A Telerobotics Construction Set with Integrated Performance Analysis *

Marjorie Skubic †, George V. Kondraske ‡, James D. Wise §, George J. Khoury ‡, Richard A. Volz †, Scott Askew ¶

Abstract

The Universities Space Automation and Robotics Consortium (USARC) was established to promote research into robotics and telerobotics for remote applications. An important part of the work of the Consortium has been the design and implementation of the Telerobotic Construction Set (TCS), which enables the building of modular telerobotic networks. On-line and off-line performance analyses are integrated into TCS using quantitative analytical models. This provides the capabilities of measuring and predicting performance and system workloads quantitatively under different conditions, such as different levels of remote autonomy or different operator interfaces. This paper presents the methodology used in the performance/workload analysis and describes recent experiments, measuring teleoperation performance under time delay.

1 Introduction

The term telerobotics has been used to represent human control of remote subsystems, including supervisory control of autonomous components [3]. Despite thirty years of research, few systems exist which enable the research, development and evaluation of telerobotic systems at both the systems level and the component level. An open research question still exists concerning the prediction of performance of telerobotic systems, given component characteristics and task information.

The Universities Space Automation/Robotics Consortium (USARC) was formed in 1990 by four universities (Rice University, Texas A&M University, the University of Texas at Austin, and the University of Texas at Arlington) in conjunction with NASA's Johnson Space Center, to promote research into robotics and telerobotics for remote applications. A main motivation for USARC's work has been the research in telerobotic control over large distances, such as control of space-based robots from Earth.

Several challenges exist in such long-distance control, including large, fluctuating delays (several seconds for telemetry and propagation delays), potential interruptions and loss of data, and limitations in the bandwidth between the telecontroller and the robot. Because classical control systems require a high bandwidth and low-delay data link between the hand controller and the robot [16], it becomes extremely difficult for an operator to directly control the motion of the robot as larger delays are incurred. Sheridan provides an excellent history and overview of problems and methods used in teleoperation [18].

Evaluation of telerobotic systems at the systems level has generally been task-oriented. That is, can an operator accomplish a particular task and how long does it take [8, 19]? Although this provides an indication of whether the system works for that particular task, it does nothing to characterize the performance of the components or indicate where the limitations and weaknesses exist. It also does not provide information which could be used to predict the performance in a different task.

Other studies have been done which measure the effect of changing one component. For instance, manual control modes have been compared to test the effect of force reflection [4, 14]. Manual and shared control modes have been compared [7]. These studies were task-oriented and did not provide a mechanism for characterizing the component or predicting the performance of a system under different tasks.

A few researchers have used strategies that provide performance prediction. In [1], Fitt's Law and a target-threshold model are used to predict movement times in a camera pointing operation. A hybrid 2-port matrix is used in [6], which attempted to predict selected aspects of system performance from component-level information.

In contrast with all of the previous efforts, we want to provide an open architecture that facilitates component-level and system-level design and evaluation. Essential requirements include the identification of a generic telerobotic systems model, approprpriate performance models for each generic component type, and quantitative characterizations of specific components. We contend that this approach will lead to the prediction of system performance in previously untried tasks.

The focus of this paper is on the roles and meth-

*This work has been supported by NASA under grant numbers NAG9-461 and NCC2-517, by the State of Texas under TATP grants 999903-267 and 999903-095, and by the NSF under grant CDA-9115123. M. Skubic is funded by the U.S. Department of Education.

†Department of Computer Science, Texas A&M University, College Station, TX 77843-3112

‡Human Performance Institute, The University of Texas at Arlington, Arlington, TX

§Department of Electrical and Computer Engineering, Rice University, Houston, TX

¶Automation and Robotics Division, NASA Johnson Space Center, Houston, TX

ods of performance/workload modeling, measurement, and analysis, as it applies to telerobotic systems. Section 2 briefly describes the USARC Telerobotic Construction Set and shows how it facilitates monitoring and analysis. Section 3 presents concepts of the performance and workload analysis theory, and how it is applied to telerobotic systems. Section 4 describes the design and selected results of two experiments, using variable time delays and human-only and shared control modes. Conclusions are given in Section 5.

2 USARC Telerobotics Construction Set

Our approach to long-distance telerobotics includes an emphasis on an open, network-based architecture that supports a heterogeneous combination of computers and facilitates component and system-level monitoring. With that in mind, a Telerobotic Construction Set (TCS) has been designed and implemented. The TCS can be used to build telerobotics systems operating over a wide-area network, for research in telecontrol over large distances (i.e., different cities). The TCS provides for the connection and configuration of a variety of resources, including robotic devices, manual controllers, cameras, computers, and force sensors [13, 5].

Figure 1 is an example of a typical modular configuration for teleoperation: an operator control site, a remote robot work site, and a monitoring site.[1] It consists of a number of clearly defined functional modules communicating via streams of data packets using a few well-defined formats. In such a diagram, the modules themselves and the data they produce and consume define the system. Rather than data being sent to a specific destination, resulting in a 'hard-wired' connection, modules specify the types and other characteristics of the data they wish to receive. A 'virtual connection' will be made to whichever modules provide the required data. Since this connection is dynamic, modules may be added to or removed from the configuration at any time with minimal impact on the rest of the system. This approach provides a means of standardizing the interface with various devices, simulations and controls without predetermining the exact implementation of each module.

2.1 System components

Conceptually, our Telerobotic Construction Set consists of three components: a data exchange mechanism which delivers the data objects from the producers to the consumers; a standard, minimal set of data objects which encapsulate the information communicated among the modules; and a set of modules which perform well-defined, generic functions, including performance monitoring.

Data exchange. The modules of the TCS are connected via a general purpose data exchange system called the TeleRobotics Interconnection Protocol (TelRIP)[2], which was developed by Rice University. In a system using TelRIP each multitasking processor runs a *router* process. The router is responsible for maintaining connections between local processes and forwarding information to other routers. Each processing module (PM) sends messages to the router indicating data types in which it is interested. When a PM distributes data, it sends it to its router, the router distributes it to any other routers interested in it, and they, in turn, send it to the appropriate local processes. Note that data is not broadcast, it is sent only to parties which have expressed an interest in it. For more detail on TelRIP, please refer to [2].

Data object vocabulary. The unit of communication between modules is the TelRIP Data Object. A data object is similar to an object oriented programming object in that it contains a self identifying data record. TelRIP data objects lack an explicit class hierarchy and embedded method code. However, they incorporate additional distinquishing features, called properties which are used in classifying and directing objects. These include:

- A timestamp, which gives the time at which an object was distributed. TelRIP maintains synchronization of timestamps among systems, accurate to within a few milliseconds.
- A source identifier or *address*.
- An intended destination address. This may indicate either a specific module, or any module providing a particular *function* (e.g., any hand controller).
- Qualifying properties (e.g. compressed vs. uncompressed, status information vs. action command, operator generated vs. program generated).

Although TelRIP does not enforce an explicit class hierarchy, it is convenient to group the object types into classes. The following classes are supported: sensor data, motion, control, configurations parameters, and communication.

Functional modules. The functional modules are implemented by a set of programs called Processing Modules. Like the data object types, the functional modules may be grouped into classes: operator interface, motion control, configuration control, performance monitoring, and video. For more details on the TCS data objects and functional modules, please refer to [15, 20].

2.2 Performance monitoring capabilities

One goal of the TCS is to provide a data gathering capability that can be used for performance monitoring. Several aspects of TCS make this straightforward,

[1] Note that the configuration might also contain more than one work site, control site, or monitoring site.

[2] The TelRIP system described in this paper is commercially available as InterAgent from Modulus Technology, Inc., Houston, Texas.

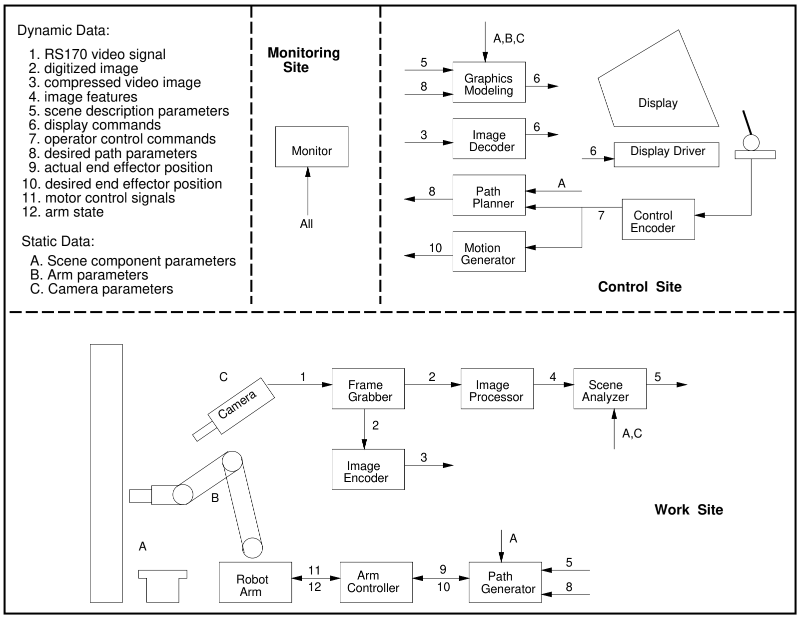

Figure 1: **A Modular Configuration for Teleoperation**

including message tracing, timestamping, and the ease of data logging. Possibly the most significant is the ease with which the undirected messaging mechansim allows a monitoring program to "tap into" any data stream. Because data are routed based on their type and properties, a monitor can receive copies of (1) all data of a particular type, (2) all data to or from a particular process or between a pair of processes, or (3) all data distributed during the entire session, simply by registering interest in the appropriate patterns. The monitor will receive the same data that other programs receive. Because this can be done without the cooperation (or even the knowledge) of the processes being monitored, monitoring can be added to any system at any time without prior preparation.

If the actual data contained in an object is not required, but only an indication of the type and properties, then the message tracing capability of the routers can be used. With message tracing, each router creates a condensed record for each data object it handles, containing the Base Data Type (BDT) identifier, size, source and destination address, and properties (including timestamp). A trace record is also generated each time a program starts or terminates. The router builds a list of these records and periodically distributes this list to any interested process.

Since only an abbreviated message header is used in a trace, the overhead is much smaller than for monitoring the actual data objects. For a typical session, tracing constitutes less than 5% of the total traffic. If no process is requesting traces, then no trace objects are distributed and no additional overhead is incurred.

Both the data objects and their trace records are timestamped. Since TelRIP maintains network timestamp synchronization among sites to within a few milliseconds, it is possible to estimate network, processing, and operator delays from this data. However, it is necessary to remember that the timestamp is the time at which the object was received by the originating router, which in a non-realtime multiprogramming environment may differ somewhat from the time at which it was distributed by the application.

There are several programs available to provide access to this raw session data. These include an X Window session monitor which displays the session configuration (programs running at each site) and can take snapshots of data traffic, a session tracer which records the trace of an entire session to disk for later analysis, and a trace tool which allows a session trace to be interactively displayed. One of the capabilities of the trace tool is to provide a synchronized picture of session activity. This can take a number of forms: (1) an ordered list of all data objects, (2) a filtered list containing only objects of a particular type or set types, or to or from a particular process, (3) a plot of average data rate as a function of time for each BDT, or (4) a chronology of program activity.

3 Theoretical basis for analysis of system performance and workload

We have employed a relatively new set of constructs and methods known as General Systems Performance Theory (GSPT) [10] as a conceptual basis for our performance and workload analysis strategy. GSPT was motivated by performance measurement issues in human performance, but evolved to address general problems of quantifying system performance. We also use GPST to help understand the contribution of multiple subsystems to a system's ability to accomplish tasks.

While GSPT is described in detail in the literature [12, 11], a brief overview of key ideas is warranted here. First, within the domain of interest, a system is identified along with its function (i.e., purpose). Note that any hierarchical level of abstraction may be selected. This discussion applies to total systems or system components. At the level selected, the *system* and the *task* are considered separately.

The system performance is modeled in terms of available performance resources (i.e., performance capacities). This is done by considering first what factors determine how well a system executes its function. Based on that determination, dimensions of performance are identified, which represent performance resources. For each dimension, a metric is defined so that a larger numerical value consistently represents *more* availability. Available performance resources can be measured using multiple test-tasks, which are typically similar to operational tasks but are designed to maximally stress the system along selected dimensions of performance. Examples of performance dimensions include speed, accuracy, smoothness, and endurance. Performance tests might measure the amount of performance resources available with test runs "as fast as possible", "as accurately as possible', "as smoothly as possible", or "for as long as possible".

Task characterization is achieved by either measuring or estimating the demands on performance resources, using the same dimensions and metrics as for the available resources. Resource availability can then be compared to resource demands to predict the level of performance. If resource availability $\geq$ resource demand (for each performance dimension), then we can predict successful task completion.

Given this framework, a quantitative definition for workload can be defined based on the stress under which a given performance resource is placed. That is, workload is equivalent to resource stress, which is equivalent to resource utilization/ resource availability. In complex systems, workloads can be identified at multiple hierarchical levels. The lower level systems where workloads approach 100% represent the factors which limit performance of the overall system.

One advantage of GSPT is that the same performance analysis strategy can be applied to both human and artificial systems. This provides a framework which allows seamless integration of both human and machine systems or components.

To apply the theoretical framework, a generic telerobotic systems model is used, as shown in Figure 2. This model contains two hierarchical levels of abstraction: (1) the overall system and (2) major subsystems. At each level, performance resource availability, utilization, demand, and workload associated with the various dimensions of performance are considered. While our current goal is to characterize performance capacities of all major subsystems in a specific telerobotic system (and ultimately predict performance at the overall system level), work to date has focused on the human operator subsystem and measures of overall system performance that involve remote manipulation. Specifically, measures of neuromotor workload associated with manual controller operation are obtained using an information theory based model that allows derivation of the amount of channel capacity utilized during an operation [9]. These are combined with measures of neuromotor channel capacity availability based on Fitts' Law type tests. In the case of the latter, the human system is removed totally from the telerobotic system context and test tasks are performed that maximally stress neuromotor channel capacities.

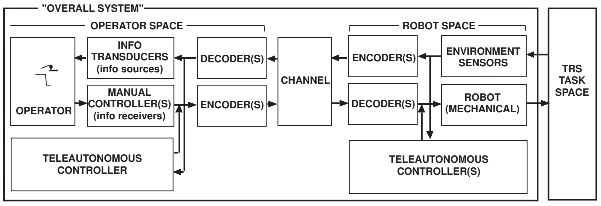

Figure 2: **Generic Module-Level Systems Performance Model**

Similar strategies would be employed for selected performance resources associated with other major subsystems. In addition, software modules have been developed that compute measures of range, speed, accuracy, and smoothness from end-effector position vs. time data. These performance parameters are being used in combination with neuromotor workload to compute metrics of work efficiency .

4 Experiments

In this section, we present an overview of how the TCS and the strategy described above are used in an experimental testbed. Two experiments are discussed. While both of these experiments have been designed to address telerobotic systems integration questions, a primary goal at present is to evolve and demonstrate the utilization of a set of quantitative methods in these endeavors.

The USARC TCS was used for both experiments, which involved distributed telerobotics between universities. Network delays due to internet connections varied over a wide range, from 50 ms to 7 seconds. The average time delay was 100 to 200 ms. More details can be found in [17, 9].

4.1 Subsystem component experiment

The goal of the first experiment was to identify objective means for selecting the best candidate for a given major subsystem component. The component of interest was a multiple degree-of-freedom manual controller. Two initially untrained subjects executed repeated trials (three per day across three days, to observe the effects of experience) with three different performance modes based on a maximal demand strategy. The basic task involved alternate positioning of the end-effector between two targets: (1) as accurately as possible, (2) as fast as possible, and (3) as fast and accurately as possible. Six-degree-of- freedom manual controller and robot end-effector position time series data were logged by the Telerobotic Performance/Workload (TPW) Manager, which is part of the USARC TCS.

Figure 3: **Manual Controller Output as a Function of Time**

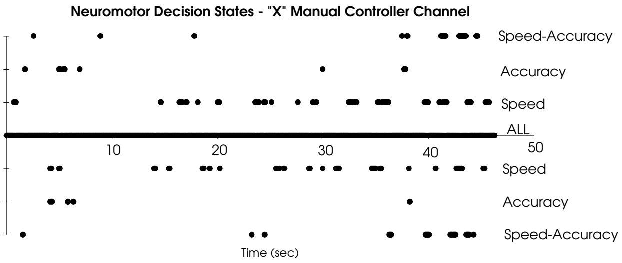

Figure 4: **Neuromotor decision state time series data for three different task modes.** Original decision states are (-1: Changing-to-Stable, 0: Stable-to-Stable, and 1: Stable-to-Changing). These are encoded here for display by multiplying data for different modes by a 1 (speed trial), 2 (accuracy trial), or 3 (speed-accuracy trial).

Figure 3 illustrates selected manual controller data for the same subject under two different performance modes, taking into account the current state of a given manual controller channel and the next state. Figure 4 illustrates a time series sample of these decision-making states for the three different task modes, in an encoded format to facilitate visualization (see figure legend). Note the difference in the relative frequency of occurrence of states encoded as positive and negative integers for each task mode. For trials stressing speed, state changes are most frequent. Trials stressing accuracy show most state changes as the target is approached. Trials that stress both speed and accuracy do not have as many state changes as speed trials, but do exhibit more than trials where only accuracy is stressed. Decision-making rate (in decisions per second) is inferred over a sliding window by determining the shortest time between decisions within the window and inverting it. In Figures 3 and 4, this is shown as rapid changes in output vs. time. The model then assumes that the operator is applying this decision-making rate.

End-effector position is recorded along with man-

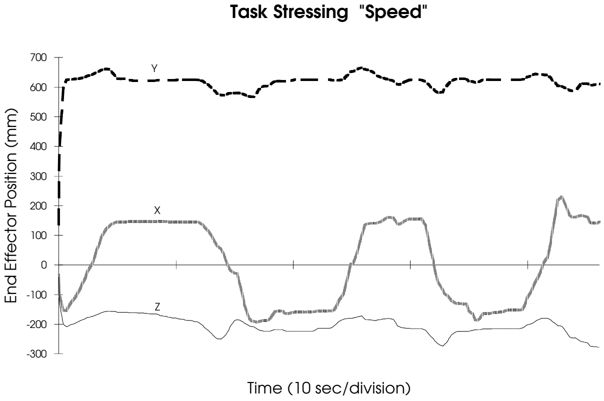

Figure 5: **End Effector Position for the Test Run "as fast as possible"**

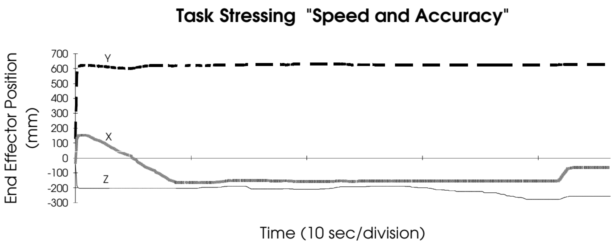

Figure 6: **End Effector Position for the Test Run "as fast and as accurately as possible"**

ual controller data and processed to compute overall task performance measures. Figures 5 and 6 show end-effector data for trials corresponding to those in Figure 4. It is clear that the number of cycles per unit time differs across different trials. A summary of measured results computed from both manual controller and end-effector data is listed in Table 1.

Mode/ Axis	Neuromotor Workload (percent)	Accuracy (1/um)	Average Speed (mm/sec)
Speed			
X	72.5	0.0009	50
Y	87.0		
Z	78.0		
Speed-Accur.			
X	<1.0	0.0010	22
Y	16.7		
Z	24.1		

Table 1:

Neuromotor workload is computed by expressing the channel capacity utilized (decision-making rate x entropy) as a fraction of an average healthy young adult's available upper extremity neuromotor channel capacity [12]. Accuracy is determined from knowledge of target position and calibrated end-effector position data, while speed is determined solely from end-effector data. These later two measures are considered to represent overall task performance capacity, while neuromotor workload indicates "what it takes" to obtain that level of overall performance with the system utilized.

4.2 Control modes experiment

To investigate the effects of increased autonomy upon telerobotic system performance, a second experiment was designed to make quantitative measurements of a task performed in both a human only and a shared mode. The human only mode used minimal autonomous controls, while the shared mode incorporated autonomous system components developed to minimize overall system workload values.

A simple task was selected, which could be performed in both the human-only and shared modes and would generate a marked quantitative contrast between the two different modes of operation. A simple peg-in-hole insertion was selected, where the operator must insert a square peg in a slotted hole with 0.01 inch clearance. The insertion task was segmented into 5 subtasks. In the human-only mode, the operator was assigned all the subtasks of a given task; in the shared mode, subtasks were assigned a level of autonomy, depending upon which system component could best execute the subtask.

For the shared mode peg insertion, shared control was accomplished by using force control to apply a vertical force downward while zeroing moments. The operator controlled the horizontal components while positioning the peg for insertion. In both the human-only and shared modes, a safety feature monitored the force exerted and halted robot operations when the maximum force limit was exceeded.

The USARC TCS described in Section 2 provided the communications infrastructure and functional modules for the test. To compensate for communication latencies, the operator used a predictive, graphical-display simulator (also part of the TCS), which moved a shaded PUMA model in real time in response to hand controller signals. The operator could then compare the shaded PUMA model to a superimposed wireframe PUMA model, which moved in response to PUMA end-effector positions, to determine the effects of network delay. However, the simulator did not account for excessive forces and moments, misalignments between peg and hole, or miscalibrations between the world model and actual workpieces.

To obtain quantitative measurements, data files were logged for control site data (time series hand controller signals and discrete commands), robot end effector position data, and network delay data for each peg-in-hole insertion. Tests were conducted over 4 sessions, until 3 successful insertions were recorded for each mode.

A successful task completion resulted when all subtasks were completed successfully. Robot operations resulting in singularities or exceeding the maximum force limits were considered failures. For each subtask and for the overall task, the task skill level was calculated (skill = success rate / average completion time). Figure 7 compares the normalized skill levels for the human-only and shared control modes. As shown, task skill is significantly increased using a shared control mode, demonstrating that the added autonomous

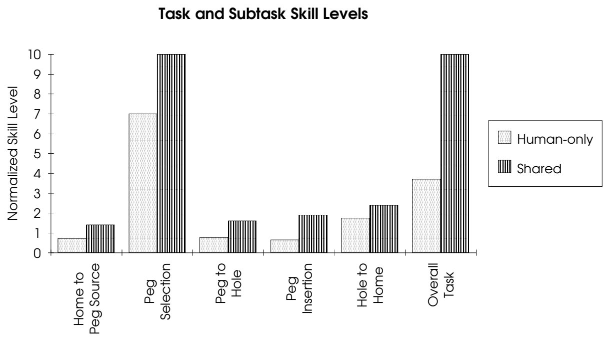

Figure 7: **Task Skill Level by Subtask and Overall Task.** Task skill = sucess rate / average completion time

features and enhanced feedback greatly improved the overall task skill level. Perhaps the biggest gain came from better error recovery. When the operator failed to execute a subtask in the shared mode, adequate feedback was provided early enough into the task so the operator could return to an earlier step and make another approach.

The results presented here are preliminary results and are designed to show how the USARC TCS can function as a telerobotic testbed. This experiment enabled us to show quantitative improvements in overall telerobotic system performance using autonomous controls. Final results will be compiled when measurement algorithms are applied to the collected data. Data collected during this experiment will aid in the development of workload and performance measurements that incorporate the effects of autonomous controls. It is anticipated that further insights into autonomous controls, including how to conduct experiments which provide objective, quantitative results, will be gained once data analysis is complete.

5 Conclusion

The USARC Telerobotics Construction Set is described, with emphasis on the performance and workload analysis strategy. The results of recent experiments are presented, which demonstrate the effectiveness of using the TCS for implementing a telerobotic testbed. Ongoing testing has reiterated the importance of a solid and reliable base for use in a testbed. Continuing efforts are being made to address limitations and provide improvements. The communications infrastructure has been upgraded to improve reliability and flexibility. A control architecture has been added to allow flexible merging of commands from numerous control sources. A new operator interface is being developed. Intelligent control modules will be added to increase autonomy at the robot site.

The operation of experiments is an ongoing process, with test sessions held weekly. Data from the test sessions are logged and saved for a database which will be used to characterize telerobotic components. The database can be used to determine where limitations exist, so efforts can be concentrated to improve weak components. Ultimately, the goal is to be able to predict the system performance level in previously untried tasks.

Acknowledgement

The authors would like to acknowledge the contributions of the entire USARC team, especially Peter Schilling, Sean Graves, and Bryan Morgan. We also want to thank the following people for their support: Don Johnson from Rice University, Delbert Tesar from the University of Texas at Austin, and Norm Chaffee and Charles Price from NASA/JSC.

References

[1] David J. Cannon and Larry J. Leifer. Speed and accuracy for a telerobotic human/machine system: Experiments with a target-threshold control theory model for fitt's law. In *Proceedings of the 1990 IEEE International Conference on Systems, Man and Cybernetics*, pages 677 – 679, 1990.

[2] L.A. Ciscon, J.D. Wise, and D.H. Johnson. A distributed data-sharing environment for telerobotics. *Presence*, 3(4), 1994.

[3] Lynn Conway, Richard A. Volz, and Michael W. Walker. Teleautonomous systems: Projecting and coordinating intelligent action at a distance. *IEEE Transactions on Robotics and Automation*, 6:146–158, April 1990.

[4] H. Das, H. Zak, W.S. Kim, A.K. Bejczy, and P.S. Schenker. Operator performance with alternative manual control modes in teleoperation. *Presence*, 1(2), Spring 1992.

[5] S. Graves, L. Ciscon, and J.D. Wise. A modular software system for distributed robotics. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, pages 2783–2785, Nice, May 1992. IEEE.

[6] B. Hannaford. A design framework for teleoperators with kinesthetic feedback. *IEEE Transactions on Robotics and Automation*, 5(4):426 – 434, August 1989.

[7] B. Hannaford, L. Wood, D.A. McAffee, and H. Zak. Performance evaluation of a six-axis generalized force-reflecting teleoperator. *IEEE Transactions on Systems, Man, and Cybernetics*, 21:620–633, May/June 1991.

[8] Gerd Hirzinger, Bernhard Brunner, Johannes Dietrich, and Johann Heindl. Sensor-based space robotics – ROTEX and its telerobotic features. *IEEE Transactions on Robotics and Automation*, 9(5):649–663, October 1993.

[9] G. J. Khoury and G. V. Kondraske. Measurement and continuous monitoring of human workload associated with manual control devices. Technical Report TR 91-011R, University of Texas at Arlington, 1991.

[10] G.V. Kondraske. Human performance: Science or art? In *Proceedings of the Thirteenth Northeast Bioengineering Conference*, pages 44–47, 1987.

[11] G.V. Kondraske. A working model for human system-task interfaces. In J. Bronzino, editor, *The Biomedical Engineering Handbook*. CRC Press, 1995. in press.

[12] G.V. Kondraske and G.J. Khoury. Telerobotic system performance measurement: Motivation and methods. In *Cooperative Intelligent Robotics in Space III*, volume 1829, pages 161–171. SPIE, 1992.

[13] G.V. Kondraske, R.A. Volz D.H. Johnson D. Tesar, J.C. Trinkle, and C.R. Price. Network-based infrastructure for distributed remote operations and robotics research. *IEEE Transactions on Robotics and Automation*, 9(5):702–704, October 1993.

[14] Sukhan Lee and Hahk Sung Lee. Modeling, design, and evaluation of advanced teleoperator control systems with short time delay. *IEEE Transactions on Robotics and Automation*, 9(5):607–623, October 1993.

[15] Bryan D. Morgan and Sean Graves. Extended teleautonomous control: TATP demo 3 report. Technical Report TR94-002, Texas A&M University, Dept. of Computer Science, 1994.

[16] Richard P. Paul. *Robot Manipulators: Mathematics, Programming, and Control.* MIT Press, 1981.

[17] P.M. Schilling, M.A. Skubic, and G.V. Kondraske. Shared human autonomous control: TATP1 demo 5 report. Technical Report TR 94-003R, University of Texas at Arlington, 1994.

[18] Thomas B. Sheridan. *Telerobotics, Automation, and Human Supervisory Control.* MIT Press, 1992.

[19] Matthew R. Stein and Richard P. Paul. Operator interaction, for time-delayed teleoperation, with a behavior-based controller. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, volume 1, pages 231 – 236, San Diego, CA, May 1994.

[20] J.D. Wise. Controlling multiple robots at multiple sites: A demonstration of a distributed telerobotics system integration. Technical Report 9405, Rice University, Dept. of Electrical and Computer Engineering, June, 1994.