ADJUSTABLE IMPEDANCE, FORCE FEEDBACK AND COMMAND LANGUAGE AIDS FOR TELEROBOTICS
(Parts 1-4 of an 8-Part MIT Progress Report)

Thomas B. Sheridan, G.Jagganath Raju, Forrest T. Buzan, Wael Yared and Jong Park

Man-Machine Systems Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139

This paper summarizes four separate projects recently completed or in progress at the MIT Man-Machine Systems Laboratory. Four others are described in a companion paper in Volume 3.

1. OPERATOR ADJUSTABLE IMPEDANCE IN MASTER-SLAVE TELEOPERATION -
G. Jagganath Raju

Abstract. A 2-port impedance network model of a single degree of freedom remote manipulation system in which a human operator at the master port interacts with a task object at the slave port in a remote location is presented. The design of the network involves the selection of feedback gains for the servomechanisms that transmit motion and force information from one port of the 2-port to the other in both directions. The methodology proposed here allows for this selection to be based on both stability requirements and specifications of desired port impedances, given models of the task and the human operator. The resulting design guidelines guarantee stability for any passive task object at the slave port and any passive human impedance at the master port.

Introduction. In remote manipulation tasks that use master-slave manipulators, the ability to successfully execute the task, and the performance level of the human operator are dependent on the impedance characteristics of the task object, the master-slave system, and the operator's arm. If the master- slave system were designed such that its impedance characteristics could be adjusted by the human operator while doing the task, it may be possible to improve the performance level of the operator significantly.

When humans manipulate objects in their environment, two senses that are extensively used are vision and the "muscle senses" that mediate kinesthesis and proprioception. The assumption made here is that the objective of the manipulation task is to identify and/or alter the location of an object in the environment. The term "telepresence" reflects the concept of transporting the human operator not in body but in sensation to the remote location. Though the "skin senses" may be blocked by a telemanipulator mechanism and/or the telecommunication channel, the "muscle senses" and vision may be replaced with high fidelity transmission channels of vision and force/displacement. In reality, owing to limitations imposed by the environment (distances, transmission medium) and technology (sensor resolution, transmission bandwidth, time delays) the transmission signals are degraded and have to be enhanced or compensated for in some way to be of real value.

A force feedback channel can provide the operator with values of the force levels in the interaction with the object, displayed on a screen, or better still convert these measurements back to a force level through a servo-actuator to redirect a sense of "feel" to the operator. The force transmission channel in the forward direction transmits the forces that the human operator would have imposed on the task had she been able to manipulate the object directly.

The approach adopted here is to model the MSM as a 2-port network with the operator-master arm interface designated as the master port and the task-slave arm as the slave port. The dynamics of the force transmission channels and the responses of the MSM at the master and slave ports can then be characterized by a set of "network functions". The sub-systems that comprise a model of a remote manipulation system are depicted in Fig. 1.

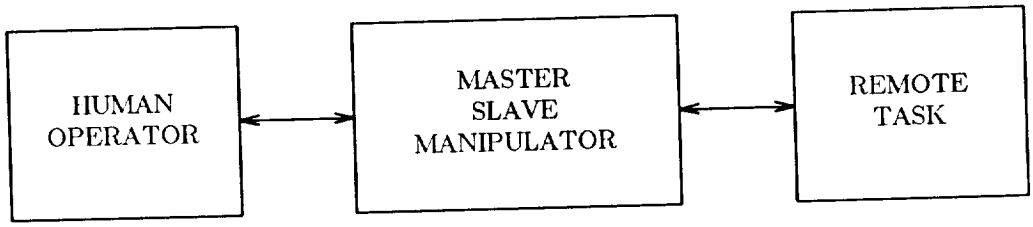

Figure 1. Remote manipulation.

The three sub-systems are: the human operator, who manipulates the master device of the MSM in a manner that results in the slave device of the MSM acting on the task to achieve the desired goal; a MSM which transmits forces and motion between the human operator and the remote task; the task object in the remote environment that is being manipulated by the slave device of the MSM.

What constitutes an "ideal" telemanipulator? Intuitively a reasonable response would be: An ideal telemanipulator is one that provides complete transparency of the interface. In other words the operator feels as if the task object were being handled directly. For force-feedback systems Handlykken [1] suggests that this can be represented by an infinitely stiff and weightless mechanical connection between the end-effector of the master arm and the slave arm.

Yokokhoji and Yoshikawa [2] argue that the ideal response of a remote manipulator system is one in which the position and force responses of the master arm are systematically equal to the responses of position and force when the operator manipulates the object directly. But from a human factors point of view, Vertut [3] suggests that the operator may sometimes get tired of holding a constant weight, and reports implementing a system with continuous variation of the force feedback ratio to reduce fatigue and improve precision. Indeed it is not all that clear that the highest level of force-feedback is universally beneficial in executing all tasks.

Hill [4] reports that in the classic peg-in-hole insertion task, the insertion phase shows little difference with or without force feedback. Bejczy and Handlykken [5] report from experimental studies that there seems to be an optimal combination of the force-feedback gain (from slave to master) and the feedforward gain (from master to slave) and that this combination may be task dependent.

It appears that as yet there is no consensus regarding an universal idealization of a remote manipulator system. Indeed to some extent the hypothesis in this work is partly motivated by the non-existence of such a universal standard, since this brings out the necessity of designing an adjustable system. An additional implication of this statement may be that the "ideal" telemanipulator is an "adjustable" one.

At the input or master port, the MSM interacts with the operator; at the output or slave port it interacts with the task object in the remote environment, as illustrated in Fig. 2.

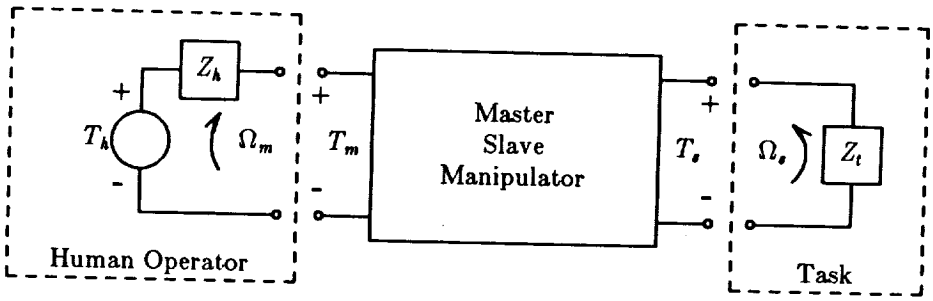

Figure 2. Electrical network model of master-slave manipulator.

Electrical network model of master-slave manipulator. At each port (master and slave) the co-energy variables of interest are the effort variables (torques T_m and T_s) and the flow variables (velocities Ω_m and Ω_s). Either of the co-energy variables at each port may be chosen to be the independent variable, and the value of the dependent variable is then determined by the parameters that characterize the MSM. If the flow variables are considered to be the inputs, the MSM can be represented by an impedance matrix [Z(s)] such that

$$\underline{T}(s) = [Z(s)]\,\underline{\Omega}(s) \qquad \underline{T}(s) = \begin{Bmatrix} T_m(s) \\ T_s(s) \end{Bmatrix} \qquad \underline{\Omega}(s) = \begin{Bmatrix} \Omega_m(s) \\ \Omega_s(s) \end{Bmatrix} \tag{0.1}$$

or

$$\begin{Bmatrix} T_m(s) \\ T_s(s) \end{Bmatrix} = \begin{bmatrix} Z_{11}(s) & Z_{12}(s) \\ Z_{21}(s) & Z_{22}(s) \end{bmatrix} \begin{Bmatrix} \Omega_m(s) \\ \Omega_s(s) \end{Bmatrix} \tag{0.2}$$

Two other representations of the MSM are obtained if one of the inputs at one port is an effort variable and the input at the other port is a flow variable, or vice-versa. These are commonly referred to as hybrid models of the 2-port. Since each of the alternative representations governs the same physical system, the elements of each matrix can be calculated in terms of the elements of one of the other representations. Two special cases of the 2-port that are relevant to our context are the *bilateral* 2-port and the *reciprocal* 2-port.

When both the off-diagonal elements of the 2-port matrix are non-zero, signal flow takes place in both directions (state changes at one port cause changes at the other) and the network is called a *bilateral* 2-port. If these two off-diagonal elements are equal the 2-port is termed *reciprocal.*

The design goals are:(1) The 2-port MSM has to be stable for any passive termination $Z_h(s)$ at the master port, and any passive termination $Z_t(s)$ at the slave port; (2) the port impedances, $Z_m(s)$ at the master port and $Z_s(s)$ at the slave port, have to match desired values specified by the designer. By combining the conditions for stability and the conditions to realize desired port stiffnesses, the constraints on selection of feedback gains for the servo-mechanisms, [K], are derived and used in the design process. A general approach for a n-port network is to require that the immitance matrix for the n-port network be positive real. This is equivalent to constraining the n-port described by the matrix [Z(s) or Y(s)] to be passive, and hence inherently stable.

The criteria that have to be satisfied by the impedance matrix [6] are: (1)$Z_{ij}(s)$ is real; (2) $Z_{ij}(s)$ is analytic in the RHS; (3) Poles of $Z_{ij}(j\omega)$ are simple, and the hermitian residue matrix at each of these poles is positive-semidefinite; (4)The determinant of the hermitian matrix of the impedance function is non-negative for all Ω.

The design was realized in hardware [7] by a one degree-of-freedom Master-Slave Manipulator. Using this apparatus, an experiment was conducted to check if the capability to adjust the impedance parameters of the MSM was gainful. From the results there are some interesting observations that can be made in a qualitative sense. The level of force-feedback that the operator feels is determined by the specification of the master port impedance. Force-feedback is useful in interacting with remote tasks up to a certain level, beyond which a performance index based on speed and success of task execution did not improve significantly. If other criteria such as operator fatigue, comfort and arm strength (a low force-feedback level would allow a weak operator to compress a stiff spring) are factored into the performance index, there is no apparent reason to believe that the highest level of force-feedback would be the best.

The experiment clearly established that the selection of the slave port impedance is dependent on the task characteristics. Hence a reasonable conclusion that can be reached is that the capability to adjust the impedances of the MSM is advantageous in executing tasks with widely differing characteristics or tasks that are made up of sub-tasks that have differing characteristics. Before contact with the task object the slave port impedance should below, and the master port impedance high, so that contact can be sensed but not result in the imposition of excessive force on the task object. Upon contact, the slave port impedance should be increased so that the task can be executed, and the master port impedance reduced to provide a comfortable but adequate level of force-feedback to the operator. The functional dependence of the impedances selected on the characteristics of the operator and the task is an area that can be explored further with the aid of design tools that have been developed in this work.

References

(1) Handlykken, M. and Turner, T., Control system analysis and synthesis for a six DOF universal, *Proc. IEEE Conf. Decision and Control*, 1980.

(2) Yokokohji, Y. and Yoshikawa, T., *Analyis of Maneuverability and Stability for Master-Slave System*, Automation Research Laboratory, Kyto University, Uji, Kyoto, Japan.

(3) Vertut, J., Fournier, R., Espiau, B. and Andre, G., Advances in a computer-aided bilateral system, *ANS Topical Meeting on Robotics and Remote Handling*, Gatlinburg, TN, 1984.

(4) Hill, J. W., *Study of Modeling and Evaluation of Remote Manipulation Tasks with Force Feedback*, SRI Project 7696 Final Report, SRI International, Menlo Park, CA, 94025, March 1979.

(5) Bejczy, A. K. and Handlykken, M., Experimental Results with a Six Degree-of-Freedom Force Reflecting Hand Controller, *Proceedings of the 17th Annual Conference on Manual Control*, UCLA, Los Angeles, CA, June 1981.

(6) Ghausi, M.S., *Principles and Design of Linear Active Circuits.* McGraw-Hill, Inc. 1965.

[7] Raju, G.J., *Operator Adjustable Impedance in Bilateral Remote Manipulation*, PhD Thesis, MIT, Cambridge, MA, Sept. 1988.

2. TELEOPERATORS WITH LARGE TIME DELAYS: A PREDICTIVE AID WITH FORCE REFLECTION - Forrest T. Buzan

Abstract. Delays in teleoperator control loops make "real time" control difficult. Operators tend to adopt a move-and-wait strategy (Ferrell, 1965, 1966). Noyes and Sheridan found that a "predictor display" which showed the kinematic response of the slave to the master commands helped the operator work faster and more continuously. This work addresses the extension of the predictor concept to include force feedback and dynamic modelling of the manipulator and the environment.

Predictive Operator Aid. If the manipulator commands are sent through a model of the manipulator system without the delay, the response of the model should predict the delayed output which we will receive from the actual manipulator. We then have the choice of using the real, delayed output or the predicted (model's) output; or we can use both in some manner.

Figure 3 shows the general system diagram for the open loop predictor (below) running in parallel with the actual manipulator (above). The delay is shown in the feedback. It can be placed in either or split between them with no loss of generality. Also, keep in mind that the forward loop gain, `K`, as well as the preceding summing junction "exist" within the human operator.

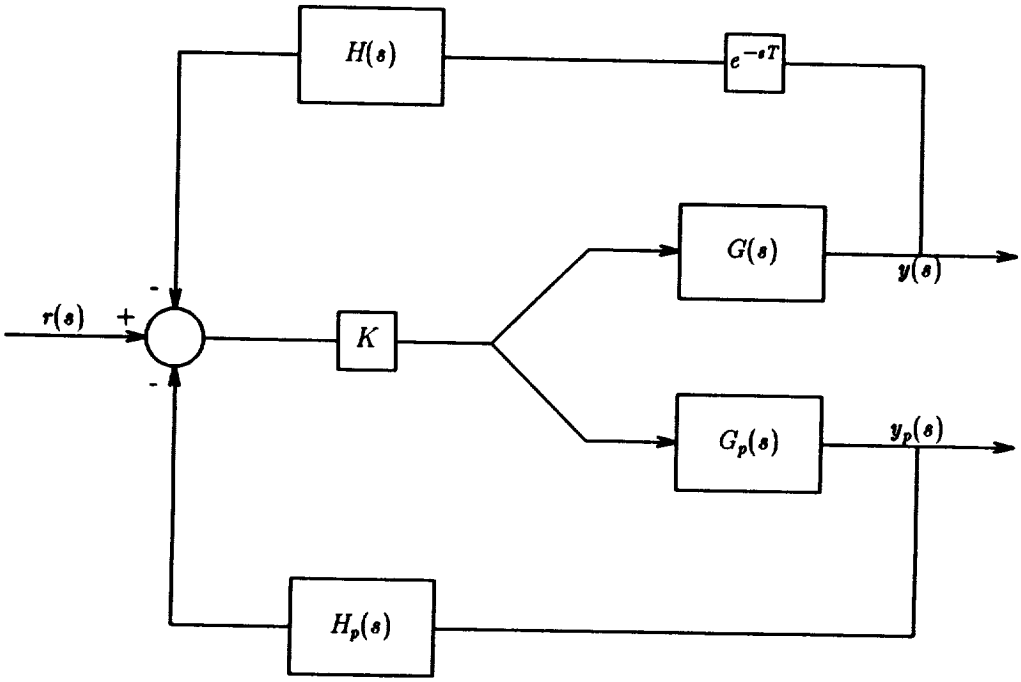

Figure 3. General system diagram for open loop predictor.

If we use only the real feedback with no attenuating filtering (H(s)=1, Hp(s)=0), we observe the standard time delayed feedback system with its well known stability problems. However, using only the predicted feedback (H(s)= 0, Hp(s)=1) we no longer have to worry about the delay induced instability, but we lose accuracy and disturbance robustness.

The predictor display has the effect of giving the operator visual feedback from both the real manipulator and the predictor. The operator then subconsciously combines them to get his "best estimate". Since the normal means of receiving force feedback is strongly coupled to the position control input, operators seem to have more trouble using delayed force feedback than delayed video. Delayed force feedback tends to look like a disturbance. We want to find the best combination of actual and/or predicted force feedback for the operator. The predictor display serves as a model.

In the most directly analogous form we provide both the delayed feedback and the predicted feedback to the operator. The predicted feedback is applied directly to the backdrive input joystick. The delayed feedback is applied to a passive joystick; this separates it more from the loop, letting the operator use it as desired. This presentation form is called "dual" force feedback.

In "complementary" force feedback we explicitly combine the two force feedbacks using complimentary low and high pass filters. The logic for this comes from the predictor display usage: for real time motion tracking the operator uses the predictor's output which seems to causally follow the commands; however, once the transient behavior has settled and the time delay has passed the operator checks the real feedback for accuracy. The low passed real force feedback provides better steady state error, but using the predicted feedback for the higher frequencies lessens the stability problem.

Experiments. Testing of operator performance using force feedback is being done using a single degree-of-freedom simulation with two backdriveable motors: one for input and direct force feedback, the other for passive, indirect force feedback. Tests compare delayed (direct and indirect), predicted, dual and complementary force feedback, and compare them to the baseline of no force feedback. Test subjects perform a series of timed trials involving two tasks: grappling of a floating mass and insertion of the mass into a slot.

References

(1) Ferrell, W.R., Remote Manipulation with Transmission Delay, *IEEE Transactions On Human Factors in Electronics*, No. 1, September 1965.
(2) Ferrell, W.R., Delayed Force Feedback, *Human Factors*, October 1966, pp. 449-455.
(3) Noyes, M. and Sheridan, T.B., A Novel Predictor for Telemanipulation through a Time Delay, *Proceedings of the 1984 Annual Conference on Manual Control*, Moffett Field, CA: NASA Ames Research Center.

3. MODELING AND COMMUNICATING OF INTENTION TO A COMPUTER - Wael Yared

Abstract. A system has been constucted to infer intent from the operator's commands and the teleoperation context, and generalize this information to interpret future commands

Issues in the design of robot command languages. This deals with a central issue confronting designers of human-computer interfaces: the modeling and communication of the human user's intent. This presupposes, of course, that the more immediate issues of defining a syntax and semantics for the interaction have been adequately dealt with. We are stressing here the distinction between the pragmatics of the interaction, on one hand, and its syntactic/semantic aspects on the other. The human user's intent relates sequences of domain actions ("plans") to changes in some model of the task environment. Its representation clearly constitutes, therefore, an issue in pragmatics.

The key words here are "change" and "environment". Interfaces such as robot command languages have only had, so far, atrophied means for the symbolic representation of change due to some agent's actions. Without such tools it is impossible to implement procedures that can reason about the relevance of an act in a plan, or that can benefit from the constraints and opportunities the environment imposes on the agent. These capabilities are, in turn, the basic building blocks for representing and exploiting intentions. A human-computer interaction that lacks these notions is artificial- not in the sense of requiring the human to learn a formal syntax, but in the much more aggravating one of lacking the purposive dimension of any human language.

The issue then is not merely one of user convenience, but of the efficiency and robustness of human-computer communication. An efficient process of task instruction is one in which few or no steps need to be repeated, modified or added when invoking that same task in a different context. A robust instruction is one that permits recovery from error when unexpected contingencies arise in the environment. To illustrate the ideas developed in the course of the present research, we use a simple command language for a 2-D cartesian manipulator as a case study. The scenario for the interaction is that of a human expert typing to the computer a description of a task in the current environment; the human is assumed to beknowledgeable and fully cooperative.

Intention recognition. The purpose of the recognition algorithm is not to check the user's beliefs and intentions against some general blueprint for plans, and then decide whether the user's plan is valid or not; that would amount to no more than a simple terminological convention. Rather, upon enumeration by the user of the actions that comprise the plan, the plan is assumed to be valid, and the system identifies in it the intended acts. Plans are primarily characterized by the intentions they comprise, and this characterization later helps in generalizing plans. The following subsections correspond to the two main layers of the algorithm: the intention recognition algorithm proper, and the plan generalization stage.

In a typical interaction, the user types to the system a sequence of primitive actions that accomplish some task in the simulated environment (e.g. retrieving a block from a warehouse row, switching blocks, etc.). As the instructions are typed, the system parses them and "interprets" them by properly directing pointers to the

corresponding executable code. At that stage the task could, in principle, be carried out by the system in the environment without any further user intervention. This is where the intention recognition routine enters the game, however. Instead of giving the user the option of having the task executed, the system proceeds to reason about the plan just defined to elicit the intentions behind it.

What the system does, in effect, is to simulate for itself the effects of that sequence of actions incrementally. To construct this chain of simulated effects, the system has to build a snapshot of its state vector and of the world model after applying each action, and update its private copy of both. Since the chain of actions has already been provided by the user, it is a simple matter to test, at each step in the plan, whether the following action is already executable or not. The system thus identifies the intended acts from the enabling acts in the plan, and stores this information for later use. Upon completion of that stage, the plan is appended to a dynamic plan library. The following points must be emphasized:

(a) The cost of this exercise is negligible: the sequence of instructions is to be entered by the user anyway, and the procedural code for the primitive actions already exists- whether intention recognition is done, or whether straightforward "dumb" plan execution is desired. Any robot command language has to include at least these, and nothing else is required to perform intention recognition. The computational cost of testing the necessary conditions for enablement is proportional to the size of the plan and thus poses no problem. Testing for the sufficient conditions could, in theory, be more problematic; I haven't looked at a worst-case scenario yet.

(b) It is not assumed that each act enumerated by the user is an intended act; rather, they are all intentional acts. The distinction is not futile, since only intended acts are the ones the user is committed to both in terms of reasoning (planning) and in guiding his or her behavior according to the intended act's success or failure.

(c) In contrast to work in planning, no action interrelationships are posited from the start. The system is given a linear, unstructured chain of acts and proceeds to infer nondeductively its corresponding structure.

(d) The intention recognition process is invisible to the user; the intention recognition routine doesn't, in any conventional sense, belong to the user-interface properly speaking, nor to the execution monitor or top-level control of a planner. Although no psychological relevance is claimed in any part of the present work, it is useful to take note of this in trying to dispel the conventional "clean" separation between "smart" planner and "dumb" top-level control.

Plan re-interpretation. A plan for a physical manipulation task is re-interpreted if that task is invoked by the user in some different context than the one in which it was originally defined. When this case arises, the plan is reevaluated and eventually modified; if modified, it is appended again to the plan library. The plan re-intepretation procedure comprises the following steps:

(a) The system locates in the plan library the most recent precedent of the plan definition, together with the latter's corresponding model of the world environment at the time of the original definition. If the plan definition includes parameters, the system performs the trivial task of substituting the current parameters for the old ones in the original plan definition. At this point the plan is re-compiled, and the system proceeds to the relevance check.

(b) In a first phase of the relevance check, the system performs a backward pass on all the subgoals of the modified plan in order to determine which of them are still relevant. A subgoal is no longer relevant if the current state of the world and the robot are identical to the ones it is there to achieve. The state of the world and of the robot a subgoal achieves is calculated by aggregating the effects of all the previous subgoals with those of the intended act of the current subgoal. Since it appears reasonable to assume that when a subgoal is irrelevant, then all its preceding siblings become irrelevant, the backward pass is stopped at the first occurrence of an irrelevant subgoal. Pointers are then redirected to ensure that the new plan only comprises the relevant subgoals.

(c) In a second phase of the relevance check, the system checks the relevance of the enabling acts for each (now relevant) subgoal. Within each subgoal, a backward pass evaluates the relevance of the enabling acts with the same mechanism that was used in the intention recognition routine. In other words, the executability of each enabling act is checked in turn (in the context of the current subgoal); when an enabling act is found executable, its preceding siblings are all no longer relevant and are discarded from the plan.

Implementation. The ideas presented in the previous sections have been implemented in an experimental robot interface called GRICE (Generalization through Recognition of Intention and Chains of Enablement). GRICE simulates a 2-D cartesian manipulator in a task environment that includes a warehouse and several blocks. The user configures the environment (places the blocks and the robot) with the mouse, or retrieves a pre-configured one from a data file. A menu then prompts the user to define a new task, retrieve a previously defined task, generalize on a previously defined task, or execute a current task.

GRICE is implemented in the C language on a Silicon Graphics IRIS 2400 system. The reasoning routines, the task execution routines, and all the data structures (about 50% of the 3000 lines of code) are totally machine-independent and domain-indepedent. Some of the domain-dependent routines (the repertoire of robot

primitives, but not the command interpreter) and the graphic simulation draw heavily on the IRIS graphics library and graphics-dedicated hardware.

Data structures for plans (both source and binary versions), world model and robot state vector are all implemented as standard doubly-linked lists, which eases their traversal in both directions. An object-oriented programming style allows the reasoning and task-execution routines to be independent of domain of application; in C, this is implemented using pointers to functions.

4. COMMAND LANGUAGE FOR OBSTACLE AVOIDANCE - Jong Park

Abstract. The purpose of this research is to design a command language system that is robust, easy to learn, and has more natural man-machine communication. A general telerobot problem selected as an important command language context is finding a collision-free path for a robot.

Background and needs. A complete task-level robot command language is long overdue, where goals for the position of objects are to be specified, rather than the motions of the robot needed to achieve these goals. Available techniques, such as the configuration-space approach of Lozano-Perez, are promising, but finding a solution takes too long to be used on-line.

In order to overcome this difficulty, and in keeping with the notions of "supervisory control", it is suggested that a human operator should play a more active role by making global decisions, letting the computer deal with the details. In the context of collision avoidance the human operator should decide how to maneuver to avoid collision, then let the robot find if his commands will work. Such a command language should be reliable, with no misinterpretation of commands, should be easy and natural for the operator to use, and should be easy to learn.

Command language. In the command language being designed, the operator types natural-language-like commands to a keyboard (a symbolic input) and simultaneously may use a joystick to indicate geometric direction or magnitude (an analogic input). He may also move a cursor to point (to an object or attributes of an object such as surfaces, edges or vertices on a video or computer-graphic screen--easier than calling their names) in coordination with a symbolic action command. Interpretation by the computer in this case will depend upon the viewpoint of the operator.

Both syntactics of the command language (e.g., the mathematical equations or logical rules constraining what is admissible) and the semantics (specification of motions and locations of objects and their attributes relative to the environment) are important. In Figure 4, for example, it may be more natural to say "A approach B until x away " than "A go y along normal to B", since the constraint "x" is likely to be more relevant to what matters to the operator than the actual distance travelled. Specifying "Go 10 inches parallel to and 1 inch away from (this) edge in (this) direction" (edge and direction indicated by translating and orienting cursor) is likely to be easier than saying "Go (87,35,67) to (93, 64, 24)" after determining the precise starting and ending coordinates. Usually the operator needs to know and specify some variables with reasonable precision, though other variables can assume a wide range. If the computer has other context information it can even make sense of a command like "Go 3 above (this) surface".

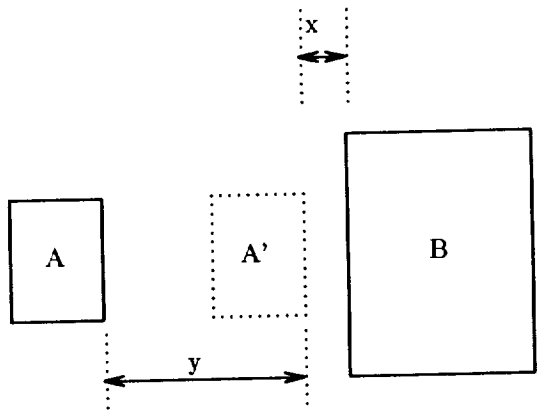

Figure 4. A approaches B.

Hardware and software. The system hardware consists of three input devices (keyboard, mouse, 6 degree-of-freedom joystick) and graphics workstation. The software consists of six modules as shown in Figure 5: natural language interface (NLI), indication interpreter (II), general interpreter (GI), guidance handler (GH), graphic simulator (GS), and executor (EX). NLI is based upon the already developed "augmented transition network" capable of interpreting limited vocabulary and grammer. Typed natural-language-like commands are interpreted by NLI. Voice commands can be added later. II works with NLI to interpret the meaning of the operator's mouse-cursor commands. GI takes input from NLI and II and determines robot motions accordingly. GH handles inputs from the 6 DOF joystick and also moves the robot. GS checks if commanded motions are free of collision, and if so EX is called to execute the motions.

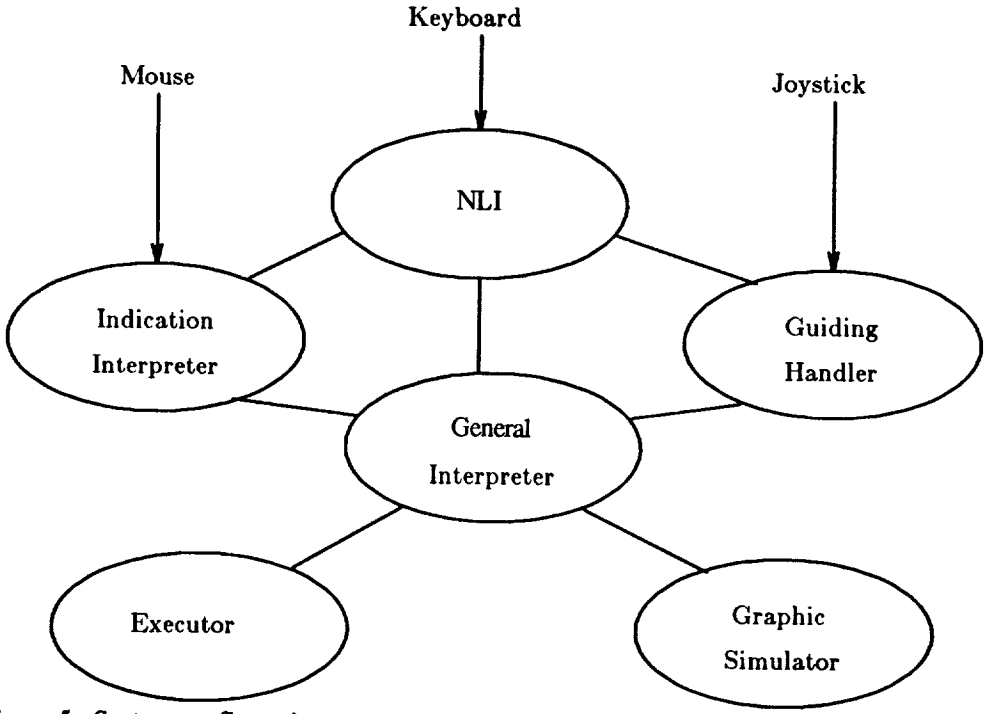

Figure 5. System configuration.

Experiments. Experimental subjects are being asked to perform a telerobotic obstacle avoidance task to evaluate the effectiveness of the command language.

ACKNOWLEDGMENT. Projects 1, 2 and 4 were sponsored by Jet Propulsion Laboratory, Project 3 by NASA Ames Research Center. Their support is gratefully acknowledged.