Robotics & Computer-Integrated Manufacturing, Vol. 6, No. 4, pp. 303-308, 1989
Printed in Great Britain

0736-5845/89 $3.00 + 0.00
Pergamon Press plc

● *Paper*

THE NASREM ROBOT CONTROL SYSTEM STANDARD

RONALD LUMIA, JOHN FIALA and ALBERT WAVERING

Intelligent Controls Group, National Institute of Standards and Technology, U.S.A.

The problem of robot control is approached from a systems standpoint where a complete control system must include all of the aspects involved in moving a robot, not just the algorithms in the classic controls literature. The NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM) provides the framework for a complete manipulator control system. It is composed of three hierarchies: task decomposition, world modeling, and sensory processing. The task decomposition hierarchy divides the task into smaller and smaller subtasks. In order to achieve the desired decomposition, the task decomposition hierarchy must often access information stored in the world modeling hierarchy, which contains a workspace representation, object descriptions, robot models, etc. The sensory processing hierarchy constantly fills the world model with processed sensor information. The NASREM functional architecture was developed for NASA's Flight Telerobotic Servicer, a two-armed robot which will build and maintain the Space Station. However, the control concepts proposed by the NASREM functional architecture immediately transfer to other applications such as in manufacturing, autonomous vehicles, mining vehicles, etc.

1. INTRODUCTION

In spite of the fact that research in robotics has been progressing for many years, there are surprisingly few systems which can be used to compare different algorithms experimentally. There has not been a major ground swell to develop testbeds although most researchers seem to agree that this is probably the only viable method for comparing two approaches. Instead, researchers have tended to promote one particular approach at the expense of alternatives blaming the problem on efficiency, computer resources, manpower limitations, and myriad other excuses.

The problem of the scarcity of testbeds is further exacerbated by the "institutional biases" injected into solutions. If the engineers like Company X microcomputers, it is most unlikely that other alternative computers have a chance of being chosen unless it is obvious from the start that Company X microcomputers cannot do the job. This is true for any institutional bias: expert systems, blackboards, whiteboards, hierarchical control. Nevertheless, the scientific approach compels the scientist to conduct an unbiased exploration of the alternatives.

There are many approaches to controlling robots.[1-8] One would hope that it is possible to develop a robot control system that can serve as a sophisticated robot controller as well as a testbed. Built on nearly 10 years of work in an automated factory environment,[9] the third-generation NBS controller (with the institution bias of hierarchical control), NASREM, is under development. It was conceived to bridge the gap between the algorithm centered approaches and the testbed concept. The primary contribution revolves around the proper definition of the levels in the hierarchy and the careful specification of interfaces so that the vast amount of literature in robotics is supported.

NASREM was originally conceived as the standard for the control system for the Flight Telerobotic Servicer so that the system could evolve with technology with minimal impact on the hardware and software. The NASREM functional architecture is suited for other applications such as manufacturing. This paper describes the purpose and overall organization of NASREM. Then, two examples of NASREM task decomposition modules are discussed in terms of function as well as interface requirements.

2. NASA/NBS STANDARD REFERENCE MODEL FOR TELEROBOT CONTROL SYSTEM ARCHITECTURE (NASREM)

One way to view a robot control system is as a two-level hierarchy.[10] The upper level is concerned with the robot actions which are task dependent but robot independent while the lower level is concerned with the robot actions which are task independent but robot dependent. If each of these levels is examined

The original version of this paper was presented at the 2nd International Symposium on Robotics and Manufacturing (ISRAM), Albuquerque, New Mexico 16-18 November 1988. The published proceedings of this meeting may be ordered from: CAD Laboratory for Systems/Robotics, EECE Dept., UNM, Albuquerque, NM 87131, U.S.A.

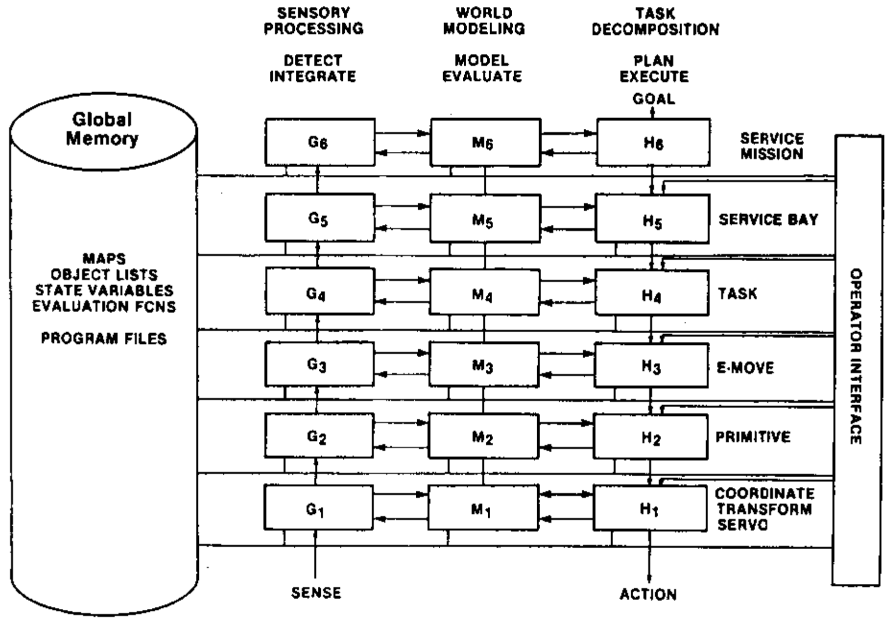

Fig. 1. NASA|NBS Standard Reference Model.

more closely, more resolution can be obtained, resulting in the fundamental paradigm of the NASREM shown in Fig. 1. The control system architecture is a three-legged hierarchy of computing modules, serviced by a communications system and a global data system.[11] The task decomposition modules perform real-time planning and task monitoring functions; they decompose task goals both spatially and temporally. The sensory processing modules filter, correlate, detect, and integrate sensory information over both space and time in order to recognize and measure patterns, features, objects, events, and relationships in the external world. The world modeling modules answer queries, make predictions, and compute evaluation functions on the state space defined by the information stored in a global data system. The global data system is a database which contains the system's best estimate of the state of the external world. The world modeling modules keep the global data system current and consistent.

2.1 *Task decomposition (plan, execute)*

The first leg of the hierarchy consists of task decomposition modules which plan and execute the decomposition of high level goals into low level actions. Task decomposition involves both a temporal decomposition (into sequential actions along the time line) and a spatial decomposition (into concurrent actions by different subsystems). Each task decomposition module at each level of the hierarchy consists of a Job Assignment Manager, a set of planners, and a set of Executors. These decompose the input task into both spatially and temporally distinct subtasks as shown in Fig. 2.

The control system architecture described here is a six-level heirarchy. At each level in this hierarchy a basic transformation is performed on the goal. Each level of the hierarchy has a fundamental philosophy which describes its behavior. These simple descriptions of the purpose of each level help the designer in organizing algorithms into the correct place in the control hierarchy.

Servo Level performs motions which are small in a dynamic sense.

Primitive Level (Prim) performs motions which are large in a dynamic sense.

Elemental Move Level (E-Move) transforms goals described from a task point of view (currently in a geometric fashion) into goals from a manipulator point of view.

Task Level is the "man-equivalent level". Goals are planned based on a geometric description of the world, incorporating "common sense" physics.

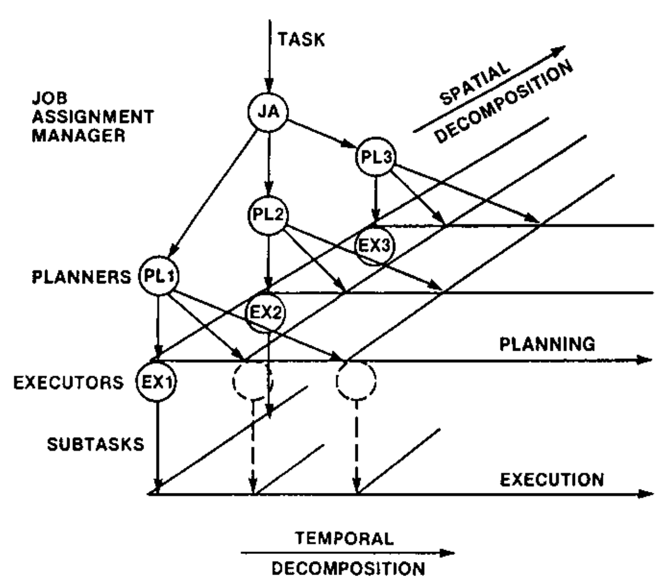

Fig. 2. Task decomposition.

Service Bay Level coordinates groups of task level robots.
Mission Level sets priorities for the activities.

2.2 *World modeling (remember, estimate, predict, evaluate)*

The second leg of the hierarchy consists of world modeling modules which model (i.e. remember, estimate, predict) and evaluate the state of the world. The "world model" is the system's best estimate and evaluation of the history, current state, and possible future states of the world, including the states of the system being controlled. The "world model" includes both the world modeling modules and a knowledge base stored in a global data system where state variables, maps, lists of objects and events, and attributes of objects and events are maintained. By this definition, the world model corresponds to what is widely known throughout the artificial intelligence community as a "blackboard".[12] The world model performs the following functions:

1. Maintain the global data system knowledge base by accepting information from the sensory system.
2. Provide predictions of expected sensory input to the corresponding sensory processing modules, based on the state of the task and estimates of the external world.
3. Answer "What is?" questions asked by the executors in the corresponding level task decomposition modules. The task executor can request the values of any system variable.
4. Answer "What if?" questions asked by the planners in the corresponding level task decomposition modules. The world modeling modules predict the results of hypothesized actions.

2.3 *Sensory processing (filter, integrate, detect, measure)*

The sensory processing hierarchy modules recognize patterns, detect events, filter and integrate sensory information over space and time, and report this information to the world model to keep it in registration with the external world. At each level, sensory processing modules compare world model predictions with sensory observations and compute correlation and difference functions. These are integrated over time and space so as to fuse sensory information from multiple sources over extended time intervals. The sensory processing modules also contain functions which can compute confidence factors and probabilities of recognized events, and statistical estimates of stochastic state variable values.

2.4 *Operator interfaces (control, observe, human I/O)*

The control architecture has an operator interface at each level in the hierarchy. The operator interface provides a means by which human operators, either in the space station or on the ground, can observe and supervise the telerobot. Each level of the task decomposition hierachy provides an interface where the human operator can assume control. The task commands into any level can be derived either from the higher level task decomposition module, or from the operator interface. Using a variety of input devices such as a joystick, mouse, trackball, light pen, keyboard, voice input, etc., a human operator can enter the control hierarchy at any level, at any time of his choosing, to monitor a process, to insert information, to interrupt automatic operation and take control of the task being performed, or to apply human intelligence to sensory processing or world modeling functions.

The sharing of command input between human and autonomous control need not be all or none. It is possible in many cases for the human and the automatic controllers to simultaneously share control of a telerobot system. For example a human might control the position of the robot's end effector while the robot automatically maintains the wrist orientation.

3. SERVO LEVEL TASK DECOMPOSITION MODULE

The Servo Level task decomposition module for controlling a robotic manipulator is described in this section. Servo is responsible for controlling small dynamic motions of the manipulator. Large motions, i.e. trajectories, are obtained by concatenating these small motions as described in the next section.

Figure 3 shows the basic structure of Servo task decomposition. Also depicted are the interfaces to the module from Primitive and Operator Control. Servo can be commanded by Primitive task decomposition (autonomous mode), by the operator through joysticks or master arms (teleoperated mode), or by a combination of Primitive and operator inputs (shared control model). It is the task of the Job Assignment module to produce a coordinated output from the two input sources. The output of Job Assignment commands the Planner module for the manipulator. The Planner feeds periodic data points to the Executor module. The data points are used by the Executor as attractors for the manipulator state. That is, the Executor module cyclically computes control signals for the actuators of the telerobot based on the difference between the current state and the desired state given by each Planner data point. Through this technique, the manipulator is moved along the desired trajectory.

The Primitive input to Servo consists of several parameters which will be described briefly here. The parameter C_z indicates the servo coordinate system. The options for C_z include joint coordinates (Cartesian) world coordinates, and (Cartesian) end effector coordinates. The attractor set for the manipulator, formed by the vectors z_d, $\dot{z}_d$, $\ddot{z}_d$, z_d, f_d, and $\dot{f}_d$, gives the desired position, velocity, acceleration, jerk, force and force rate for the manipulator. These vectors are in Servo coordinates.

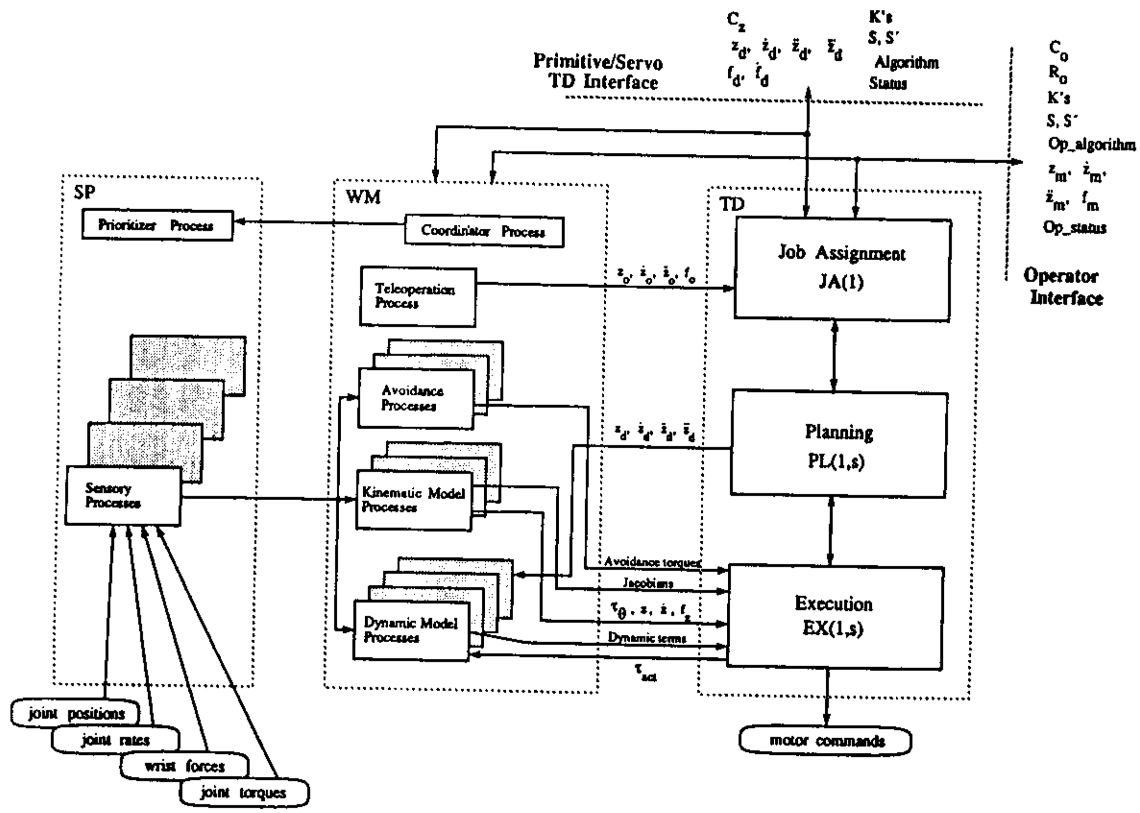

Fig. 3. Servo Level task decomposition.

The Ks are the gain coefficients which multiply the error vectors in the control equations. The parameters S and S' are selection matrices used to select different control modes for different axes of the servo coordinate system. The Servo-algorithm selects the specific control algorithm to be used by Servo in approaching the given attractor. The Status parameter informs Primitive of the status of the Servo module.

A similar interface exists for the operator command input. In this interface, the parameter C_0 specifies the servo coordinates desired by the operator. The parameter R_0 indicates how redundancy resolution is to be performed during shared control when C_0 is underspecified with respect to C_z. The attractor set from Operator Control is $\{z_m, \dot{z}_m, \ddot{z}_m, f_m\}$. The K's and S's from Operator Control are similar to the parameters found in the Primitive command. The Op-algorithm selects the algorithm desired by the operator. This parameter also determines overall control mode, i.e. teleoperated, autonomous, or shared. Op-status returns status to Operator Control.

The interfaces for Servo task decomposition allow a large variety of servo control algorithms to be used with the architecture. A large number of examples from the literature are detailed in Ref. 13.

4. PRIMITIVE LEVEL TASK DECOMPOSITION MODULE

As stated previously, the Primitive Level task decomposition module determines manipulator behavior for motions which are "large in a dynamic sense". The function of the module is to transform a static description of a desired motion into a time sequence of closely spaced Servo goal points, or attractors.

There are a number of different ways in which motions may be specified in a time-dependent manner. For example, the desired end effector path may be specified as a function of a single parameter which indicates the fraction of the path traveled. Desired forces along a path may also be commanded for constrained motions. Instead of a path specification, it may be more appropriate to simply indicate the desired direction of movement, along with some conditions which indicate when the motion should be terminated. This type of motion specification is useful for generalized damper motions, for example. A third class of motions consists of those for which the important goal is the final state, and the exact path to be followed in achieving the desired goal state is impossible to specify *a priori*. An example of this type of motion command is a vision-servoed move to attain a position relative to a moving object.

The structure and interfaces of Primitive task decomposition (Prim) are shown in Fig. 4. Prim receives commands from the Elemental Move Level (E-Move) task decomposition module, or from the Operator Control. The Input command interface includes provisions for specifying the trajectory generation algorithm, the coordinate system of the position and/or force command descriptions, the names of the objects being manipulated, and the termination conditions for the motion. Important factors for Prim to take into

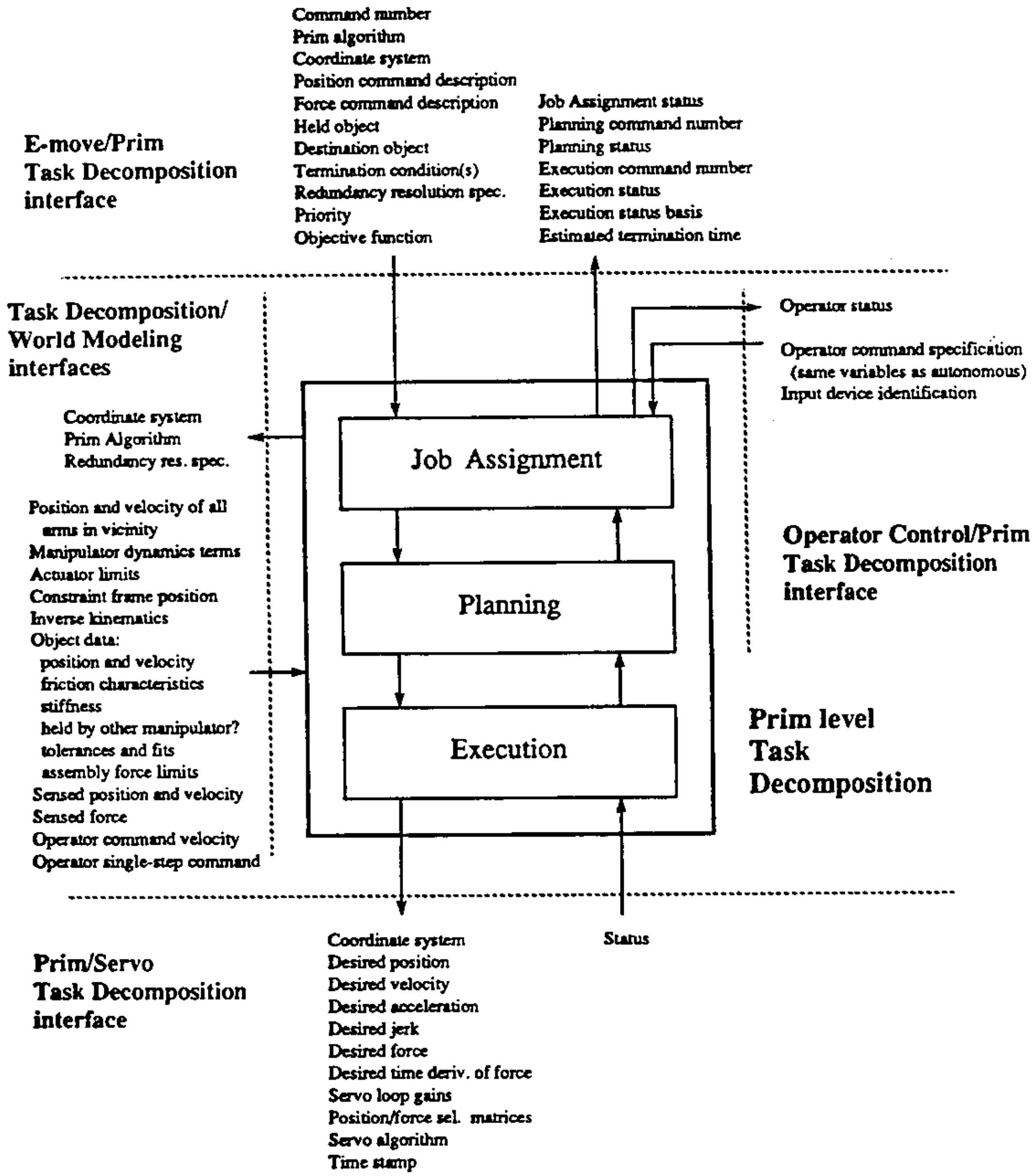

Fig. 4. Primitive Level task decomposition.

account in generating the trajectory are indicated in an objective function to be minimized, and a redundancy resolution specification is included to direct the use of redundant manipulator degrees of freedom. A priority is given to the manipulator for the motion to help resolve conflicts over shared workspace during trajectory execution.

In common with the other task decomposition modules, Primitive consists of Job Assignment, Planner, and Executor submodules. These submodules are cyclically executing processes which together perform the trajectory generation functions of Prim. The Prim Job Assignment module manages the queue of input commands from E-Move and the operator, and coordinates the transition between autonomous and manual operation. An input command queue is necessary at the Prim level so that smooth transitions between consecutive path segments may be planned.

The Prim Planner performs trajectory planning according to the algorithm and parameters specified in the input command. For motion along a desired path segment, the Planner determines time functions of manipulator position, velocity and acceleration to perform the path. In doing so, the Planner must take into account such factors as manipulator and payload dynamics actuator limitations, and allowable path error. The Planner also determines trajectory parameters to use with sensory-interactive and other non-preplanned movements, and Servo feedback gains to achieve appropriate manipulator impedance.

The Prim Executor module computes the small intermediate motions which are commanded to Servo to execute a trajectory. The Executor module does this by evaluating the planned trajectory functions, or by executing the proper sensory-interactive trajectory algorithm. The Executor module also monitors sensor states to determine when termination conditions have been achieved, and performs several functions to ensure that the commands to Servo will be valid.

The interfaces to Prim, and the operation of the Prim Job Assignment, Planner, and Executor modules for a number of different trajectory generation algorithms are discussed further in Ref. 14.

CONCLUSION

This paper has described the NASREM architecture in terms of its purpose and overall organization. It was shown through the examples and two of the task decomposition modules that the NASREM concept is sufficiently flexible to act as both a robot controller and a testbed for robot control algorithms. The NASREM standard can be used to control any manipulator. Consequently, although initially developed for space applications, the NASREM architecture can be used as the standard for other applications such as in manufacturing.

REFERENCES

1. Albus, J. S.: Control concepts for industrial robots in an automated factory. *SME Technical Paper MS77–745*, 1977.
2. Brooks, R. A.: A robust layered control system for a mobile robot. *IEEE J. Robotics Automation* **2-1**: 1986.
3. Byler, E., Peterson, J.: High performance architecture for robot control, *Proc. of Workshop on Space Telerobotics*, Pasadena, 1987.
4. Crowley, J. L.: Navigation for an intelligent mobile robot. *IEEE J. Robotics Automation* **1**-1: 1985.
5. Hayward, V., Paul, R. P.: Introduction to RCCL—a robot control C library. *IEEE Conference Robotics and Automation*, Atlanta, 1984.
6. Meystel, A.: Nested hierarchical controller with partial autonomy. *Proc. of Workshop on Space Telerobotics*, Pasadena, 1987.
7. Saridis, G. N.: Intelligent robotic control. *IEEE Trans. Automatic Control* **28**: 1983.
8. Schenker, A., French, R., Sirota, A.: The NASA/JPL telerobot testbed: An evolvable system demonstration. *IEEE Int. Conf. on Robotics and Automation*, 1987.
9. Barbera, A. J., Albus, J. S., Fitzgerald, M. L.: Hierarchical control of robots using microcomputers. *9th Int Symp. on Industrial Robots*, 1979.
10. Paul, R. P.; Problems and research issues associated with the hybrid control of force and displacement. *Proc. of Workshop on Space Telerobotics*, 1987.
11. Albus, J. S., Lumia, R., McCain, H. G.: NASA/NBS Standard Reference Model For Telerobot Control System Architecture (NASREM). NBS Technical Note 1235, Also available as NASA document SS-GSFC-0027.
12. Barr, A., Feigenbaum, E.: *The Handbook of Artificial Intelligence*. Los Altos, William Kaufman, 1981.
13. Fiala, J. C.: Manipulator Servo level task decomposition. Technical Note, Robot Systems Division, National Bureau of Standards, Gaithersburg, MD, April, 1988.
14. Wavering, A. J.: Manipulator primitive level task decomposition. Technical Note, Robot Systems Division, National Bureau of Standards, Gaithersburg, MD, May, 1988.