures) for each of the left and right hands.

Hand pose information consists of a float denoting whether the pose is supplied with confidence, followed by $xyzw$ frame orientations and xyz positions for the root, wrist, and then proximal, medial and distal frames for each of the five digits. Finally, xyz positions are given for each of the tips of the five digits. Next, the hand pose confidences are floats denoting the likelihood of the “curled”, “ell”, “okay”, “pinch”, “pointing”, “relaxed open” and “spread” poses. These confidence scores range between 0.0 and 1.0, and sum to 1.0 for each hand. Finally, the positions of the hand’s 27 joints are given, from thumb carpometacarpal adduction/abduction to pinky proximal interphalangeal flexion/extension. These messages are prepared as UTF-8 encoded plain text and sent to clients over TCP/IP network sockets [51].

3.5 Robot Operating System (ROS)

ROS is an open-source meta-operating system for robots, “a flexible framework for writing robot software” [55]. At its core, ROS is a framework for messaging between various robot control programs. “A system built using ROS consists of a number of processes, potentially on a number of different hosts, connected at runtime in a peer-to-peer topology… The fundamental concepts of the ROS implementation are nodes, messages, topics, and services. Nodes are processes that perform computation… [and communicate asynchronously] with each other by passing messages… A node sends a [strictly typed] message by publishing it to a given topic, which is simply a string such as ‘odometry’ or ‘map.’” [56] Synchronous transactions are handled by services, which are “defined by a string name and a pair of strictly typed messages: one for the request and one for the response.” [56]

In just a few short years, ROS has become the *de facto* standard in the global robotics research community. Cutting edge research is disseminated both by traditional forms of publication but also by distribution of ROS packages and source code. For this and other reasons, the Nuclear Robotics Group joined the ROS-Industrial consortium [57], [58], and has recently completed the transition to developing its software in ROS.

However, ROS is not currently well equipped to perform real-time control of industrial robots. ROS development at Willow Garage occurred in tandem with development of the PR2 personal robot, which was designed to interact with quasi-static indoor environments. Though PR2 never became a commercial success, the early influence of its design requirements continues to echo throughout many ROS packages. Ubuntu, ROS' host operating system, lacks a real-time kernel, making deterministic-time processing impossible. ROS messaging is based on XML-RPC, which is designed more for simplicity than for efficiency. Nodes running on the same host routinely share huge volumes of uncompressed plain text data via TCP/IP network sockets when a variety of other mechanisms such as shared memory would be far more efficient. As such, the Nuclear Robotics Group (chiefly Brian O'Neil) developed a hybrid robotic control system that combines the low-level robustness of AX with the advanced capabilities and ease of integration of ROS.

3.6 MOVEIT!

Early versions of ROS were shipped with a package for robot arm motion planning (rather uninspiringly called the *arm_navigation* package) which, as features were added between releases, became increasingly unwieldy over time. Eventually, the developers of ROS at Willow Garage realized that in order for ROS to achieve its potential they needed to create a robot arm control package that catered to the needs of both novice and expert users, i.e. one that was both simple and powerful.

The *MoveIt!* ROS package exceeds the capabilities of its predecessor. Like *arm_navigation*, it leverages the capabilities of many ROS packages, wrapping them in a simple and powerful API and user interface for robot arm motion planning. By default, forward and inverse kinematics calculations are performed by the Open Robot Control Software (Orocos) Kinematics and Dynamics Library (KDL), which has been packaged as a plugin so that users can easily integrate customized solvers. Motion planning functionality is drawn from the Open Motion Planning Library (OMPL) [59]. Collision detection is provided by the Flexible Collision Library (FCL) [60]. The state of the robot and its workspace, referred to in *MoveIt!* as the "planning scene", is stored in a MongoDB database management system (DBMS), and is interactively displayed in the *Rviz* visualization tool, which uses the Open Source 3D Graphics Engine (Ogre3D) [61]. *MoveIt!* works with simulated or real robot hardware, communicating trajectory points and reading the robot joint states from robot drivers via ROS messages that conform to the requirements of the *moveit_controller_manager*.

3.7 IRAD (INDUSTRIAL RECONFIGURABLE ANTHROPOMORPHIC DUAL-ARM SYSTEM)

The core of IRAD is a pair of Yaskawa Motoman SIA5 7 degree of freedom (DOF) industrial robots equipped with ATI six-axis force-torque sensors and Robotiq 3-fingered adaptive robot gripper.

Figure 8: Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) system

The SIA5 design is based on that of the SIA10, and is notable for its offset elbow, which was intended to improve the performance of glovebox operations by mimicking the human tendency to perform work on a surface with elbows up, and to perform work above a surface with elbows resting on the surface. The SIA5's specifications are provided in Figure 9.

SIA5D SPECIFICATIONS		
Structure		Articulated
Mounting		Floor, Wall or Ceiling
Controlled Axes		7
Payload		5 kg (11 lb)
Vertical Reach		1,007 mm (39.6")
Horizontal Reach		559 mm (22")
Repeatability		±0.06 mm (±0.002")
Maximum Motion Range	S-Axis (Turning/Sweep)	±180°
	L-Axis (Lower Arm)	±110°
	E-Axis (Elbow)	±170°
	U-Axis (Upper Arm)	+115 / -90°
	R-Axis (Wrist Roll)	±180°
	B-Axis (Bend/Pitch/Yaw)	±110°
	T-Axis (Wrist Twist)	±180°
Maximum Speed	S-Axis	200°/s
	L-Axis	200°/s
	E-Axis	200°/s
	U-Axis	200°/s
	R-Axis	200°/s
	B-Axis	230°/s
	T-Axis	350°/s
Approximate Mass		30 kg (66.2 lb)
Power Rating		1 kVA
Allowable Moment	R-Axis	14.7 N • m
	B-Axis	14.7 N • m
	T-Axis	7.35 N • m
Allowable Moment of Inertia	R-Axis	0.45 kg • m^2
	B-Axis	0.45 kg • m^2
	T-Axis	0.11 kg • m^2
Protection Class	Standard	Not rated
	XP Version* (option)	N/A

* XP Version: Yaskawa Motoman's eXtra Protection package

Figure 9: Yaskawa Motoman SIA5 Specifications [62]

Although IRAD's SIA5s shipped with Motoman DX100 controllers, IRAD uses a pair of Agile Planet[4] AX Controllers. At the heart of the AX system is a computer that runs user control programs built using the AX API on the Windows CE Real-Time Operating System (RTOS). The control computer

[4] Agile Planet is a spinoff of the University of Texas at Austin's Robotics Research Group, and is now a subsidiary of the Motoman Robotics Division of Yaskawa America, Inc.

communicates with the robot servo drives via a dedicated programmable logic controller (PLC) for each robot using the EtherCAT industrial networking protocol [63]. This system is capable of 1000 Hz control rates using inexpensive commercial off-the-shelf (COTS) hardware.

Control programs for the Windows CE system are created using Agile Planet's Microsoft Application Development Kit (MS-ADK), which allows AX-based motion control programs to be built in C/C++ using Microsoft Visual Studio 2008. The AX control software is best described as middleware, allowing users to monitor the state of their robots, perform forward and inverse kinematics and dynamics calculations in real-time, and provide trajectory points to any suitable robot via the AX API.

In the earliest stages of this research, a proof-of-concept user interface for hands-free robotic teleoperation was developed using the AX and Win32 APIs on the Windows 7 operating system. Agile Planet provided Denavit-Hartenberg parameters and a collision model for a generic 6 DOF robot similar to the Motoman MH3, and the results of the AX's simulated dynamics were visualized in Roboworks. While it was possible to demonstrate that the method made it possible to perform hands-free robotic teleoperation, the system operated too slowly to be useful, as the AX motion planner was designed to come up with smooth, repeatable trajectories at a rate of no more than two per second.

3.8 METRICS FOR TELEOPERATION INTERFACES

The usability of an operator interface is typically measured in terms of its ease to learn, ease to remember, error rate, subjective satisfaction, and task completion time. These metrics are certainly applicable to interfaces where operators can only transition between a finite set of discrete states, such as a command line interface or a menu driven user interface. However, in spatial interactions with dynamic systems such as in teleoperation, the task completion time doesn't directly capture information about the complexity of the task or the cognitive load on the operator. Task completion time, error rate and contact forces will vary with complexity and cognitive load, but these are more suggestive than conclusive.

We propose that a useful measure of task complexity (from the point of view of the operator) and of the degree of operator context switching is the **command rate**, i.e. the number of operations performed per unit time. High task completion times traditionally indicate low efficiency. However, high task completion times with high command rates imply that the operator is spending substantial effort on trial-and-error i.e. planning, replanning, and associated context switching, and that additional training or instrumentation will boost operator productivity. High task completion times with low command rates could indicate that the interface is much slower than the operator, that the task is highly complex and the operator is spending excessive time planning, and so forth. Low task completion times with high command rates suggest that implementing additional automation and controls could boost operator productivity, as operator productivity will diminish due to the high rate of context switching caused by a high rate of planning/replanning. In most circumstances, low command rates will be preferable, suggesting streamlined operations and a high degree of automaticity when realized on non-trivial tasks.

3.9 SUMMARY

The development of a low cost, robust, responsive, efficient, user-centered interface for robotic teleoperation requires the selection and integration of appropriate sensors, control software and robot hardware.

The availability of inexpensive, accurate and robust depth image sensors has opened up new possibilities for the development of NUI-based software, among them the gestural user interface. Although largely designed for use in two-dimensional computing, gestural user interfaces are able to perform real time hand tracking, which is ideally suited for three-dimensional computing applications such as robotic teleoperation.

Both ROS and the ecosystem of packages built around ROS such as *MoveIt!* are freely available, widely used and supported by a large community of users and developers, offering an ideal development environment. Similarly, the COTS components, anthropomorphic dimensions and industrial accuracy and robustness of IRAD make it an ideal platform for the development of a robotic teleoperation system.

Finally, any new system must be tested. Existing usability metrics have been developed for use in two-dimensional computing based on the desktop paradigm, where operator inputs lead to discrete state transitions, and continuous operator feedback is atypical. Although conceptually simple, we propose that the command rate is an important parameter for characterizing the usability of three-dimensional and other computing interfaces.

Chapter 4: System Implementation

4.1 DESIGN REQUIREMENTS

As discussed in Chapter 2, the design of a human-robot interface for a robotic teleoperation system requires careful consideration, since many of its desirable characteristics are incompatible with each other. The list of requirements includes:

- Resources
  - Low capital costs
  - Low ongoing/maintenance costs
  - Long-term availability of necessary resources
- Capabilities
  - High productivity/low task completion time
  - Capable of working with heavy payloads
  - Capable of gross and fine motion
- Robustness
  - Robust to system failure
  - Robust to user error
- Human factors
  - Easy to learn/simple but powerful
  - Easy to use and ergonomically sound
  - Easy to remember how to use
  - Compatible with all likely operators (i.e. left/right hand, old/young operators, small/large hands, operators with disabilities, etc.)

4.2 SYSTEM IMPLEMENTATION

The robot platform used as the remote manipulator in the development of this system was IRAD (the Industrial Reconfigurable Anthropomorphic Dual-arm system [16], [62], [64]), a port-deployed glovebox robot platform under development by The University of Texas at Austin's Nuclear Robotics Group (NRG) for Los Alamos National Laboratory.

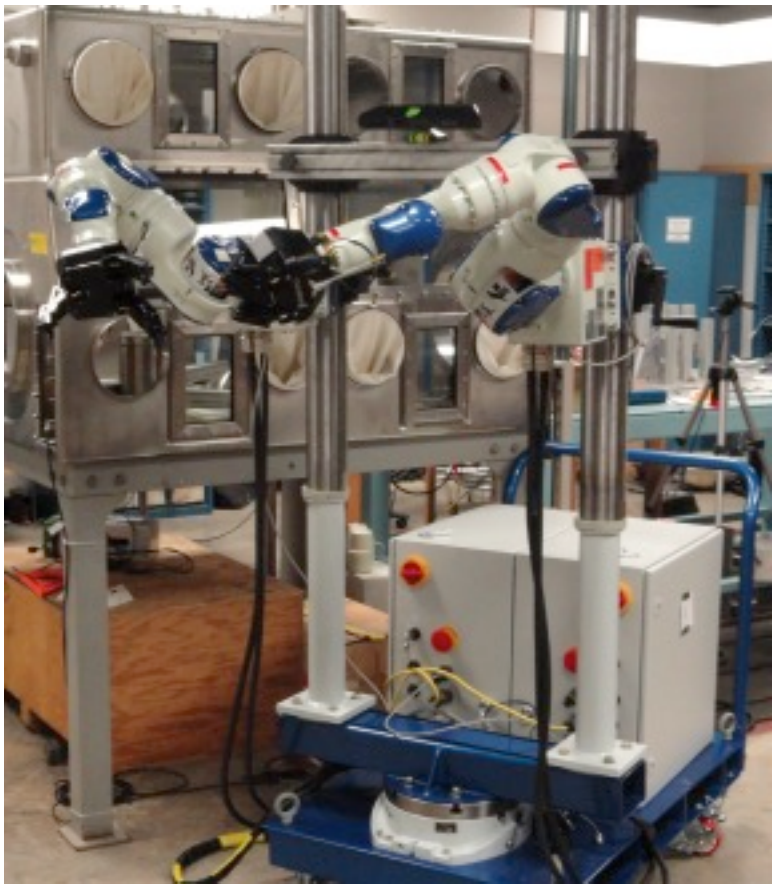

Figure 10: Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) system

IRAD comprises two Yaskawa Motoman SIA5 industrial manipulators, each with seven degrees of freedom (DOF) and a payload of five kilograms. The comparatively low payload of the SIA5 is a consequence of the roughly anthropomorphic dimensions, however these are necessary in order to deploy the IRAD system into existing glovebox glove ports. Further, as an industrial manipulator the SIA5 is designed for repeatability, and its design allows virtually zero mechanical compliance.

Fitted to the tool plate of each robot is a three-finger adaptive robot gripper (S-model) made by Robotiq, and are controlled by setting registers via Modbus over TCP/IP. While each digit can be controlled individually, the Robotiq gripper performs force closure, which greatly simplifies the grasping process.

An Allen Bradley PLC-based controller developed by Agile Planet controls each SIA5 robot. These controllers are capable of cycling up to 1000 Hz, and monitor and execute commands sent over an EtherCAT connection from a computer running the Windows CE real time operating system (RTOS). The computer system used by NRG uses the AT-Cewin software from Acontis Technologies which allows the user to concurrently run Windows 7 and Windows CE on the same computer system by dedicating one or more of the cores of a multicore CPU to the RTOS.

The RTOS runs a control program that makes extensive use of Agile Planet's MS-ADK libraries, which are based on the OSCAR software developed by UT Austin's Robotics Research Group. For the purposes of this research, the control program monitors the state of the robot and executes position commands it receives over a TCP/IP socket connection.

A block diagram overview of the IRAD system is provided in Figure 11.

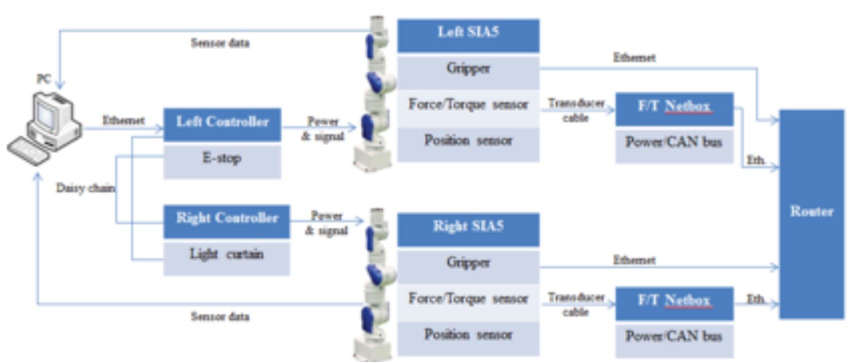

Figure 11: Overview of the Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) system [65]

The use of a robotic manipulator permits us to leverage existing technologies and software for inverse kinematics, motion planning, visualization of the state of the remote manipulator, and so forth.

This system uses unilateral operator control achieved by means of optical hand tracking. The operator relies on their own proprioception and visual feedback to maintain situational awareness, and the remote manipulator relies on a force-torque sensor, a world model and other sensors to avoid collisions and regulate contact forces. While torque models, force limiting and virtual compliance capabilities have been developed by Kyle Schroeder [66] and Andy Zelenak [67] of the Nuclear Robotics Group, the purpose of this work is to develop a teleoperation interface. As such, this work considers only the kinematic aspects of the problem, and was performed without use of the force-torque sensors, leaving dynamic considerations for future work.

4.3 HUMAN-ROBOT INTERFACE SOFTWARE

We require high reliability and system stability, and we desire a high degree of automaticity, i.e. complex tasks or tasks requiring the operator's constant attention should be delegated to the greatest extent possible. Thus, we choose position control over velocity control. We further require safe operation in the presence of unexpected operator motion, e.g. a sneeze. This can be addressed by requiring the operator to maintain closure of a permissive interlock (e.g. a dead-man switch) to permit motion, or to positively actuate a control input in order to effect motion. This system maintains continuous observation of the operator's hand pose but requires discrete motions to be initiated by a push-button control.

Position commands provided to the RTOS are generated on a computer system running the Ubuntu operating system and the Robot Operating System (ROS, a meta-operating system used extensively in robotics research). In particular, two ROS packages known as *MoveIt!* and ROS-Industrial were extensively used in this work to visualize and control the IRAD system.

To facilitate integration, ROS programs are organized as the nodes of a bidirectional graph that communicate via XML-RPC, allowing various hardware drivers and algorithms to share data in a simple and efficient way.

The user interface comprises the following nodes designed and implemented by the author:

- *constraints.py* – programmatically adds an object to the planning scene that acts as a ground plane for the purposes of collision-free motion planning
- *powermate.py* – processes signals from the Griffin Powermate device and publishes them to the /powermate and /powermate_button topics
- *spatial_interface_controller.py* – connects to the hand tracking software, decodes the hand position message, discards irrelevant data, publishes the wrist pose to the /hand_position topic and as a *tf* transform
- *robotiq_driver* – extends the functionality of the *robotiq_s_model_control* software by monitoring the gripper's status, calculating the positions of the digits and publishing them to the /joint_states topics
- *execute_motion.py* – listens to the /powermate_button topic for a button press, then plans and executes the arm motion
- *rviz_robotiq Rviz* plugin – extends the functionality of the *robotiq_s_model_control* software by providing a GUI interface to control the gripper from within *Rviz*
- *rviz_zoom Rviz* plugin – adjusts zoom level of the *Rviz* virtual camera according to the position of the Powermate so that the camera zooms in as the motions are scaled up/down

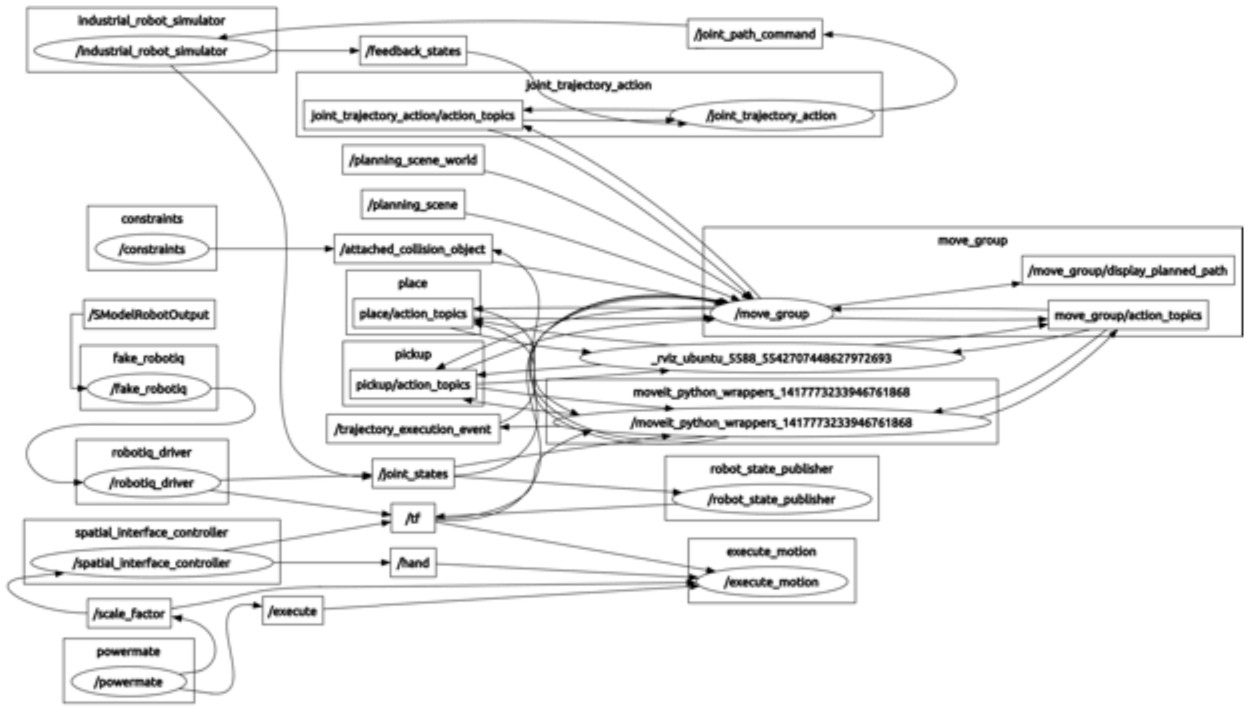

Figure 12: ROS Node Graph Describing High Level Human-Robot Interface Software

These interface with the following nodes produced by others:

- *robotiq_s_model_control/SModelTcpNode* (by Robotiq) – this node communicates with the Robotiq gripper, reading and writing its registers via Modbus/TCP.
- *MoveIt!*/ROS-Industrial
  - *Rviz* – visualizer and graphical user interface for robot planning and control
  - *industrial_robot_simulator* – when not connected to a robot, emulates joint motions by linearly interpolating between the start and end positions
  - *joint_trajectory_action* – sends trajectory points to the controller and monitors the progress of the robot
  - *robot_state_publisher* – publishes a complete *tf* transform tree for the robot by calculating the forward kinematics of the robot using /joint_states messages and the /robot_description parameter (which contains the URDF)

- *agile_planet_driver* – the *robot_state_node* and *joint_streaming_node* nodes, as well as the *AX_ROS* control program run on the Windows CE RTOS, were originally created by Brian O'Neil of the Nuclear Robotics Group, in order to provide a ROS-Industrial compliant interface between ROS and the Agile Planet AX controllers. These were substantially rewritten by the author in order to improve reliability, as well as to implement non-blocking network sockets

4.4 HAND TRACKING SERVER

The Ubuntu/ROS computer system, in turn, takes commands from a third computer system running the Nimble Software Development Kit (SDK) from 3Gear Systems, which uses the depth imagery from a PrimeSense Carmine RGB-D camera to model the state of a user's hands. The positions of the wrists as well as each joint of the hands are sent via 3Gear's proprietary protocol to a TCP/IP network socket.

4.5 ALGORITHMS

The software developed in this work can be grouped into three categories: driver nodes, operator interface nodes, and robot control nodes.

4.5.1 Driver Nodes

The *powermate.py* node, the *spatial_interface_controller.py* node and the *robotiq_driver* node as well as the *joint_trajectory_action* and *robot_state_publisher* nodes can be thought of as driver nodes, in the sense that they listen for/request data from a device interface/API and make it available to other ROS nodes by publishing it to a ROS topic.

The *spatial_interface_controller.py*, *robot_state_node* and *joint_streaming_node* nodes, and the *AX_ROS* control program additionally require non-blocking TCP/IP socket connections in order to pass ASCII/UTF-8 string data between programs running on different computers.

The structure of these nodes is described by the following pseudocode.

- *advertise* topics on which data will be published
- *subscribe* to topics which will be monitored
- *connect* to device
- *define callbacks* (to be called asynchronously)
    - *process* data e.g. from subscribed topics
- *while* ROS not shut down
    - *listen* for events
    - *process* events
    - *publish* data to ROS topic

In the case of the *powermate.py* node, device connection was made via the *evdev* interface, with states described by boolean or integer datatypes. All other nodes besides the *robotiq_driver* node (which communicated with a vendor-supplied driver) transmitted and received ASCII/UTF-8 string data. This data was processed using streams, iterators and Standard Template Library (STL) string functions in the C++ language, and by native string functions in the Python language.

Care must be taken when using buffered floating-point representations of strings. The AX_ROS control program driver contained a bug that caused the driver software to throw velocity violation exceptions, seemingly at random. Only after several weeks were spent rewriting substantial portions of the driver was it discovered that the root cause was the string parser implementation. Floating point numbers were separated by a single space, however if a floating point number extended beyond the string buffer, the digits up until the end of the string buffer were interpreted as the complete floating point number. When the next part of the string was read into the string buffer, any remaining digits were interpreted as a new floating point number; typically a very large one, since the decimal point was part of the previous string. While pushing raw data to a socket is highly efficient, there is something to be said for the use of markup languages, which could easily have prevented this bug and saved weeks of work.

4.5.2 Operator Interface Nodes

The *rviz_zoom* and *rviz_robotiq* plugins are not nodes in their own right, but are run as part of the *Rviz* node. *Rviz* uses the Qt framework, and these plugins are used to process operator interactions with customized user interface elements.

The plugin architecture prohibits the use of blocking operations, which cause the *Rviz* node to freeze or crash. Therefore, asynchronous operations must be buffered and handled periodically, or handled via callbacks.

Their structure is described by the following pseudocode:

- *initialize* (called as part of the *Rviz* initialization process)
    - *instantiate* queue (or other data structure)
- *update* (called periodically by setting a timing event)
    - *process* events
    - *publish* data to ROS topic
- *define callbacks* (to be called asynchronously)
    - *process* data e.g. from subscribed topics

4.5.3 Robot Control Nodes

The *constraints.py* and *execute_motion.py* nodes handled interactions with *MoveIt!* via its API, using components of the *moveit_commander* package.

The *constraints.py* node, built with extensibility in mind, was only used to create a plane in the collision model representing an artificial surface above the table on which the robot was mounted so that the robot could not collide with the steel tabletop. This surface is, in fact, not defined as a plane but as a thin box, so that fast-moving objects can't "pierce" the surface without a collision being registered by the collision model.

The *execute_motion.py* node is essentially a wrapper for *moveit_commander*, itself a Python wrapper for *MoveIt!* When an execute command is received on the /execute topic, the *execute_motion.py* node requests that *MoveIt!* plan and execute a motion.

The structure of robot control nodes is virtually identical to the driver nodes, with the exception of the device connection.

- *advertise* topics on which data will be published
- *subscribe* to topics which will be monitored
- *define callbacks* (to be called asynchronously)
    - *process* data e.g. from subscribed topics
- *while* ROS not shut down
    - *listen* for events
    - *process* events
    - *publish* data to ROS topic

4.6 SUMMARY

Compared to the multi-million dollar teleoperation systems used for nuclear materials handling at Oak Ridge and Le Havre, this complete system can be purchased for significantly less than $200,000. The use of widely used commercial hardware and open-source software components keeps capital and maintenance costs to a minimum and increases robustness, while ensuring the long-term availability of system components. IRAD's SIA5 manipulators are robust and inexpensive, and Joseph Hashem of the Nuclear Robotics Group found the SIA5's repeatability to be approximately 10 microns [68]. The SIA5 is therefore capable of gross (anthropomorphic scale) as well as highly accurate (sub-millimeter) motions.

The RTOS controller and high-bandwidth controllers ensure the system is responsive to operator inputs and cameras were selected and configured to provide low-latency visual feedback. Finally, the hands-free teleoperation interface has been designed from scratch with a variety of human factors in mind, in order to maximize operator productivity, health and safety, minimizing the impact of system failure and user error.

The system's low payload, while not strictly a weakness, does constrain its potential applications. Each of IRAD's SIA5 manipulators is rated to handle 5 kg, including the 2.4 kg gripper, however this is sufficient for many glovebox tasks, e.g. picking up and organizing hand tools. If a particular application requires a higher payload, IRAD can easily be converted to use almost any commercially available manipulator. It should also be noted that the rated payload is determined using very conservative assumptions to ensure that the system is able to execute motions at the rated maximum acceleration with the payload at full extension. This is clearly unnecessary for almost any conceivable operational scenario, and is most likely due to actuator power limits rather than structural limits. Thus, an area of future work may be to quantify the actual payload of the system within the operational task space.

We conclude that the proposed system meets the necessary requirements. A block diagram of the system design is shown in Figure 13.

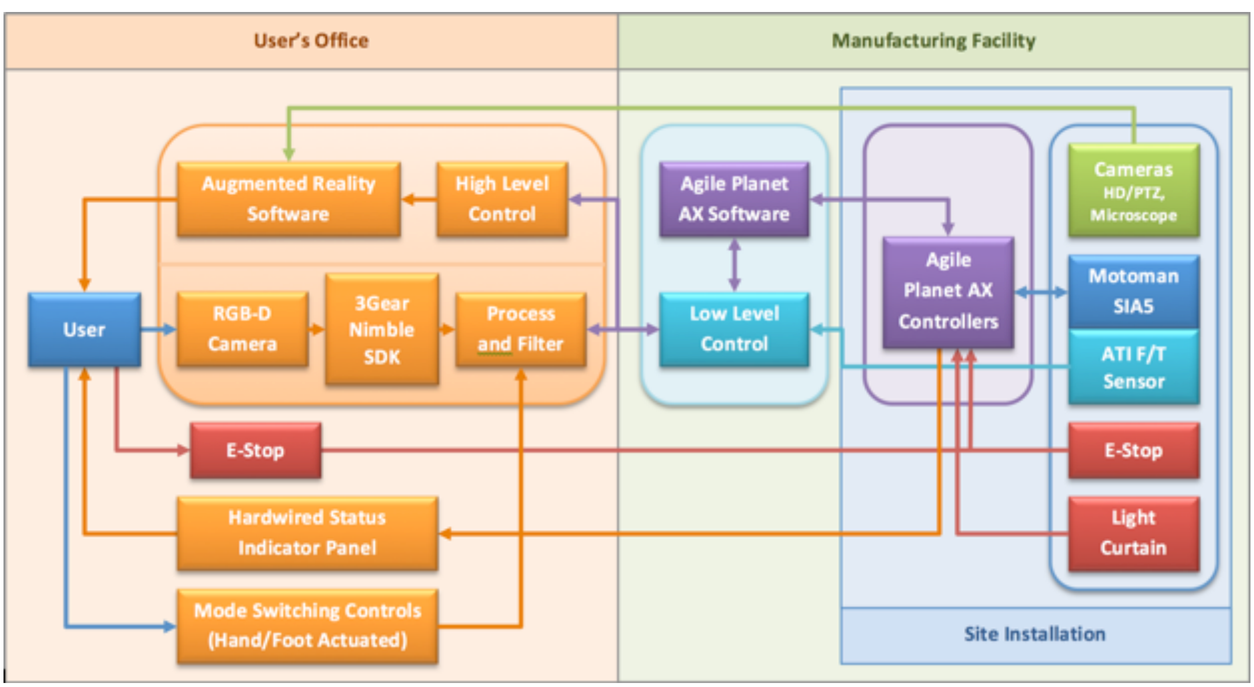

Figure 13: Block Diagram of the Proposed Teleoperation System

Chapter 5: Demonstrations and Usability Surveys

In order to quantitatively measure system performance, two sets of experiments were undertaken in which volunteers performed a series of teleoperation tasks requiring increasing levels of dexterity. A research proposal was submitted to University of Texas at Austin's Institutional Review Board (IRB) guidelines for human subject experimentation, which was approved subject to criteria of ethics training, voluntary informed consent, and privacy protection, under IRB exemption 2014-06-0026.

5.1 METHODOLOGY

5.1.1 Single Operator

The methodology and results of the first experimental campaign are repeated here, having been first published in [69]. A volunteer with some knowledge of robotics but who was initially unfamiliar with the task, robot and workspace was recruited. The task, threading a strand of 0.30 mm diameter all-purpose polyester sewing thread into the 0.7x6.5mm eye of a 2.5" 15 gauge (1.15mm OD) needle, was chosen in order to test the operator's ability to perform fine work with the IRAD system, and to compare the interface developed in this work to existing interfaces for teleoperation.

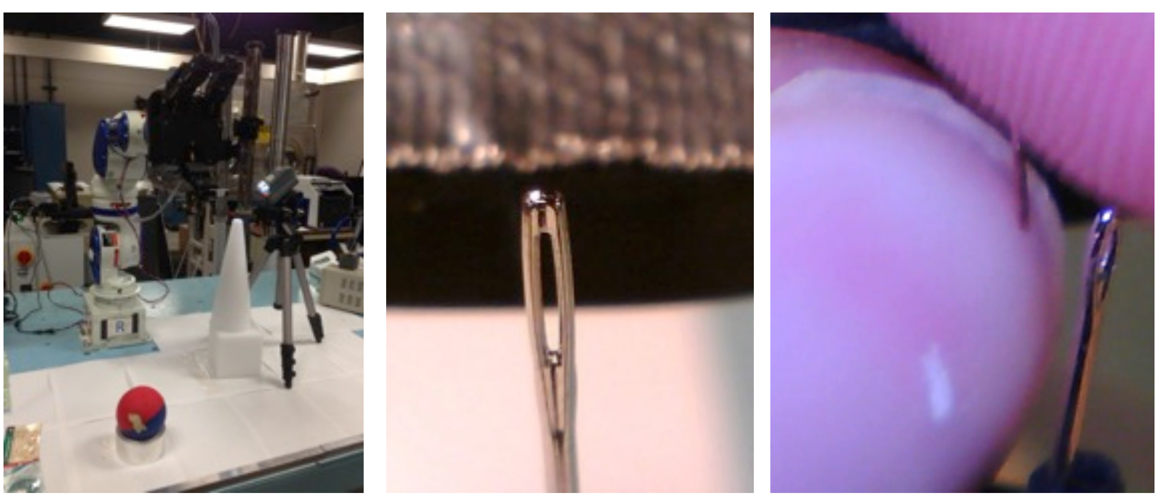

Figure 14: Single Operator Experiment - Setup, Large Needle, Small Needle with Copper Strand

The operator was provided with a brief explanation of how to operate the teleoperation system, given a chance to practice, and then asked to perform the task with each of the following teleoperation interfaces:

- Command Line
- GUI (interactive marker)
- Optical Hand Tracking

The operator repeated the task three times with each interface. Each attempt was timed, and the number of operations counted, so that the command rate (i.e. the number of operations per unit time) could be determined.

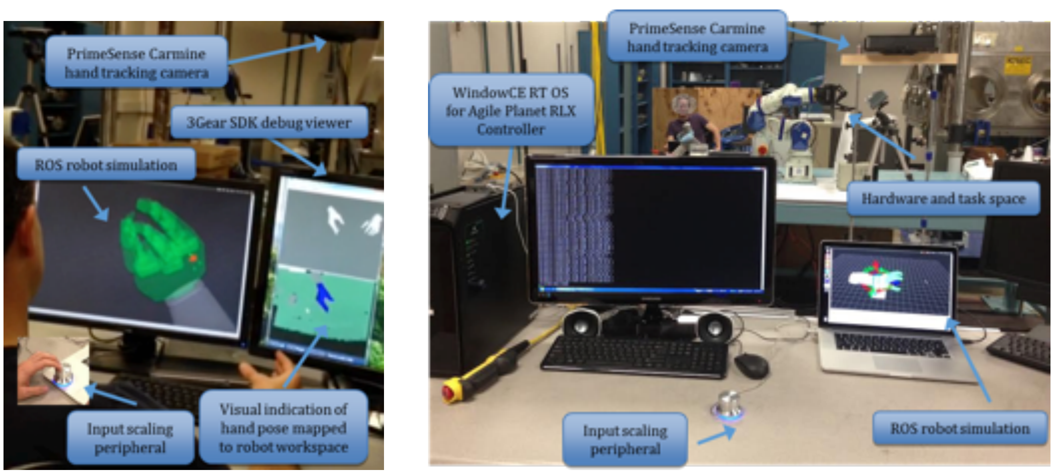

Figure 15: System User Interface

5.1.2 Multiple Operators

In the second experimental campaign, multiple volunteers with some knowledge of robotics but who were initially unfamiliar with the task, robot and workspace were recruited and asked to perform a set of tasks, with user surveys performed after the completion of each task. Each task was timed, and the number of operations counted, so that the command rate (i.e. the number of operations per unit time) could be calculated.

Operators were given a brief explanation of how to operate the teleoperation system, given a chance to practice, and then asked to perform each of the following tasks:

- Pick Up Ball: pick up a foam ball approximately 4” (10cm) in diameter
    - GUI (interactive marker)
    - Hands-free Teleoperation Interface
- Pick Up Can: pick up a can approximately 4” (10cm) in diameter
    - GUI (interactive marker)
    - Hands-free Teleoperation Interface
- Thread a Large Needle: maneuver a strand of 0.30 mm diameter all-purpose polyester sewing thread into the 0.7x6.5mm eye of a 2.5” 15 gauge (1.15mm OD) needle
    - GUI (interactive marker)
    - Hands-free Teleoperation Interface
- Thread a Small Needle: maneuver a strand of 0.30mm diameter copper wire into the 0.33x1.33mm eye of a size 55/6 sharp (0.53mm OD) needle
    - GUI (interactive marker)
    - Hands-free Teleoperation Interface

This set of tasks was chosen with multiple objectives in mind. While the first campaign validated the viability of the hands-free teleoperation interface, the purpose of the second campaign was to examine operator performance on a more diverse set of tasks, involving both gross motion and fine motion. As none of the operators had previous exposure to the hands-free teleoperation interface, and each operator attempted each task only once, the marginal likelihood of operators making substantial gains in proficiency during the course of the experiment made the choice to progress from simpler to more difficult tasks an arbitrary one.

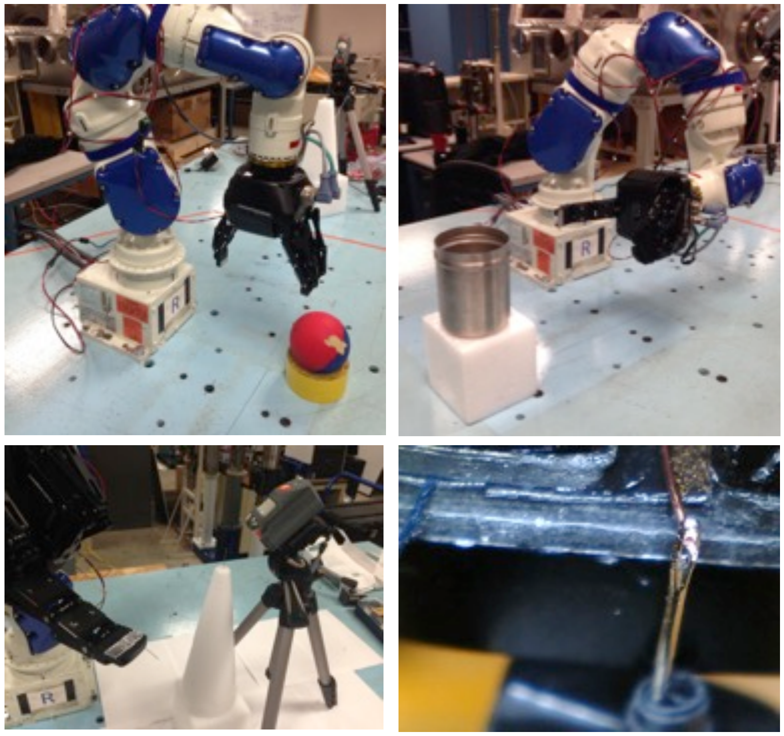

Figure 16: Multiple Operator Experiment – Ball, Can, Needle Setup and Microscope View

5.2 MEASUREMENTS AND SURVEY QUESTIONS

5.2.1 Task Completion Time

The task completion time was recorded for all tasks along with the number of executed operations. This allowed for the calculation of the command rate, i.e. the number of operations per unit time.

5.2.2 Survey Questions

During the second experimental campaign (multiple operators), operators were asked to fill out the short survey shown in Figure 17 after the conclusion of each task.

Session:		N/A	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	I felt in control most of the time.						
2	That was frustrating.						
3	I knew what was going on at all times.						
4	I would improve with practice.						
5	The interface was easy to use.						
6	I don't feel any more tired than when I started.						
7	The system was responsive to my commands.						
8	I couldn't concentrate.						
9	This was an enjoyable experience.						
10	I was able to complete the task.						

Comments:

Figure 17: Operator Post-Task Survey

Survey questions were designed to measure operator perception of system controllability, situational awareness, and/or ease of use. Questions were phrased positively or negatively in an attempt to reduce acquiescent and extreme response bias. The usability survey question matrix is provided in Table 3.

C	A	E		Question
+			**1**	I felt in control most of the time.
–			**2**	That was frustrating.
+	+		**3**	I knew what was going on at all times.
			4	I would improve with practice.
		+	**5**	The interface was easy to use.
	–	+	**6**	I don't feel any more tired than when I started.
+	+		**7**	The system was responsive to my commands.
	–		**8**	I couldn't concentrate.
		+	**9**	This was an enjoyable experience.
		+	**10**	I was able to complete the task.

Table 3: Usability Survey Question Matrix

Operator responses were scored from 1 (Strongly Disagree) to 5 (Strongly Agree) and the results aggregated. Negative question scores were converted to positive scores by a linear mapping ($p = 6 - n$).

5.3 Experimental Results

5.3.1 Single Operator Task Completion Times

	Command Line	GUI	Hands-Free
Avg. Time (s)	151	81	104
Avg. Num. Ops.	16.7	10.5	29.7
Avg. Cmd. Rate (s^{-1})	0.102	0.138	0.288

Table 4: Single Operator Task Completion Times

5.3.2 Multiple Operator Task Completion Times

	GUI			Hands-Free		
	Avg. Time (s)	Avg. Num. Ops.	Avg. Cmd. Rate (s^{-1})	Avg. Time (s)	Avg. Num. Ops.	Avg. Cmd. Rate (s^{-1})
Task 1	40.75	6.250	0.1533	11.00	3.250	0.2923
Task 2	38.50	6.000	0.1560	27.00	4.000	0.1663
Task 3	96.75	13.75	0.1534	182.5	48.25	0.2870
Task 4	166.0	27.50	0.1682	230.0	49.00	0.2130

Table 5: Multiple Operators Task Completion Times

5.3.3 Multiple Operator Usability Scores

	GUI			Hands-Free		
	Control	Ease of Use	Awareness	Control	Ease of Use	Awareness
Task 1	4.06	4.44	3.56	2.44	3.31	3.19
Task 2	4.44	4.19	3.63	2.38	3.19	2.88
Task 3	3.75	3.81	3.50	3.75	3.63	3.56
Task 4	3.94	3.50	4.00	3.56	3.50	4.00
Average	**4.05**	**3.98**	**3.67**	**3.03**	**3.41**	**3.41**

Table 6: Multiple Operator Usability Scores

5.3.4 Operator Comments

In both campaigns, operators were critical of two aspects of the hands-free teleoperation interface, the first being insufficient visual feedback and situational awareness, and the second being the use of relative rather than absolute orientations. All operators agreed that the visual cues provided for scale and for the position of the gripper frame (from which the commanded pose was relatively defined) were insufficient. Further, all but one operator found it extremely difficult to adapt to the command axes changing with changing gripper orientation, and stated that they would prefer to work in an absolute/world coordinate frame.

5.4 ANALYSIS

5.4.1 Single Operator

As noted in [69], even though the fewest commands were issued using the command line, and performing high precision operations with the command line is conceptually simplest, the command rate was far lower than that achieved with the other interfaces, and so operator productivity was lowest.

By contrast, the command rate for the hands-free teleoperation interface was more than twice that of the GUI and nearly three times that of the command line interface, which strongly suggests that the operator was more able to focus on performing the task than the issuing of commands, i.e. that the operator spent far less time and effort on context switching. In fact, it was determined that this rate was limited by the cycle time of the motion planner, not by the operator or interface. It should also be noted that commands in both the command line and GUI were issued along individual Cartesian axes, effectively providing operators with virtual fixtures, which made it much more difficult for operators to perform unintentional movements.

The completion time was greater for the hands-free interface than the GUI, most likely due to the operator's deep familiarity with graphical interfaces and complete lack of familiarity with the hands-free interface. It was observed that over the course of the three attempts the operator's task completion time decreased from 137 to 65 seconds, suggesting that familiarity can be gained and learning can be accomplished very quickly.

5.4.2 Multiple Operators

Again, the command rate achieved with the hands-free teleoperation interface was approximately twice that of the GUI, and this rate was limited by the cycle time of the motion planner, not by the interface. This strongly suggests that the operators were more able to focus on performing tasks than on the issuing of commands, i.e. that the operators spent far less time and effort on context switching. Again, commands in the GUI were composed from translations and rotations about individual Cartesian axes, effectively providing operators with virtual fixtures, which made it much more difficult for operators to perform unintentional movements with the GUI than with the hands-free teleoperation interface.

Operators generally perceived that the hands-free teleoperation interface was less usable than the GUI interface, though it should be remembered that operators had never seen or used such an interface before, and had no training other than a short verbal explanation before they began. While the difference in usability became less pronounced as the tasks became more difficult, tasks were attempted in increasing order of difficulty, so it's likely that usability scores were improving simply because operators were simply becoming more familiar with the hands-free teleoperation interface.

Chapter 6: Discussion and Conclusions

6.1 Discussion

This work has demonstrated the viability of a hands-free interface for user-centered teleoperation, and experimental results and analysis were included in Chapter 5. The initial prototype system developed was used to perform gross motion (pick-and-place) as well as fine motion (threading a needle), validating the approach. While this system can be used to perform useful work in its current state, a number of improvements have been identified that could significantly improve the performance of the system.

6.2 Future Work

The focus of this work was on kinematic rather than dynamic interaction, and so the system currently has no ability to detect or respond to contact forces on the manipulator or end-effector. These capabilities have been separately developed for IRAD by Kyle Schroeder [66] and Andy Zelenak [67] of the Nuclear Robotics Group, and should be straightforward to integrate into the RTOS control software, allowing this work to be extended to contact tasks. Currently, the ROS-AX driver interface allows trajectories to be interrupted but this requires the robot to come to a stop. Adding reactive motion capabilities will require modifications to the driver interface to permit trajectories to be modifiable on the fly, as well as the implementation of trajectory blending algorithms in order to ensure smooth transitions.

Further extensions include implementing dual-arm control by using foot pedals to execute trajectories as well as to control the scale factor; testing the system with other robot platforms, such as VaultBot; and performance of glovebox dexterity tests e.g. the Purdue Pegboard [70] and Minnesota Dexterity Test [71] to enable direct productivity comparisons with gloved glovebox workers.

As discussed in Chapter 2, operator productivity is a function of the rate at which operators can issue commands without error, which is currently a function of the path planner cycle time as well as the time-optimality of the generated paths. The sampling-based motion planners used in this work trade optimality for speed, or vice versa; even so, currently OMPL's optimal sampling-based motion planner is on the order of 10-100 times too slow for real-time hand position tracking. Further, the paths these motion planners generate are not repeatable, which has significant negative consequences for assembly and other manufacturing tasks. Optimal motion planners have been developed for ROS, but are not widely used or supported. For example, the *trajopt* motion planner has many desirable features, but the code is no longer under active development and will require rework before it can be used with the current version of ROS. Development of a true real-time time-optimal motion planner for ROS with inverse kinematics that can intelligently exploit redundancy in 7+ DOF serial chains (as opposed to standard pseudoinverse-based inverse kinematics that yield minimum norm solutions) will have significant impact on the ROS and ROS-Industrial community.

Finally, it should be recognized that the real and virtual instrumentation that enables operator situational awareness is a vital and inseparable part of any teleoperation system, and in many ways is more important than the control system. Television cameras are a technology as old as teleoperation; while cameras have come a lot further than teleoperators, the difference between a fixed television camera and the eyes is not unlike the difference between teleoperation and working with the hands. As discussed in Chapter 2, operator productivity is a function of response time, which is a function of stimulus intensity. Performing work with only a distorted, pixelated, flat representation of a fixed perspective that's almost certainly too far from the task to provide feedback at the optimal intensity will result in long sensory integration times, excess cognitive load, and reduced productivity. Operators clearly stated a desire for more prominent visual feedback of scale and reference, user-selectable reference frames, as well as more cameras and other instrumentation to facilitate greater situational awareness. There is tremendous scope to improve both instrumentation and information presentation, both in software (e.g. augmented reality) and in hardware (e.g. hyperspectral imaging).

6.3 CONCLUSIONS

A hands-free interface for robotic teleoperation was developed to enable accurate, safe, low cost and efficient robotic teleoperation. Historically, teleoperation research has focused on the technical challenges of ensuring stability and minimizing delay; today's challenges call for a multidisciplinary approach to optimizing operator productivity. The inherent limitations of the bilateral/master-slave approach mean that future performance gains can only be achieved by delegating control authority, i.e. by relinquishing feedback in favor of autonomy. However, neither complete autonomy nor pure feedback control is necessary or desirable; shared autonomy with minimal feedback offers significant advantages both in terms of cost and performance. Moving forward, today's task-focused high feedback model must eventually give way to a goal-focused model that emphasizes shared autonomy, with a human interface that emphasizes feedforward precision rather than corrective feedback, and employs appropriate sensory feedback only where autonomy is impossible.

A prototype system was developed and demonstrated, performing tasks between anthropomorphic and sub-millimeter scales, in both simulation and hardware. Initial results indicate that the hands-free teleoperation system permits a higher command rate than traditional interfaces, and is currently limited not by the operator but by the cycle time of the robot control software. It offers the potential to improve operator health and safety as well as facility security, and enables operators to work with smaller parts and samples than are currently possible.

Glossary

ALARA	As Low As Reasonably Achievable
API	Application Programming Interface
ASIC	Application Specific Integrated Circuit
CMOS	Complementary Metal-Oxide Semiconductor
CPU	Central Processing Unit
COTS	Commercial Off The Shelf
DOF	Degrees Of Freedom
IRAD	Intelligent Reconfigurable Anthropomorphic Dual-arm system
KDL	Orocos Kinematics and Dynamics Library
NRG	Nuclear Robotics Group
NUI	Natural User Interface
PLC	Programmable Logic Controller
OMPL	Open Motion Planning Library
RGB-D	Red, Green, Blue, and Depth

ROS	Robot Operating System
RTOS	Real-Time Operating System
SDK	Software Development Kit
SLERP	Spherical Linear interpolation
TCP/IP	Transmission Control Protocol/Internet Protocol
TRL	Technology Readiness Level
URDF	Unified Robot Description File
USB	Universal Serial Bus
XML-RPC	eXtensible Markup Language-Remote Procedure Call

Bibliography

[1] D. A. Norman, *The Design of Everyday Things*, 2nd ed. New York: Basic Books, 2002.

[2] R. C. Goertz, "Master-Slave Manipulator," Argonne National Laboratory, Argonne, IL, ANL-4311, 1949.

[3] S. Y. Nof, Ed., *Handbook of Industrial Robotics*, 2nd ed., vol. 1. New York, New York, USA: John Wiley & Sons, 1999.

[4] C. Dillingham, "Transforming Pits into Clean Energy," *National Security Science Magazine*, pp. 18–25, Feb-2012.

[5] D. Wigdor and D. Wixon, *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Burlington, MA: Morgan Kaufmann, 2011.

[6] H. B. Suay and S. Chernova, "Humanoid robot control using depth camera," in *Proceedings of the 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2011.

[7] L. Cheng, Q. Sun, H. Su, Y. Cong, and S. Zhao, "Design and implementation of human-robot interactive demonstration system based on Kinect," in *2012 24th Chinese Control and Decision Conference (CCDC)*, 2012, pp. 971–975.

[8] W. Song, X. Guo, F. Jiang, S. Yang, G. Jiang, and Y. Shi, "Teleoperation Humanoid Robot Control System Based on Kinect Sensor," in *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, 2012, vol. 2, pp. 264–267.

[9] A. Malima, E. Ozgur, and M. Cetin, "A Fast Algorithm for Vision-Based Hand Gesture Recognition for Robot Control," *2006 IEEE 14th Signal Process. Commun. Appl.*, 2006.

[10] D. Saffer, *Designing Gestural Interfaces*. Sebastopol, Canada: O'Reilly, 2009.

[11] W. V. F. B. Tomas Dekens Yorgos Patsis and F. Capman, "A Multi-sensor Speech Database with Applications towards Robust Speech Processing in hostile Environments," in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, 2008.

[12] R. Y. Wang, S. Paris, J. Popović, and J. Popovi, “6D Hands: Markerless Hand-Tracking for Computer Aided Design,” in *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST ’11*, 2011, pp. 549–557.

[13] M. T. Taylor, A. Chandak, L. Antani, and D. Manocha, “Resound: interactive sound rendering for dynamic virtual environments,” in *Proceedings of the 17th ACM international conference on Multimedia*, 2009, pp. 271–280.

[14] S. Micera, J. Carpaneto, and S. Raspopovic, “Control of hand prostheses using peripheral information,” *IEEE Rev. Biomed. Eng.*, vol. 3, pp. 48–68, 2010.

[15] R. C. Goertz, “Report on Argonne National Laboratory’s Activities,” in *Proceedings of the 1964 Seminars on Remotely Operated Special Equipment Held at Nuclear Rocket Development Station, Jackass Flats, Nevada, November 4 and 5, 1964*, 1964, vol. II, pp. 41–47.

[16] C. Kapoor, “A Reusable Operational Software Architecture for Advanced Robotics,” The University of Texas at Austin, 1996.

[17] R. Y. Wang and J. Popović, “Real-Time Hand-Tracking with a Color Glove,” in *ACM SIGGRAPH 2009 Papers*, 2009, pp. 63:1–63:8.

[18] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Berlin: Springer, 2008.

[19] T. B. Sheridan, “Telerobotics,” *Automatica*, vol. 25, no. 4, pp. 487–507, Jul. 1989.

[20] T. B. Sheridan, “Teleoperation, telerobotics and telepresence: a progress report,” vol. 0661, no. 2, pp. 205–214, 1995.

[21] P. Hokayem and M. Spong, “Bilateral teleoperation: An historical survey,” *Automatica*, vol. 42, no. 12, pp. 2035–2057, Dec. 2006.

[22] R. C. Goertz, “Mechanical Master-Slave Manipulator,” *Nucleonics*, 1954.

[23] J. Needham, “Science and Civilization in China: Volume 6, Biology and Biological Technology, Part 5, Fermentations and Food Science.” Cambridge University Press, Cambridge, UK, 2000.

[24] R. C. Goertz, “Manipulator Development at the ANL,” *Trans. Am. Nucl. Soc.*, vol. 7, p. 467, 1964.

[25] W. R. Ferrell and T. B. Sheridan, "Supervisory control of remote manipulation," *Spectrum, IEEE*, vol. 4, no. 10, pp. 81–88, Oct. 1967.

[26] "Central Research Laboratories Telemanipulators Catalog," 2011. [Online]. Available: http://www.destaco.com/assets/docs/en/catalogs/crl/Telemanipulators.pdf.

[27] R. Bicker, K. Burn, Z. Hu, W. Pongaen, and A. Bashir, "The early development of remote tele-manipulation systems," in *International Symposium on History of Machines and Mechanisms*, 2004, pp. 391–404.

[28] R. C. Goertz, J. R. Burnett, and F. Bevilacqua, "Servos for remote manipulation," Argonne, IL, 1953.

[29] D. A. Lawrence, "Stability and transparency in bilateral teleoperation," *Robot. Autom. IEEE Trans.*, vol. 9, no. 5, pp. 624–637, Oct. 1993.

[30] S. Sanders, "Remote operations for fusion using teleoperation," *Ind. Robot An Int. J.*, vol. 33, no. 3, pp. 174–177, 2006.

[31] R. W. Daniel and P. R. McAree, "Fundamental Limits of Performance for Force Reflecting Teleoperation," *Int. J. Rob. Res.*, vol. 17, no. 8, pp. 811–830, Aug. 1998.

[32] T. B. Sheridan and W. R. Ferrell, "Remote Manipulative Control with Transmission Delay," *Hum. Factors Electron. IEEE Trans.*, vol. HFE-4, no. 1, pp. 25–29, Sep. 1963.

[33] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2010.

[34] The University of Western Australia, "The UWA Telerobot," 2004. [Online]. Available: http://telerobot.mech.uwa.edu.au/Telerobot/index.html. [Accessed: 12-Jan-2014].

[35] T. B. Sheridan, *Telerobotics, automation, and human supervisory control*. Cambridge, MA: MIT Press, 1992.

[36] M. E. Cournoyer, V. E. Garcia, U. F. Gallegos, and D. W. Wilburn, "Investigation of injury/illness data at a nuclear facility," *J. Chem. Heal. Saf.*, vol. 18, no. 5, pp. 17–25, 2011.

[37] J. C. Rothwell, M. M. Traub, B. L. Day, J. A. Obeso, P. K. Thomas, and C. D. Marsden, "Manual Motor Performance in a Deafferented Man," *Brain*, vol. 105, no. 3, pp. 515–542, 1982.

[38] J. Bodner, H. Wykypiel, G. Wetscher, and T. Schmid, "First experiences with the da Vinci™ operating robot in thoracic surgery," *Eur. J. Cardiothorac. Surg.*, vol. 25, no. 5, pp. 844–851, 2004.

[39] D. Kubus, I. Weidauer, and F. M. Wahl, "1kHz is not enough — How to achieve higher update rates with a bilateral teleoperation system based on commercial hardware," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 5107–5114.

[40] D. L. Kohfeld, "Simple reaction time as a function of stimulus intensity in decibels of light and sound.," *J. Exp. Psychol.*, vol. 88, no. 2, pp. 251–257, 1971.

[41] S. S. Stevens, "On the psychophysical law," *Psychol. Rev.*, vol. 64, no. 3, pp. 153–181, 1957.

[42] W. E. Hick, "On the rate of gain of information," *Q. J. Exp. Psychol.*, vol. 4, no. 1, pp. 11–26, Mar. 1952.

[43] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement.," *J. Exp. Psychol.*, vol. 47, no. 6, p. 381, 1954.

[44] R. J. Kosinski, "A Literature Review of Reaction Time," 2013. [Online]. Available: http://biae.clemson.edu/bpc/bp/lab/110/reaction.htm.

[45] M. S. A. Graziano and T. N. Aflalo, "Mapping behavioral repertoire onto the cortex," *Neuron*, vol. 56, no. 2, pp. 239–51, Oct. 2007.

[46] A. Cooper, *The Inmates are Running the Asylum*, 2nd ed. Indianapolis, IN: SAMS, 2004.

[47] J. J. Craig, *Introduction to robotics: mechanics and control*. Upper Saddle River, NJ, USA: Pearson/Prentice Hall, 2005.

[48] PrimeSense, "PrimeSense 3D Sensors." [Online]. Available: http://www.i3du.gr/pdf/primesense.pdf.

[49] K. Shoemake, "Animating Rotation with Quaternion Curves," *SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 245–254, Jul. 1985.

[50] J. Kuipers, "Quaternions and Rotation Sequences," *Proc. Int. Conf. Geom. Integr. Quantization*, vol. 1, pp. 127–143, 1999.

[51] 3Gear Systems, “Gestural User Interface: Protocol,” *3Gear Systems Website*, 2014. [Online]. Available: http://threegear.com/latest/doc/protocol.html.

[52] Z. Zhang, “A Flexible New Technique for Camera Calibration,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000.

[53] R. Y. Wang, S. Paris, and J. Popovi, “6D Hands : Markerless Hand Tracking for Computer Aided Design.”

[54] A. Torralba, R. Fergus, and W. T. Freeman, “Tiny Images,” Cambridge, MA, 2007.

[55] Open Source Robotics Foundation, “ROS.org | Powering the world’s robots,” 2014. [Online]. Available: http://ros.org.

[56] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “ROS: an open-source Robot Operating System,” in *ICRA Workshop on Open Source Software*, 2009, vol. 3, no. 3.2.

[57] S. Edwards and C. Lewis, “ROS-Industrial - Applying the Robot Operating System (ROS) to Industrial Applications,” in *IEEE Int. Conference on Robotics and Automation, ECHORD Workshop*, 2012.

[58] “ROS-Industrial,” 2014. [Online]. Available: http://www.rosindustrial.org.

[59] I. A. Sucan, M. Moll, and L. E. Kavraki, “The Open Motion Planning Library,” *IEEE Robot. Autom. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[60] J. Pan, S. Chitta, and D. Manocha, “FCL: A general purpose library for collision and proximity queries,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 3859–3866.

[61] Torus Knot Software, “Ogre3D,” 2014. [Online]. Available: http://www.ogre3d.org.

[62] Yaskawa Motoman Robotics, “SIA5D Datasheet,” *DS-453-E*, 2014. [Online]. Available: http://www.motoman.com/datasheets/SIA5d.pdf.

[63] D. Jansen and H. Buttner, “Real-time Ethernet: the EtherCAT solution,” *Comput. Control Eng.*, vol. 15, no. 1, pp. 16–21, 2004.

[64] “3-Finger Adaptive Robot Gripper,” 2014. [Online]. Available: http://robotiq.com/en/products/industrial-robot-hand. [Accessed: 05-Feb-2014].

[65] J. Hashem, “The Requirements and Implementation of Dynamically-Deployed Robotic Systems for Use in Confined, Hazardous Environments,” The University of Texas at Austin, 2012.

[66] K. A. Schroeder, M. Pryor, and T. Harden, “A Black Box Model for Estimating Joint Torque in an Industrial Serial Manipulator,” in *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2013, pp. V06BT07A062–V06BT07A062.

[67] A. Zelenak, M. Pryor, and K. Schroeder, “Integration of Compliance and Collision Detection Algorithms for Industrial Systems,” in *Proceedings of the Decommissioning and Remote Systems Embedded Topical Meeting at the American Nuclear Society 2014 Annual Meeting, June 15-19 2014, Reno, Nevada.*, 2014, pp. 103–105.

[68] J. Hashem, J. Hunter, and M. Pryor, “Automating X-Ray and Non-Destructive Testing Applications,” in *Transactions of the American Nuclear Society, Vol. 109, November 10-14 2013*, 2013, pp. 517–520.

[69] J. Thompson and M. Pryor, “Spatial Interface for User-Centered Robotic Teleoperation and Demonstration on a High Precision Task,” in *Proceedings of the IROS 2014 Workshop on Telerobotics for Real-Life Applications: Opportunities, Challenges, and New Developments*, 2014.

[70] J. Tiffin and E. J. Asher, “The Purdue Pegboard: Norms and studies of reliability and validity,” *J. Appl. Psychol.*, vol. 32, no. 3, p. 234, 1948.

[71] J. Desrosiers, A. Rochette, R. Hébert, and G. Bravo, “The Minnesota Manual Dexterity Test: reliability, validity and reference values studies with healthy elderly people,” *Can. J. Occup. Ther.*, vol. 64, no. 5, pp. 270–276, 1997.