JPL Publication 89-7, Vol. V

# Proceedings of the NASA Conference on Space Telerobotics

## Volume V

G. Rodriguez
H. Seraji
Editors

January 31, 1989

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 89-7, Vol. V

# Proceedings of the NASA Conference on Space Telerobotics

## Volume V

G. Rodriguez
H. Seraji
Editors

January 31, 1989

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

This publication was prepared by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

# ABSTRACT
## NASA Conference on Space Telerobotics

These proceedings contain papers presented at the NASA Conference on Space Telerobotics held in Pasadena, January 31-February 2, 1989. The Conference was sponsored by the NASA Office of Aeronautics and Space Technology, together with ARC, LRC, GSFC, JSC, MSFC, KSC and JPL. The theme of the Conference was man-machine collaboration in space. The Conference provided a forum for researchers and engineers to exchange ideas on the research and development required for application of telerobotics technology to the space systems planned for the 1990s and beyond. The Conference: (i) provided a view of current NASA telerobotic research and development; (ii) stimulated technical exchange on man-machine systems, manipulator control, machine sensing, machine intelligence, concurrent computation, and system architectures; and (iii) identified important unsolved problems of current interest which can be dealt with by future research. There were about 500 international participants including about 100 from abroad.

An international program committee was established for the conference. A.K. Bejczy and H. Seraji of JPL acted as co-chairs for this committee. Members of the committee were

J. Amat, University of Barcelona, Spain
G.A. Bekey, University of Southern California
P.R. Belanger, McGill University, Canada
R.C. Bolles, Stanford Research Center
J.G. Bollinger, University of Wisconsin
W.J. Book, Georgia Institute of Technology
J.M. Brady, Oxford University, UK
F.E.C. Culick, California Institute of Technology
R.J.P. deFigueiredo, Rice University
W.R. Ferrell, University of Arizona
E. Freund, University of Dortmund, FRG
A.A. Goldenberg, University of Toronto, Canada
R. Jain, University of Michigan
T. Kanade, Carnegie-Mellon University
I. Kato, Waseda University, Japan
A.J. Koivo, Purdue University
P.D. Lawrence, University of British Columbia
J.Y.S. Luh, Clemson University
H.E. Rauch, Lockheed Palo Alto Research Lab
A. Rovetta, Polytechnic University of Milan
G.N. Saridis, Rensselaer Polytechnic Institute
T.B. Sheridan, Massachusetts Institute of Technology
L. Stark, University of California, Berkeley
D. Tesar, University of Texas at Austin
H. Van Brussel, Catholic University of Leuven
R.A. Volz, Texas Tech University

The Conference was organized by the Telerobotics Working Group of the NASA Office of Aeronautics and Space Technology. M. Montemerlo of NASA Headquarters and S.Z. Szirmay co-chair this working group. Representatives to this group from NASA centers and other research organizations are

D. Akin, Massachusetts Institute of Technology
J. Bull, Ames Research Center
R. Davis, Kennedy Space Center
S. Fisher, Ames Research Center
J. Haussler, Marshall Space Flight Center
A. Meintel, Langley Research Center
J. Pennington, Langley Research Center
D. Provost, Goddard Space Flight Center
C. Price, Johnson Space Center
L. Purves, Goddard Space Flight Center
C. Ruoff, Jet Propulsion Laboratory
E.C. Smith, Marshall Space Flight Center
M. Zweben, Ames Research Center

# ACKNOWLEDGMENTS

Acknowledgments are due to R. Doshi, D. Diner, J. Barhen and A. Fijany of JPL and Professors L. Stark of UCB and H. Stephanou of George Mason University for their help in organizing invited technical sessions and discussion panels. Appreciation is due to P. McLane for setting up registration and conference facilities, and to L. Anderson for technical editing of the proceedings. Acknowledgments are also due Donna L. Milton of JPL for handling the local arrangements and coordination and administrative aspects of the Conference.

CONTENTS

Volume I

TELEROBOT ARCHITECTURES . . . . . . . . . . . . . . . . . . . . . . . 139

ROBOT SENSING AND PLANNING . . . . . . . . . . . . . . . . . . . . . . 217

VOLUME II

VOLUME III

VOLUME IV

VOLUME V

APPENDIX A

APPENDIX B

APPENDIX C

# PLENARY SESSION

Page intentionally left blank

# TELEROBOTIC ACTIVITIES AT JOHNSON SPACE CENTER

Charles R. Price

Chief, Teleoperator Systems Branch
Johnson Space Center
Houston, Texas 77058

## ABSTRACT

In today's expanding and exciting field of telerobotics, the Johnson Space Center continues to fulfil its responsibilities for technical leadership in the development and operations of the manned spacecraft for the United States. The JSC telerobotics efforts span three major thrusts: Sustaining and expanding the capability of the Shuttle manipulator, developing and integrating the multiple telerobotic systems of the Space Station, and fostering and applying research in all areas of telerobotics technology within the government, private, and academic sectors.

## 1. Introduction

Engineering development and mission operations of the manned spacecraft of the United States is the responsibility of Johnson Space Center. To continue the achievements of Mercury, Gemini, Apollo, Skylab, Apollo-Soyuz, and Space Shuttle, JSC currently maintains a labor force of 3400 civil servants and 5600 contractors, on 1600 acres in Houston, Texas, containing an acquired facility base of $ 810 million (book value) and operated at an annual budget of $ 750 million. The primary function of JSC is to put humans into space, meet the objectives of each mission, and to return the crews safely to earth. The general approach used at JSC is to derive "top down" the requirements and design of the spacecraft and crew procedures to meet high level program objectives while also fostering and applying technology developments in a "bottom up" manner to enhance system performance and safety.

Telerobotics activities at JSC have been ongoing for fifteen years and follow the top down / bottom up philosophy. At present, about two hundred employees are active in telerobotics projects throughout JSC which address all aspects of this multifaceted emerging discipline. Major programs supported include Space Shuttle, Space Station, Flight Telerobot Servicer, Office of Aeronautics and Space Technology, Small Business Innovative Research, several university programs, Office of Space Commercialization, and the JSC Director's discretionary fund.

## 2. Telerobotics Activites for the Space Shuttle

For Space Shuttle, JSC continues to refine the Payload Deployment and Retrieval System (PDRS) which consists of the Remote Manipulator System and its ancillary equipment: The Manipulator Positioning Mechanism, closed circuit television, lights, grapple fixtures on the payloads, the Shuttle general purpose computer, and the Shuttle Orbiter vehicle itself. The PDRS has performed successfully on all of its eighteen flights, both for its nominal functions and also as a resource for innovative solutions for problems arising in realtime.

The PDRS is the operational state of the art for space teleoperators. Its capabilities includes the grapple, transport, orientation, and release of payloads (from the payload bay) and conversely, the track, capture, grapple, transport, orientation, and berthing of satellites (into the payload bay). Additionally, it can provide transport, positioning, and orientation of EVA (Extra Vehicular Activity) crew via a mobile foot restraint attachment, local illumination via mounted lights, local directional television, positioning of enviromental sensing payloads into the orbital freestream, and power and data interface services to grappled payloads. Finally, the PDRS has been used as a resource for creative solutions to unexpected problems arising during flights such as force closure of a recalcitrant folding antenna, breaking ice from a vent nozzle, and use as a "flyswatter" to flip a toggle switch on an unresponsive satellite.

JSC provided the technical management and integration for the development of the PDRS and continues this function for the sustained use of the PDRS. For each mission application, JSC conducts non-realtime and realtime, crew in the loop simulations and analyses, mission planning and design, flight and ground procedures development, and mission control. JSC provides the requirements definition, integration, and flight worthiness verification of all functional upgrades and obsolescence-driven upgrades. JSC also sponsors, co-sponsors, and implements flight experimentation using the PDRS. For example, JSC is designing and implementing the on orbit Dexterous Manipulation Demonstration manifested for February, 1992, which will demonstrate force torque feedback, a magnetic end effector, and a boresighted closed circuit television end effector / payload alignment aid.

## 3. Telerobotics Activities for the Space Station

The Space Station has baselined multiple telerobotic flight systems from several different sources. These systems include 1) the Mobile Service Center which consists of the Canadian provided Mobile Remote Servicer and Special Purpose Dexterous Manipulator and the JSC provided Mobile Transporter, 2) the Goddard Spaceflight Center (GSFC) provided Flight Telerobot

Servicer, and 3) the JSC provided Assembly Work Platform. Furthermore, JSC is responsible for the delivery of the Multipurpose Application Console which is the crew workstation for control of all electrically activated flight systems (of which the telerobotics systems are a subset). JSC has also proposed the development of a free flying EVA crew support, voice commanded robot based on the successful ground demonstration of the EVA Retriever.

For Space Station, JSC is defining the functional and performance requirements for the Mobile Service Center in general and the JSC deliverable, the Mobile Transporter, in particular. These requirements are being based largely on the non-realtime and realtime simulations activities at JSC. These simulation facilities will be also used to verify the performance of flight components and systems as they are developed and integrated for flight. Major, innovative upgrades in these simulators and tools are being brought online now and will be used throughout the development, verification, and operation of the Space Station.

For the Flight Telerobot Servicer, JSC provided onsite at GSFC participation in the Skunkworks design concept effort, the in-house Phase B activity, and currently the Phase C/D source evaluation board. JSC is also currently providing to GSFC engineering support in the form of Space Station assembly tasks definition, EVA crew interface requirements, analyses on crew workstation concepts for both Orbiter and Space Station, robotic task definitions, communications systems analysis, and integration support for both Space Shuttle and Space Station.

The on-orbit assembly of the Space Station is a major challenge to define and implement and is heavily dependent upon the Space Shuttle. JSC's experience base in Shuttle is being applied to this problem in the definition of upgrades to the PDRS, displays and controls, communications, and other Orbiter systems that are required for proper support of the Space Station assembly flights.

## 4. Telerobotics Technology Activities at JSC

As an active member of the NASA's Telerobotics Intercenter Working Group that is sponsored by the Office of Aeronautics and Space Technology (OAST), JSC is pursuing the tranfser and application of telerobotics research from other NASA centers as well as conducting technology development itself. These efforts are fostered through OAST, Small Business Innovative Research, several university programs, Office of Space Commercialization, Space Shuttle flight experiments, Space Station advanced technology, and the JSC Director's Discretionary Fund.

Telerobotic research at JSC spans all five component disciplines as defined by the Intercenter Working Group:

Sensing and perception, planning and reasoning, control execution and mechanisms, crew interface, and system architecture and integration. In sensing and perception, JSC is pursuing intelligent ranging and tracking sensors, tactile sensing, and image processing; in planning and reasoning JSC s developing trajectory planners that include provisions for moving object avoidance and reactive planners for the accomodation of systen faults; in control execution and mechanisms, JSC has control efforts in hierarchical and parallel control laws, kinematic redundancy, smart hands, closed loop force torque control, and robot friendly components'; in crew interface JSC is pursuing flat panel color displays, advanced hand controllers, control and monitoring of multiple manipulators with total degrees of freedom of fifty or more, presentation of hidden workspace to operators; and in system architecture and integration JSC is pursuing evaulations of component and system fault tolerance through like and unlike redundancy patterned after the Shuttle avionics fault tolerant design.

A major three year design and integration telerobotic technology effort that is currently mid way complete is the JSC EVA Retriever. This inhouse project is a ground demonstration of a voice commanded, supervisory controlled robot that flies a space qualified Manned Manuevering Unit on a precision air bearing floor and has repeatably demonstrated the real time discernment of objects, path planning, rendezvous, grappling of a chosen object, and the return to home base. In its final configuration, EVA Retriever will be able to operate among moving objects while avoiding collision.

## 5. Telerobotics Facilities at JSC

JSC has a full spectrum of technical discipline laboratories that test, evaluate, and develop generic space technologies such as sensors, computing elements, human factors standards, controls, and actuators which have application to telerobotics. JSC also has the Advanced System Development Laboratory that prototypes proof of concept integrated telerobotic systems.

Flight crew interaction with concepts and design occurs at all levels but is especially emphasized in the full scale simulators. The real time, Systems Engineering Simulator (SES) is a high fidelity dynamics, multiple mission phase design tool that can simulate on orbit Shuttle, Space Station, Manned Manuevering Unit, and Orbital Manuevering Vehicle concurrent environments for crew and system dynamic interaction. The SES is bringing online a leading edge computer graphics system update to its out the window scene generation system that will provide high fidelity lighting and luminance modelling capabilities that will be critical to the viewing and cueing by the flight crew of the telerobotic construction and operations.

In contrast, the Manipulator Development Facility provides a full scale physical Shuttle payload bay and hydraulically operated PDRS that provides contact forces of grappling and berthing payloads. The Shuttle Mission Simulator is used for PDRS procedures and protocol design and checkout between the Shuttle crew and the mission controllers and experiment investigators. The Weightless Environment Training Facility provides a neutral bouyancy for interaction among EVA suited crew, the PDRS, and payloads. Finally, the Shuttle Avionics Integration Laboratory provides a ground instantiation of the Shuttle flight system that continues to perform the verification of all changes to the Shuttle software and avionics hardware. The functionality of these Shuttle dedicated facilities will be reproduced in Space Station equivalent facilities.

## 6. Concluding Remarks

JSC is experienced and active in the development and operations of putting humans into space, having them achieve mission objectives, and returning them safely to Earth. JSC also has an extensive base in the development and operation of the world's only space teleoperator, the Shuttle Payload Deployment and Retrieval System. JSC also is very active in the pursuit of telerobotic technology development and the transfer and application of telerobotic technologies developed by other NASA centers, academia, and the private sector. NASA's commitment to the Space Station offers major challenges and opportunities in the development, integration, on-orbit assembly and operation of multiple telerobotics systems. Our next step is to achieve humans and robots, together working productively in space.

Let's do it !

(For a copy of the charts and color photographs used for this paper in the JSC Plenary Session, contact the author at (713) 483-1523 )

# ROBOT ARM MODELING AND CONTROL

# APPLICATION OF RECURSIVE MANIPULATOR DYNAMICS TO HYBRID SOFTWARE/HARDWARE SIMULATION

Christopher J. Hill
Lockheed Engineering Services Company, Houston, Texas
Kenneth A. Hopping
Boeing Electronics Company, Bellevue, Washington
Charles R. Price
NASA, Johnson Space Center, Houston, Texas

## ABSTRACT

Computer simulations of robotic mechanisms have traditionally solved the dynamic equations of motion for an N degree-of-freedom manipulator by formulating an N dimensional matrix equation combining the accelerations and torques (forces) for all joints. This paper describes the use of an alternative formulation that is strictly recursive. The dynamic solution proceeds on a joint by joint basis, so it is possible to perform inverse dynamics at arbitrary joints. The dynamics formulation is generalized with respect to both rotational and translational joints, and is also directly extendable to branched manipulator chains.

This paper describes a hardware substitution test in which a servo drive motor was integrated with a simulated manipulator arm. The form of the dynamics equation permits calculation of acceleration given torque or vice-versa. Computing torque as a function of acceleration is required for the hybrid software/hardware simulation test described. For this test, a joint servo motor is controlled in conjunction with the simulation, and the dynamic torque on the servo motor is provided by a load motor on a common driveshaft.

## INTRODUCTION

The Manipulator Emulator Testbed (MET) is a simulation facility designed to support concept studies, evaluation and other engineering development activities for a variety of manipulator configurations. In particular, the testbed is intended to support development of simulations of the Space Station Freedom Remote Manipulator System and related systems.

One of the problems faced by the users of simulators for a space robot is that the models used to simulate the behavior of the robot do not always simulate the real robot perfectly. It is desirable during model development to have manipulator components and subject them to realistic loading to assist in verification of the simulations. One goal of the MET is to provide a facility for comparing models with actual hardware component performance. The test described was developed to demonstrate the feasibility of using a software simulation to provide a realistic environment while controlling a real servo motor.

The first implementation of this concept involved attaching the MET to a motor test bed.

## TEST ARM CONFIGURATION

The present test was devised to demonstrate the capability of integrating a real motor with a simulated arm. A simple configuration for developing this capability is a two-link planar arm with rotational joints. A two-link arm is the minimum configuration that will show link interaction effects.

The arm used for the testing is depicted in Figure 1. The motor substitution is performed on the base joint, so the outboard joint is always simulated.

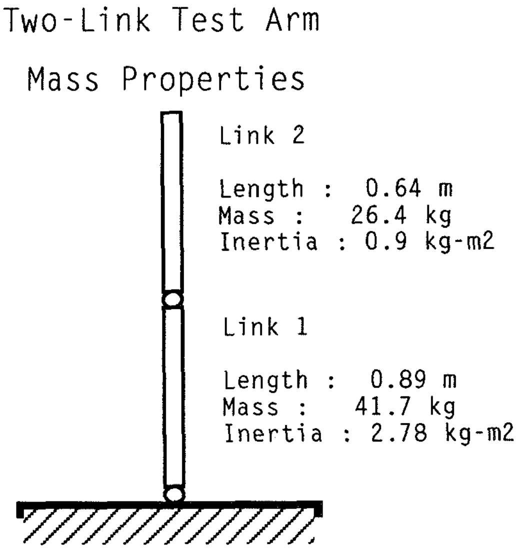


Figure 1

The test case used for the tests described was to start the arm in a "straight out" configuration, as shown in Figure 1, with initial rate of zero. The servos were commanded to produce a joint rate of 0.03 rad/sec.

## MOTOR TEST BED

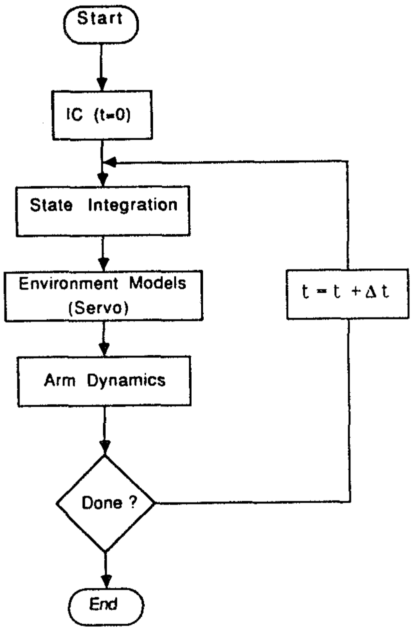
The motor test bed includes two small DC servo motors mounted on a common shaft. These motors are referred to as the "drive" motor and the "load" motor. The drive motor is the motor that simulates the joint servo motor on the physical arm. The load motor provides a load on the drive motor that emulates the load that would be "felt" by that motor in a real arm.

The motor test bed also includes an analog interface board mounted in the host computer, and power and signal conditioning amplifiers. The motors are driven by independent linear amplifiers. The load motor amplifier is set up as a current-controlled amplifier where the output current (and therefore shaft torque) is proportional to the control voltage. The drive motor amplifier is voltage-controlled.

The motor shaft rotation rate is read and fed back to the controlling computer. The shaft rate passes through a second-order low-pass filter to minimize noise. It may be desirable to provide other feedback, in particular, shaft acceleration, but this capability is not currently provided in the testbed.

## THE MANIPULATOR EMULATOR TESTBED SIMULATION

The Manipulator Emulator Testbed (MET) is a generic manipulator simulation designed to be modular and expandable. A high-level flowchart describing the MET simulation is presented in Figure 2.



The MET Simulation

Figure 2

The Initial Conditions (IC) preprocessor used in the MET uses a syntax much like the "C" Programming Language preprocessor. Use of the preprocessor allows the user to tailor the input form to the database describing the arm being analyzed.

The integration scheme used is the Modified Euler method.

A recursive rigid-link arm dynamics model (G. Nasser) was developed for use in the MET.

"Environment" models can include servo models, plume impingment models, Coriolis models and other external influences on the arm dynamics. The only environment model used for this testing is the Servo model. The servo model takes the joint state and joint rate commands as input and produces either applied joint torque or joint acceleration as output.

For this testing, the MET was configured to run on a single PC/AT, although parallel computation configurations are also available.

## INVERSE DYNAMICS

One of the features of the recursive dynamics used is the capability of performing inverse dynamics at a particular joint. The motor substitution test apparatus feeds back motor shaft rate to the simulation. This rate is differentiated numerically to obtain shaft acceleration. The inverse dynamics is used to link this shaft acceleration with the rest of the arm dynamics. At the substituted joint, the joint torque is in essence computed as a function of the arm configuration and acceleration, rather than the inverse as is normally done.

Nasser's basic equation for link dynamics is:

$$L_i U_i = F_i \qquad (1)$$

where:

$$U_i = \begin{pmatrix} U_i^R \\ \ddot{\theta}_i \end{pmatrix} \qquad (2)$$

If instead we define:

$$U_i = \begin{pmatrix} U_i^R \\ \tau_i \end{pmatrix} \qquad (3)$$

then:

$$L_i = I_{6x6}$$

and:

$$U_i = F_i = A^*_{i,i-1} \begin{bmatrix} \vartheta_{i,i-1} \\ \omega_{i,i-1} \end{bmatrix} + B^*_{i,i-1} \qquad (4)$$

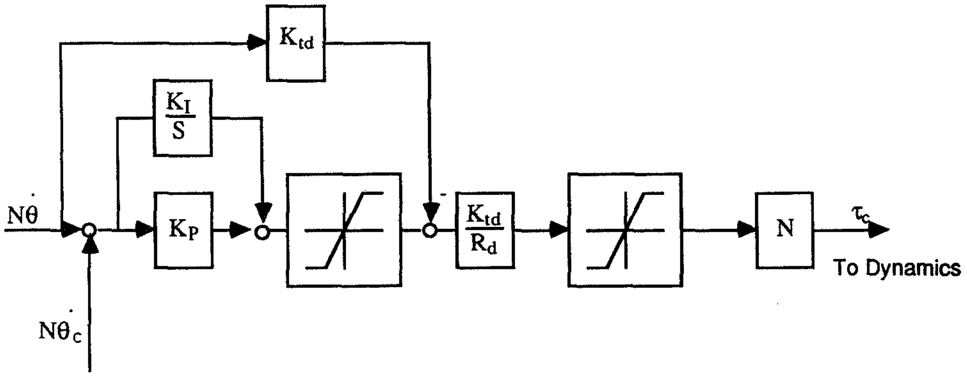
This relation is used in the Motor Substitution Test to provide the load that would be imposed on the joint drive motor by the arm, and to use this load to command the load motor.

## RIGID GEARBOX SERVO

In the interest of keeping computational requirements for this testing to a minimum, a simple servo model was selected. The name "Rigid Gearbox" arose to distingush this model from the compliant gearboxes used in analyses of the Space Shuttles' Remote Manipulator System. The Rigid Gearbox servo model consists of a proportional-integral servo controller, a dc motor with internal resistance, a torque constant and back-emf. The voltage applied to the drive motor and the torque output have limits applied.

The torque on the motor output shaft is multiplied by the gear ratio and supplied to the dynamics.

A block diagram of the Rigid Gearbox Servo is depicted in Figure 3. The values used in the model are listed in Table 1.



Rigid Gearbox Servo

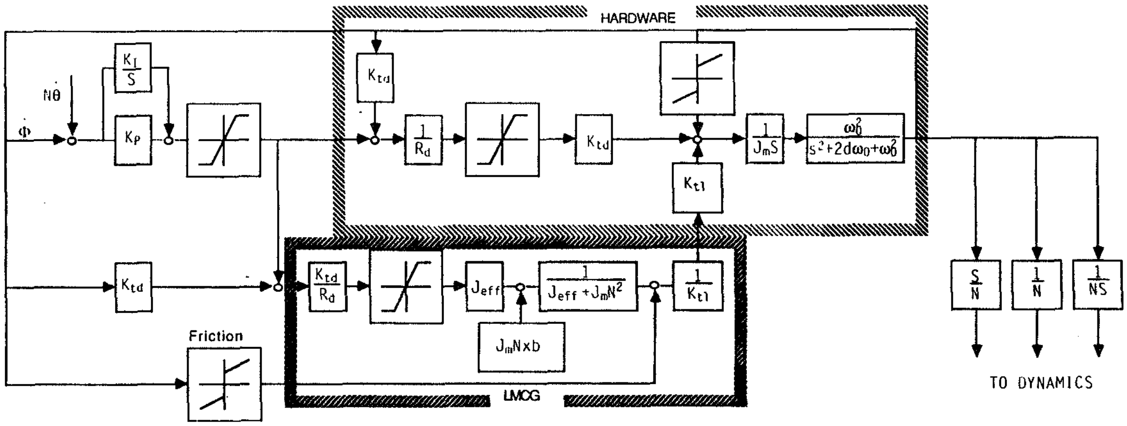
Figure 3

TABLE 1

| | | |
|---|---|---|
| N | Gearbox Ratio | 1570 |
| Kp | Proportional Error Gain | 0.020 V/rad/s |
| Ki | Integral Error Gain | 0.125 V/rad |
| Vlim | Voltage Limit (Both Motors) | 20.0 V |
| Rd | Drive Motor Resistance | 2.05 Ohm |
| Ktd | Drive Motor Torque Const. | 0.061 N-m/Amp |
| Tlim | Torque Limit | 0.144 N-m |
| Jm | Combined Motor Shaft Inertia | 6.38e-5 N-m-m |
| Wo | Cut-off Frequency | 50 Hz. |
| Ktl | Load Motor Torque Constant | 0.072 N-m/A |
| | Coulomb Friction Coefficient | 0.0205 N-m |
| | Viscous Friction Coefficient | 2.58e-5 N-m |

## MOTOR SUBSTITUTION SERVO

The motor substitution servo is designed to behave similarly to the rigid gearbox servo, while incorporating the effects of the dynamics into the load motor. A block diagram of the motor substitution servo is presented in Figure 4. The parameters used are listed in Tables 1 and 2.



Motor Substitution Servo

Figure 4

Both the drive motor and the load motor are driven in this model. The term "drive motor" is used to denote the replaced servo motor, and "load motor" denotes the motor used to apply the equivalent arm load onto the drive motor.

A proportional-integral controller identical to that used in the rigid servo model is used to provide voltage commands to the drive motor. The upper dashed-outline block of Figure 4 represents both motors, the common shaft joining them, their amplifiers and the motor rate filter.

The torque constant, armature resistance, and friction values of both motors were experimentally determined. The friction model used is a combined Coulomb and viscous model, which represents the behavior of the motors fairly well. Both motors are considered with a single set of friction values, rather than being considered separately. The amplifier used on the load motor accepts a current command, so the armature resistance of this motor was not determined.

The load motor command generator computes the load that is applied to the joint drive servo.

Dynamics of motor shaft:

$$J_m \ddot{\phi}_m = \tau_d + \tau_l \tag{5}$$

where:

$\tau_d$ = drive motor torque

$\tau_l$ = load motor torque

Rigid Gearbox:

$$\ddot{\phi} = N \ddot{\theta} \tag{6}$$

Noting that $\left(A^*_{i,i-1}\right)$ elements 0,6 thru 5,6 are 0, we define:

$$J_{eff} = -\left(A^*_{i,i-1}\right)_{66} \tag{7}$$

and:

$$b = -\left(B^*_{i,i-1}\right)_6 \tag{8}$$

Load torque:

$$N\tau_l = -J_{eff}\ddot{\theta} - b \tag{9}$$

Solving eqns (5) and (9), we obtain:

$$\tau_l = \frac{-1}{N^2 J_m + J_{eff}}\left(J_{eff}\tau_d + J_m N b\right) \tag{10}$$

The motor torque divided by the combined motor inertia gives actual motor shaft acceleration. The analog tachometer is used to read shaft rate, which passes through a second-order low-pass filter. The filtered shaft rate is integrated to determine motor shaft position, and differentiated to determine motor shaft acceleration. The position, velocity and acceleration are then divided by the gearbox ratio and fed directly into the arm state

The servo runs at a higher execution frequency than the arm dynamics. Generally, the servo is run at 100. Hz while the arm dynamics are updated at 25 Hz.

## MOTOR SUBSTITUTION SIMULATION

The motor substitution simulation was developed to test the concepts used for the motor substitution servo. The hardware components of the motor substitution servo are simulated in software. The filter is simulated using a second-order Butterworth filter.

## RESULTS

Several plots are presented showing joint rate response of the simulated test arm and the substitution arm in Figures 5-8.

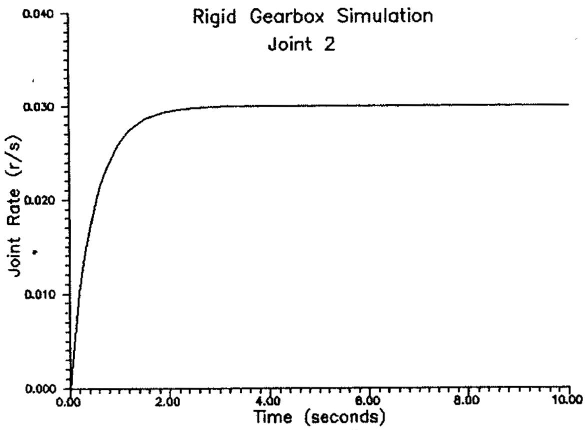


Figure 5

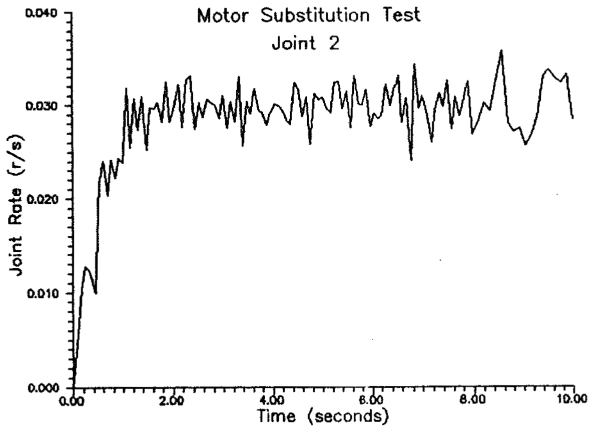


Figure 7

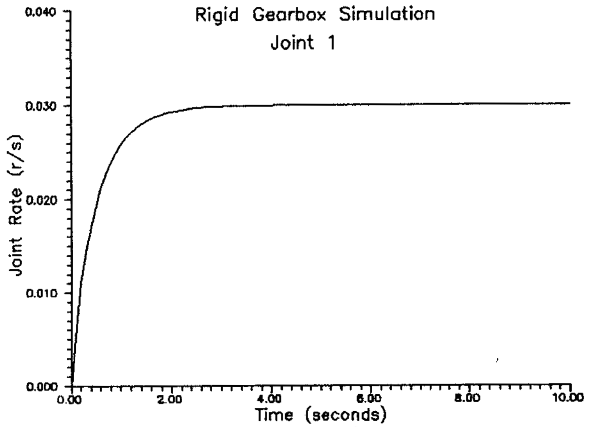


Figure 6

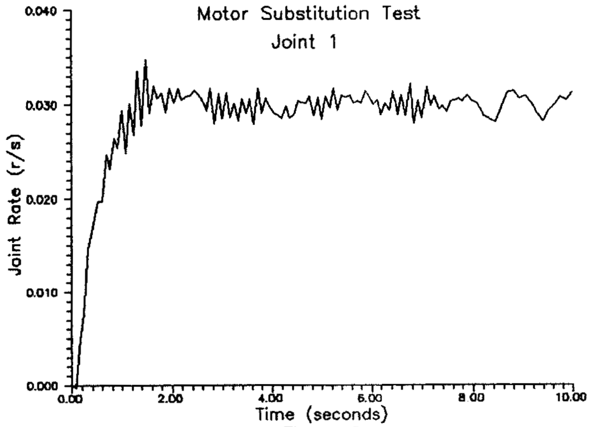


Figure 8

Figures 5 and 6 show the response of the first and second joint in the "pure" simulation configuration. Figures 7 and 8 show the comparable data for the hybrid simulation case, with Joint 1 substituted, and Joint 2 simulated, as before.

In general, there is good agreement between the simulation response and the response of the hardware substitution data. There is some noise-induced oscillation apparent in the hardware substitution plots. Sources of the noise include mis-alignment of the motor shafts, unevenness of the torque with rotation, and rate sensor noise. Oscillations in the first (hardware) joint excite oscillations in the second (software) joint, as expected.

## CONCLUSIONS

One of the more troublesome aspects of the testing described was the use of numerical differentiation, which is highly susceptible to high-frequency noise. In future tests of this type, it would be desirable to use rotational accelerometers to measure shaft acceleration directly. It is planned to use faster computing hardware in future testing. This should allow the use of 6- or 7-jointed arms, and should allow for performing motor shaft dynamics at a significantly higher frequency. For this facility to be useful for Space Station arm simulation, it is anticipated that the servo loop will be required to run in 1 or 2 milliseconds, or approximately 5 to 10 times faster than is currently possible.

Most significant, though, is that this test demonstrates that it is possible to interface hardware with a simulation. The authors believe that this capability will significantly enhance our ability to accurately simulate the behavior of space robots.

## ACKNOWLEDGMENTS

The authors would like to acknowledge the support and valuable assistance of John Chladek, Donald Barron and Carl Adams of NASA/JSC. Richard Theobald and Benjamin Bourgeois (LESC) also provided invaluable assistance in developing servo models, as well as guidance and support. Karl Zimmer (NASA), who developed the hardware testbed described, deserves special acknowlegment, as does George Nasser (LESC), who developed the dynamics model used.

## REFERENCES

1. Nasser, G., "A Recursive Newton-Euler Formulation of Manipulator Dynamics", NASA Conf. on Space Telerobotics, Jan 89, Pasadena, Calif.

2. Hopping, K. A., and Widjaja, D., "Manipulator Emulator Test Bed Geometry Model", Lockheed-EMSCO paper for NASA / Systems Development and Simulation Division, Engineering Directorate, January 1988.

# KINEMATICS & CONTROL ALGORITHM DEVELOPMENT AND SIMULATION FOR A REDUNDANT TWO-ARM ROBOTIC MANIPULATOR SYSTEM

Michael P. Hennessey, Paul C. Huang, and Charles T. Bunnell
Advanced Systems Center
FMC Corporation
1200 South Second Street
Minneapolis, MN 55459-0043

**Abstract**

An efficient approach to cartesian motion and force control of a 7 degree of freedom (dof) manipulator is presented. It is based on extending the active stiffness controller to the 7 dof case in general and use of an efficient version of the gradient projection technique for solving the inverse kinematics problem. Cooperative control is achieved through appropriate configuration of individual manipulator controllers. In addition, other aspects of trajectory generation using standard techniques are integrated into the controller. The method is then applied to a specific manipulator of interest (Robotics Research T-710). Simulation of the kinematics, dynamics, and control are provided in the context of several scenarios; one pertaining to a noncontact pick and place operation, one relating to contour following where contact is made between the manipulator and the environment, and one pertaining to cooperative control.

## 1 Introduction

Cartesian motion and force control of a manipulator is needed for many different types of automation applications such as material handling and assembly [1]. Because of the complexity of some potential applications (e.g. in space and in certain military environments) and because of the inherent limitations of many 6 dof manipulators (e.g. singularity problems), 7 (or more) dof manipulators are being proposed for these applications. As a result, there are many interesting and challenging problems, particularly with respect to kinematics, control algorithms, and controller implementation aspects. Kinematic problems stem largely from two sources: (1) the inverse positional kinematics solution is not unique, and (2) it typically does not exist in closed form. As a consequence of the nonclosed form issue, control is complicated from the standpoint that highly modular approaches may not be viable (e.g. use of individual joint position servos). Also, despite the continual increase in performance and decrease in cost of controller hardware, algorithm efficiency is still an issue. Below, the focus is on efficient motion and force control of a 7 dof articulated manipulator.

For the 7 dof manipulator case, few kinematic configurations permit closed form inverse positional (in general "location") kinematic solutions. For the remaining cases, other approaches must be taken, such as use numerical iteration to solve for the inverse positional kinematic solution or knowing the inverse Jacobian, servo at the cartesian level. Baker and Wampler [2] refer to all kinematic inversion methods as either "global" or "local" methods. In both cases, the redundancy can be used to optimize a criterion of interest. With respect to the first approach, convergence and computational efficiency can be a serious issue and perhaps somewhat suprisingly, it may not always be necessary to calculate the joint angles corresponding to a particular end effector location, hence obviating the need to solve the inverse positional kinematics problem. The second approach avoids the difficulties associated with inverting the positional kinematics, but requires a different controller implementation. A recent technique in this catagory is the gradient projection technique [2,3].

Many different control algorithms have been proposed for motion and force control of a manipulator, including: (1) hybrid control, (2) modified hybrid control, (3) active stiffness control, (4) modified resolved acceleration control, (5) impedance control, (6) operational space control, (7) free joint control, and (8) modified free joint control [4]. At first glance it appears as though there are many radically different control schemes, when in fact there are not [5]. All of the above control laws fit into roughly only two catagories: (1) "hybrid" control and (2) active stiffness control. However, at a practical level both types of controllers can perform hybrid type control depending on how the control system is configured. In hybrid control, position and force are controlled in basically orthogonal directions, and in stiffness control the nominal position is controlled and its endpoint stiffness specified.

Based on the above discussion there are many different approaches for solving the 7 dof manipulator kinematics and control problem. In the interests of stability coupled with a desire to minimize computational requirements (for implementation reasons primarily), the basic approach taken in this paper is to combine the active stiffness controller with an efficient version of the gradient projection technique. Also, since its origination, the active stiffness controller (or a slight variation of it) has been experimentally verified to work on several nonredundant manipulators and is well known for being computationally efficient (e.g. in [6]). By appropriately configuring individual manipulator controllers, cooperative control can be achieved.

# 2 General Formulation

The general block diagram of the system showing the active stiffness controller extended to 7 dof is depicted in Figure 1. The controller consists of the following major elements: (1) location feedback loop, (2) joint rate feedback loop, (3) force feedback loop, (4) torque compensation, (5) calculation of location error, (6) direct kinematics, (7) gravity compensation, and (8) trajectory generator. The location (position and orientation) feedback loop determines the manipulator's effective stiffness and differs from the original formulation in that the servoing is done at the cartesian level because of the nonclosed form inverse positional kinematics issue. To determine the cartesian feedback contribution we observe that the location feedback given by [6] is:

$$K_\theta \delta\underline{\theta} \equiv (J^T K J)\delta\underline{\theta} \equiv (J^T K)(J\delta\underline{\theta}) \equiv (J^T K)\delta\underline{X} \tag{1}$$

so that $J^T K$ is the cartesian location feedback. Here $J$ is the manipulator Jacobian of the end effector location expressed in world coordinates, $K$ is the diagonal arm stiffness matrix, $\delta\theta$ is the joint error vector, and $\delta\underline{X}$ is the resulting cartesian error vector expressed in world coordinates. We note that the feedback from Equation (1) ignores the nullspace of the Jacobian. The joint rate feedback loop (needed for stability purposes), the force feedback loop, and the torque compensation loop characterized by $G$ which is especially useful for contact situations, follow essentially as before [6]. Calculation of the location error is defined by $\Delta$, the 4x4 homogeneous error transformation given by:

$$\Delta = T_c T_a^{-1} = \begin{bmatrix} 1 & -\delta z & \delta y & \Delta x \\ \delta z & 1 & -\delta x & \Delta y \\ -\delta y & \delta x & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where $T_c$ and $T_a$ are the commanded and actual manipulator transformations and $\delta\underline{X}^T = [\Delta x, \Delta y, \Delta z, \delta x, \delta y, \delta z]$. We note that $\Delta$ must be in world coordinates and that average values for $\delta x, \delta y$, and $\delta z$ may be derived as implied by Equation (2). Finally, the direct kinematics, gravity compensation, and the trajectory generator will be developed in conjunction with application to a particular manipulator.

# 3 Application to a Particular 7 DOF Manipulator

The general formulation will now be applied to the Robotics Research T-710 manipulator [7].

## 3.1 Direct Kinematics

The location of the 7th joint's coordinate frame in base coordinates ( $T^7_{base}$) may be expressed as the product of successive 4x4 homogeneous transformation matrices ($A^i_{i-1}$) and is given by [8,9]:

$$T^7_{base} = A^1_{base} A^2_1 A^3_2 A^4_3 A^5_4 A^6_5 A^7_6 \tag{3}$$

where:

$$A^i_{i-1} = \begin{bmatrix} P_i & \underline{r}_i \\ \underline{0} & 1 \end{bmatrix} \tag{4}$$

with $P_i$ and $\underline{r}_i$ given recursively by:

$$P_i = P_{i-1}P_{i-1,i} = P_{i-1}P^{Ref}_{i-1,i}P^{\theta_i}_{i-1,i} = P_{i-1}P^{Ref}_{i-1,i}\begin{bmatrix} c_i & 0 & s_i \\ 0 & 1 & 0 \\ -s_i & 0 & c_i \end{bmatrix} \tag{5}$$

and:

$$\underline{r}_i = \underline{r}_{i-1} + P_{i-1}\underline{h}_{i-1,i} \tag{6}$$

Here $c_i \equiv \cos\theta_i, s_i \equiv \sin\theta_i$, $\underline{h}_{i-1,i}$ is the vector from the origin of the $i-1th$ coordinate frame to the $ith$ coordinate frame expressed in the $i-1th$ coordinate frame, and $\underline{r}_i$ denotes the position vector of joint $i$ in world coordinates with

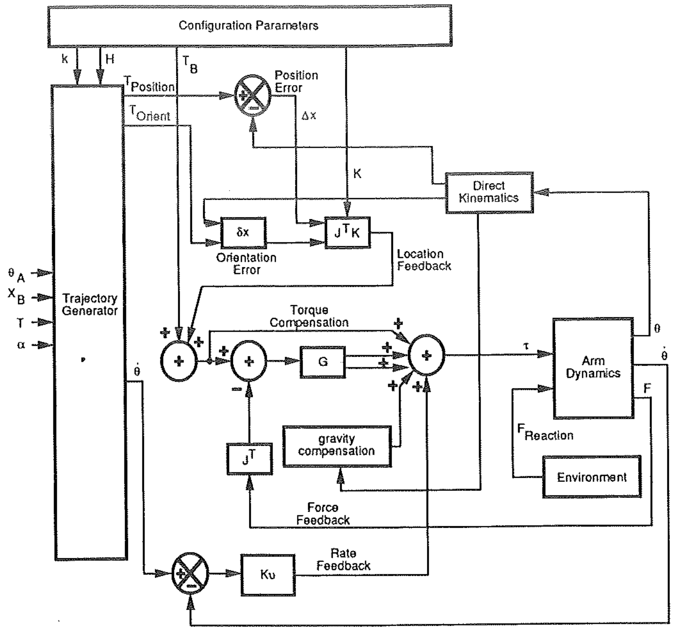


Figure 1 Block diagram of manipulator controller and system

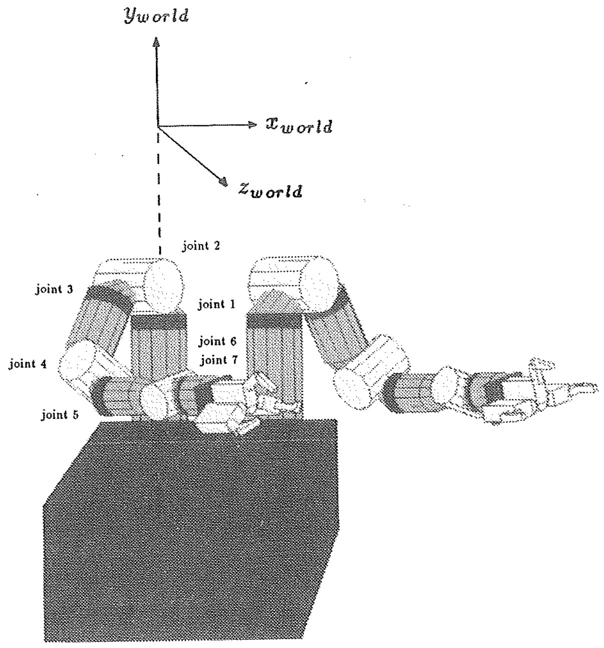
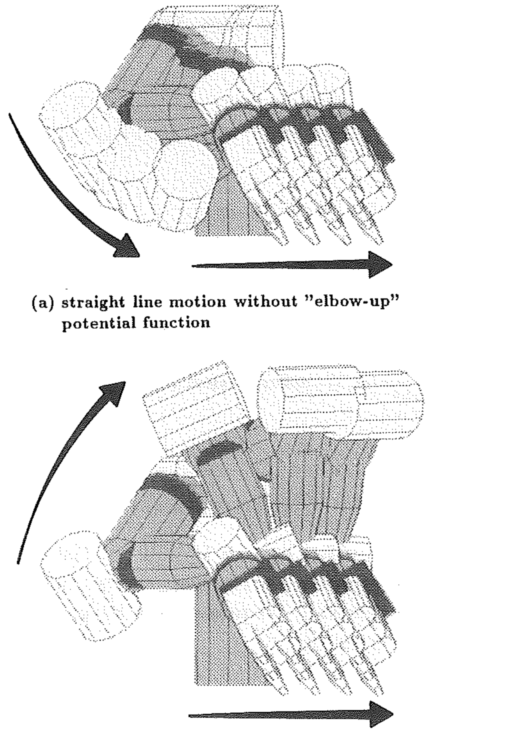


Figure 2(a) Kinematic configuration of several Robotic Research T-710 manipulators



(a) straight line motion without "elbow-up" potential function

(b) straight line motion with "elbow-up" potential function

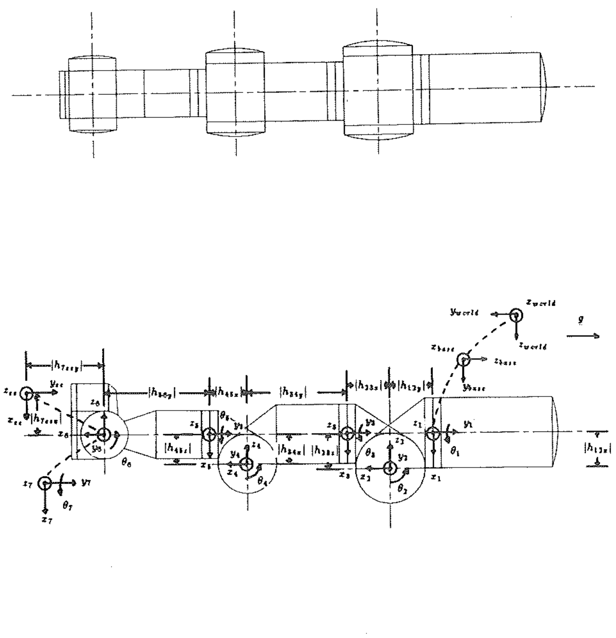


Figure 2(b) Direct kinematics of Robotic Research T-710 manipulator

Figure 3 Illustration of gradient projection technique using "elbow-up" potential function

representative kinematic data (i.e. $P^{Ref}_{i-1,i}, \underline{h}_{i-1,i}$) given in the Appendix. For the Robotics Research T-710 manipulator shown schematically in Figure 2 the total manipulator transformation $T^{7}_{base}$ may be shown to be given by (excluding the base and end effector transformations):

$$T^{7}_{base} = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where the positional entries are given by:

$$\begin{aligned} p_x = & -(c_1 s_3 + s_1 c_2 c_3)(h_{34x} + h_{45x} s_4 - h_{45z} c_4 - h_{56y} s_4) \\ & + s_1 s_2 (h_{34y} - h_{45x} c_4 - h_{45z} s_4 + h_{56y} c_4) \\ & - h_{23x} s_1 s_2 + h_{23z} s_1 c_2 - h_{12x} s_1 \end{aligned}$$

$$\begin{aligned} p_y = & (-s_1 s_3 + c_1 c_2 c_3)(h_{34x} + h_{45x} s_4 - h_{45z} c_4 - h_{56y} s_4) \\ & - c_1 s_2 (h_{34y} - h_{45x} c_4 - h_{45z} s_4 + h_{56y} c_4) \\ & + h_{23x} c_1 s_2 - h_{23z} c_1 c_2 + h_{12x} c_1 \end{aligned} \tag{8}$$

$$\begin{aligned} p_z = & s_2 c_3 (h_{34x} + h_{45x} s_4 - h_{45z} c_4 - h_{56y} s_4) \\ & + c_2 (h_{34y} - h_{45x} c_4 - h_{45z} s_4 + h_{56y} c_4) \\ & - h_{23x} c_2 - h_{23z} s_2 + h_{12y} \end{aligned}$$

We note that Equation (8) represents three equations in four unknowns or the mechanical decoupling of the major and minor portion of the manipulator linkage. While not proven here, because of certain offsets (i.e. $\underline{h}_{i-1,i}$) these equations can not be solved in closed form. With $T^{7}_{base}$ defined from above we can include the base and end effector transformations. For now, let:

$$A^{base}_{world} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

and:

$$A^{ee}_{7} = \begin{bmatrix} 1 & 0 & 0 & h_{7eex} \\ 0 & 1 & 0 & h_{7eey} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

so that the entire transformation ($T$) is given by:

$$T = A^{base}_{world} T^{7}_{base} A^{ee}_{7} \tag{11}$$

## 3.2 Gravity Compensation

Gravity compensation is used to null out the effects of gravity loading of the manipulator, thereby lessening the role of feedback and improving performance. Of course for space applications this is not needed. Compensation is achieved by calculating anticipated torques driven by gravity only and uses the recursive Newton-Euler formulation. These torques ($\underline{\tau}_{gravity}$) are given by [9]:

$$\tau_{gravity_i} = \underline{s}_i^T \underline{t}_i \tag{12}$$

where:

$$\underline{t}_i = \underline{t}_{i+1} + (\underline{r}_{i+1} - \underline{r}_i) \times \underline{f}_{i+1} + \underline{h}_{i,i} \times m_i \underline{g} \tag{13}$$

$$\underline{f}_i = \underline{f}_{i+1} + m_i \underline{g} \tag{14}$$

$$\underline{h}_{i,i} = P_i \underline{r}_{cg_{i,i}} \tag{15}$$

Here $\underline{s}_i$ is the $ith$ joint axis unit vector in world coordinates, $\underline{r}_{cg_{i,i}}$ is the center of gravity vector of the $ith$ link in the $ith$ coordinate frame, $\underline{f}_i$ and $\underline{t}_i$ are the applied forces and torques to joint $i$, $m_i$ is the mass of link $i$, and $\underline{g}$ is the gravity vector expressed in world coordinates ($[0, -9.81 m/sec^2, 0]^T$).

## 3.3 Trajectory Generator

The purpose of the trajectory generator is to provide the controller with smooth input trajectories. In general this applies to both cartesian (in world or end effector coordinates) and joint interpolated motion for both autonomous and teleoperated cases. The focus here is on the most difficult and interesting case; namely autonomous cartesian control (in world coordinates). Figure 1 illustrates the specific inputs and outputs of the trajectory generator. Inputs are the initial joint angles ($\underline{\theta}_A$) (at point "A"), the commanded cartesian location transformation ($T_B$) (point "B" expressed as three positions and three Euler angles), the travel time (T), and a normalized acceleration parameter ($\alpha \geq 4/$T) – assume a symmetric trapezoidal velocity profile. Time varying outputs are the location transformation ($T(t)$) and the joint rates ($\dot{\theta}(t)$). The location transformation may be derived without knowledge of the inverse kinematics and $\dot{\underline{\theta}}(t)$ will be derived using the gradient projection technique. Also, when properly configured the trajectory generator can be used to solve the inverse positional kinematics problem by providing an initial guess for the joint angles and saving the last set of angles available internally to the trajectory generator. Of course the solution is not unique.

### 3.3.1 Location Transformation

To permit smooth simultaneous changes in orientation and position of the end effector, the required orientation change takes place about a constant vector $\underline{k}$ in world coordinates while the origin of the end effector's local coordinate frame is translating along a straight line. In both cases, trapezoidal velocity profiles are used to smooth the motion. This is equivalent to use of the following transformation to describe the end effector's time varying location:

$$T(t) = \begin{bmatrix} Rot(\underline{k}, \theta_k(t)) & \underline{p}(t) \\ \underline{0} & 1 \end{bmatrix} \tag{16}$$

where $\dot{\theta}_k, \dot{\underline{p}} = [\dot{x}, \dot{y}, \dot{z}]^T$ have trapazoidal velocity profiles (for $0 \leq t \leq$T) with symmetric ramp slopes $\pm\alpha$ and $Rot(\underline{k}, \theta_k)$ is a rotation matrix corresponding to a rotation of $\theta_k$ about a constant vector $\underline{k}$ in world coordinates given by [8]:

$$Rot(\underline{k}, \theta_k) = \begin{bmatrix} k_x k_x vers\theta_k + \cos\theta_k & k_y k_x vers\theta_k - k_z \sin\theta_k & k_z k_x vers\theta_k + k_y \sin\theta_k \\ k_x k_y vers\theta_k + k_z \sin\theta_k & k_y k_y vers\theta_k + \cos\theta_k & k_z k_y vers\theta_k - k_x \sin\theta_k \\ k_x k_z vers\theta_k - k_y \sin\theta_k & k_y k_z vers\theta_k + k_x \sin\theta_k & k_z k_z vers\theta_k + \cos\theta_k \end{bmatrix} \tag{17}$$

Here $vers\theta_k = 1 - \cos\theta_k$. The vector $\underline{k} = [k_x, k_y, k_z]^T$ and $\theta_k$(T) are evaluated in detail in Paul [8] and use the following net rotation matrix as input:

$$\Delta_{orientation} = T_{B3x3} T_{A3x3}^{-1} \tag{18}$$

with $T_{A3x3}, T_{B3x3}$ referring to their respective orientation submatrices.

### 3.3.2 Efficient Gradient Projection Technique

**Overview** In addition to using the manipulator's joints to permit cartesian end effector motion the manipulator because of its redundancy possesses self-motion; that is, the joints can move without the end effector moving. The gradient projection technique uses this motion of the manipulator to attempt to optimize a criterion of interest and has been proposed recently by several researchers [2,3]. The self-motion of the arm is characterized by the nullspace of the Jacobian (i.e. $\dot{\underline{\theta}} | J\dot{\underline{\theta}} = \underline{0}$) and it is desired to express the self-motion joint rates as the *gradient* of a potential function $H$ of interest *projected* onto the nullspace so that:

$$\dot{\underline{\theta}}_{null} = (I - J^\dagger J)k\nabla H \tag{19}$$

where $J^\dagger$ is the Moore-Penrose psuedo-inverse of $J$ (i.e. $J^T[JJ^T]^{-1}$) and $k$ is the nullspace gain. In the interests of implementation, an optimized version of the gradient projection technique is applied to the manipulator (i.e. $J^\dagger$ is never actually computed). It was developed and is described in detail in Dubey et al [3]. Basically, the technique takes advantage of the existence of a 3x3 $\underline{0}$ block in the Jacobian for spherical wristed manipulators and an assumption regarding singularities induced by the remaining 4 joints. Specifically, which of the first four columns of the Jacobian when removed still permits the remaining matrix to be invertible. The end result consists of several simplified sets of 3 linear equations in 3 unknowns to be solved, which can be done quite easily.[1]

[1] A practical detail concerns the use of a numerical integrator to estimate $\underline{\theta}(t)$ for use in evaluating the inverse Jacobian. As an aside, this estimate is subject to drift and is not accurate enough to serve as a joint position command signal, but good enough for evaluating $J$. With respect to integration techniques, both a fourth order Runge-Kutta and a simple Euler integrator have been used with success for reasonable speeds (for actual implementation the Euler integrator is perferred because of its simplicity).

**Application to Robotics Research T-710 Manipulator** Major inputs to the efficient gradient projection technique are the manipulator Jacobian which for efficiency reasons only is expressed in the third coordinate frame for the wrist (not the end effector) and the desired potential function. We will also have reason to evaluate the world to third coordinate frame transformation, the end effector to wrist velocity transformation and for convenience $J$ will be evaluated in this section (recall that it is needed in the location and force feedback loop).

The $ith$ $(i = 1, 2, ..., 7)$ column of the wrist Jacobian expressed in the third coordinate frame is given by:

$$\left[\begin{array}{c} J_3^w \end{array}\right]_i = \left[\begin{array}{c} \underline{s}_{3i} \times (\underline{r}_{3w} - \underline{r}_{3i}) \\ \underline{s}_{3i} \end{array}\right] \tag{20}$$

where $\underline{s}_{3i}, \underline{r}_{3i}$, and $\underline{r}_{3w}$ are the $ith$ joint axis unit vector, the ith joint location vector, and the wrist location vector expressed in the third coordinate frame, respectively. For the Robotics Research T-710 manipulator the Jacobian $J_3^w$ is given by:

$$J_3^w = \left[\begin{array}{ccccccc} (J_3^w)_{11} & (J_3^w)_{12} & 0 & h_{34x}s_4 + h_{45x}c_4 - h_{56y}c_4 & 0 & 0 & 0 \\ (J_3^w)_{21} & (J_3^w)_{22} & 0 & -h_{34x}c_4 + h_{45x}s_4 - h_{56y}s_4 & 0 & 0 & 0 \\ (J_3^w)_{31} & (J_3^w)_{32} & (J_3^w)_{33} & 0 & 0 & 0 & 0 \\ s_2c_3 & -s_3 & 0 & 0 & -s_4 & c_4s_5 & -s_4c_6 - c_4c_5s_6 \\ c_2 & 0 & 1 & 0 & c_4 & s_4s_5 & c_4c_6 - s_4c_5s_6 \\ s_2s_3 & c_3 & 0 & 1 & 0 & c_5 & s_5s_6 \end{array}\right] \tag{21}$$

where the lengthy entries are given in [10].

Next, an example use of the potential function will be given. In general one may envision it to be some combination (likely a weighted sum) of various subpotential functions associated with singularity avoidance, joint limit avoidance, or other criteria of interest of a heuristic nature. To attempt to avoid the wrist singularity, one could choose a potential function like $H = s_6^2$ to be maximized. For the joint limit problem, one may attempt to minimize $H = \sum_{i=1}^{7}(\theta_i - \theta_{icenter})^2$. Finally, as a heuristic example, if it is desired to keep the elbow "up" perhaps for collision avoidance reasons, one would like the $y_{world}$ component of the $-y_3$ axis (i.e. $c_2$) to be large (see Figure 2), so choose $H = (c_2 + 1)^2$ to be maximized with $k = 1$. Figure 3 shows the effect of this particular potential function as the manipulator's end effector traverses along a straight line.

The transformation projecting the end effector velocities in the world coordinate frame ($\dot{\underline{X}}$) onto the third coordinate frame is given by:

$$\dot{\underline{X}}_3^{ee} \equiv \left[\begin{array}{c} \underline{V}_3^{ee} \\ \underline{\omega}_3^{ee} \end{array}\right] = \left[\begin{array}{cc} R_{world}^3 & \underline{0} \\ \underline{0} & R_{world}^3 \end{array}\right] \dot{\underline{X}} \equiv \left[\begin{array}{cc} R_{world}^3 & \underline{0} \\ \underline{0} & R_{world}^3 \end{array}\right] \left[\begin{array}{c} \underline{V} \\ \underline{\omega} \end{array}\right] \tag{22}$$

where:

$$R_{world}^3 = \left[\begin{array}{ccc} -c_1s_3 - s_1c_2c_3 & -s_2c_3 & -s_1s_3 + c_1c_2c_3 \\ s_1s_2 & -c_2 & -c_1s_2 \\ c_1c_3 - s_1c_2s_3 & -s_2s_3 & s_1c_3 + c_1c_2s_3 \end{array}\right] \tag{23}$$

The end effector to wrist velocity transformation allows one to transform commanded end effector velocities into corresponding wrist velocities and is given by:

$$\underline{V}_3^w = \underline{V}_3^{ee} - \underline{\omega}_3^{ee} \times \underline{r}_3^{ee} \tag{24}$$

$$\underline{\omega}_3^w = \underline{\omega}_3^{ee} \tag{25}$$

where $\underline{r}_3^{ee}$ is given by:

$$\underline{r}_3^{ee} = \left[\begin{array}{c} h_{7eex}[-c_4s_5s_7 - c_7(s_4s_6 - c_4c_5c_6)] - h_{7eey}(s_4c_6 + c_4c_5s_6) \\ h_{7eex}[-s_4s_5s_7 + c_7(c_4s_6 + s_4c_5c_6)] + h_{7eey}(c_4c_6 - s_4c_5s_6) \\ -h_{7eex}(c_5s_7 + s_5c_6c_7) + h_{7eey}s_5s_6 \end{array}\right] \tag{26}$$

and $\underline{V}_3^w$ and $\underline{\omega}_3^w$ are the translational and rotational velocities of the wrist expressed in the third coordinate frame.

Having already evaluated the Jacobian needed for trajectory generation ($J_3^w$), by using the above relationships we can evaluate the Jacobian needed for location and force feedback ($J$) which describes the end effector cartesian rates in the world coordinate frame by means of:

$$J = \left[\begin{array}{c} (R_{world}^3)^T(J_3^w)_V + (R_{world}^3)^T \left[\begin{array}{cccc} (J_3^w)_{\omega 1} \times \underline{r}_3^{ee} & (J_3^w)_{\omega 2} \times \underline{r}_3^{ee} & \ldots & (J_3^w)_{\omega 7} \times \underline{r}_3^{ee} \end{array}\right] \\ (R_{world}^3)^T(J_3^w)_\omega \end{array}\right] \tag{27}$$

where:

$$J_3^w \equiv \left[\begin{array}{c} (J_3^w)_{V(3x7)} \\ (J_3^w)_{\omega(3x7)} \end{array}\right] \equiv \left[\begin{array}{c} (J_3^w)_{V(3x7)} \\ \left[\begin{array}{cccc} (J_3^w)_{\omega 1(3x1)} & (J_3^w)_{\omega 2(3x1)} & \ldots & (J_3^w)_{\omega 7(3x1)} \end{array}\right] \end{array}\right] \tag{28}$$

# 4 Kinematics, Dynamics, and Control (KDAC) Simulation

To determine anticipated performance of the above described controller and gain confidence in the approach, various simulation tools were developed/aquired and the kinematics, dynamics, and control were simulated for the Robotics Research T-710 manipulator. The simulation tools of KDAC consist primarily of MATRIXx/AR (Automation and Robotics) [9] augmented with application "blocks" for performing kinematic and control functions and a graphics animation package for displaying the results. Rigid body manipulator dynamics are modeled using appropriately configured blocks arranged to support the recursive Newton-Euler formulation of dynamics. Ideal actuators are assumed as an initial baseline, although this need not be a constraint in the simulation (e.g. actuator inertia and frictional effects can be easily included). Representative manipulator data used in the simulation are given in the Appendix. Several scenarios were simulated based on Jackson [11] which are representative of tasks required of general purpose 7 dof articulated manipulators. The first scenario is concerned with noncontact cartesian position control, the second pertains to contour following of a flat, compliant surface, and the third pertains to cooperative control. While a formal stability analysis was performed only for the first scenario, for the other scenarios, the effects of such parameters as the controller frequency, the stiffness matrix, and the environmental stiffness were investigated empirically.

## 4.1 Cartesian Position Control

Figure 4 illustrates the performance of the system (position command/actual plots) as the end effector attempts to move along a straight line approximately in the $x_{world}$ direction while retaining its orientation with an average speed of 6 inches/sec. The magnitude of the worst tracking error is 4 mm and after an additional 0.1 second the error drops to less than 1 mm (for all directions). When the controller frequency is changed from 100 Hz to 25 Hz the behavior of the system is virtually identical (not shown here). Also, when the stiffness matrix is decreased uniformly by a factor of 100, one notices a fairly insignificant degradation in performance as shown in Figure 5. In conclusion, for a modest speed of 6 inches/sec, the system is fairly robust to changes in the controller frequency and the stiffness matrix when performing noncontact straight line motion. A local stability analysis was performed using a continous linear model at a particular location. The resultant eigenvalues of the closed loop system were all stable (i.e. in left half plane) and when the rate gain was set to zero, all of the eigenvalues were on the imaginary axis. As expected then the rate gain introduces damping and pulls the open loop poles off of the $j\omega$ axis into the left half plane.

## 4.2 Contour Following

Contour following is a generic process in which the manipulator's end effector maintains contact with the surface of the environment while traversing along it. For the specific simulations performed, the manipulator attempted to maintain contact with a flat compliant surface parallel to the world xz plane while moving laterally along the surface approximately in the $x_{world}$ direction. Modeling of the interaction between the manipulator and the environment assumed a massless spring; in general an upgrade of this would be to include mass and damping effects (i.e. "impedance"). Figure 6 shows the nominal performance of the system for an environmental stiffness ($K_{env}$) of 10 kN/m. It is that of a stable lightly damped system (some oscillation). As the controller frequency decreases to 25 Hz the performance degrades considerably (see Figure 7) and is unstable. For a very stiff environment (i.e. $K_{env} \geq 1MN/m$) the system is asymptotically stable (Figure 8), but with considerable oscillation. Both the controller frequency and the environmental stiffness are seen to significantly affect the performance of the system.

## 4.3 Cooperative Control

Modeling cooperative was treated as an extension of the manner in which contour following was modeled. To simplify matters somewhat (i.e. ignore certain coupling effects), the interaction between the two manipulators was modeled as a massless spring and only translational motion was considered. Figure 9 shows the performance of the system as both manipulators laterally translate 0.94 m in the $-x_{world}$ direction while attempting to maintain a 10 $N$ compression force on the spring (in the $x_{world}$ axis direction) that ties the two arms together. Wihin numerical limits, convergence to the steady-state force level is achieved after 0.6 sec of the 3 second trajectory.

# 5 Implementation Issues

Of primary importance in the interests of actual implementation are the details of the bridge between the simulation environment and actual implementation and the computational burden of the above algorithms. This is especially true since for convenience purposes a hybrid approach was taken in developing the necessary simulation blocks (i.e. some are standard blocks and some are custom blocks). Of course, there are many other implementation issues, such as hardware considerations, etc. which will not be discussed here. With respect to the first issue, for implementation purposes only the gravity compensation and the direct kinematics block would follow the algorithm based on the standard blocks

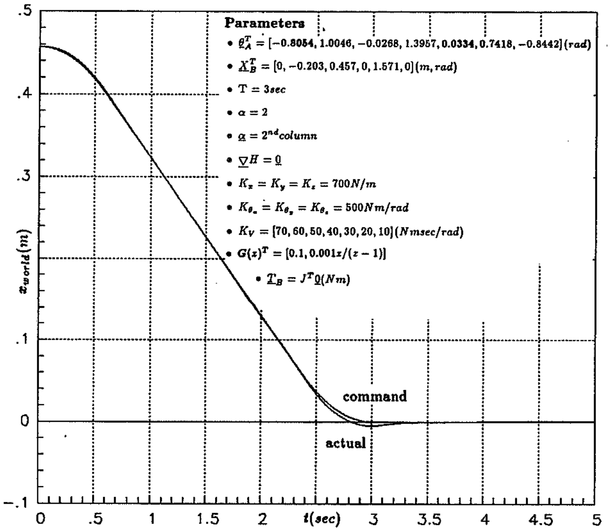


Figure 4 Position ($x_{world}$) command/actual versus time – scenario number 1 (100 Hz, nominal $K$)

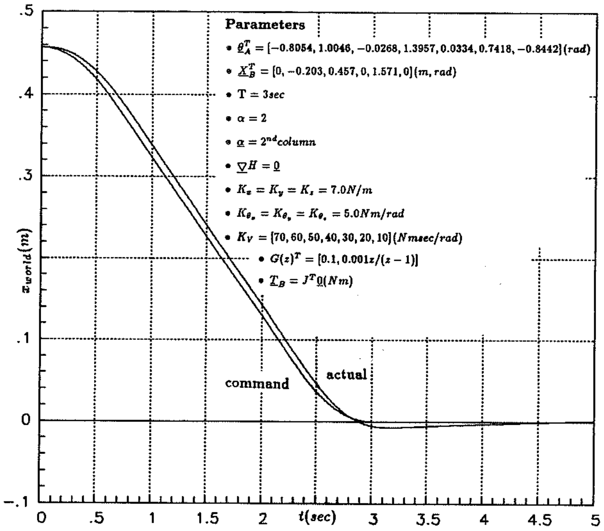


Figure 5 Position ($x_{world}$) command/actual versus time – scenario number 1 (100 Hz, nominal $K/100$)

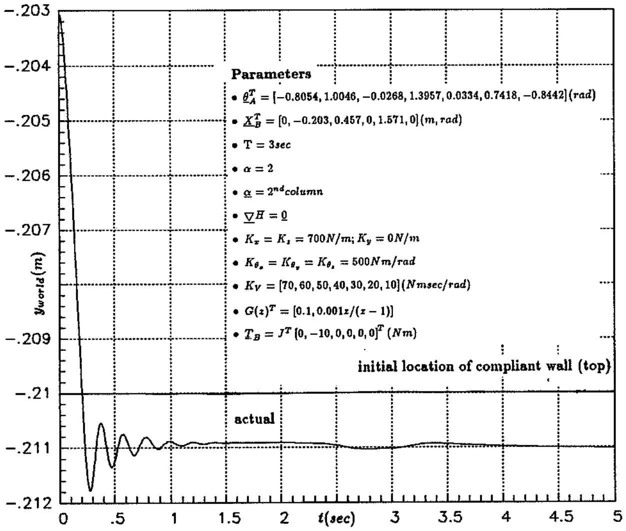


Figure 6 Position ($y_{world}$) actual versus time – scenario number 2 (100 Hz, $K_{env} = 10kN/m$)

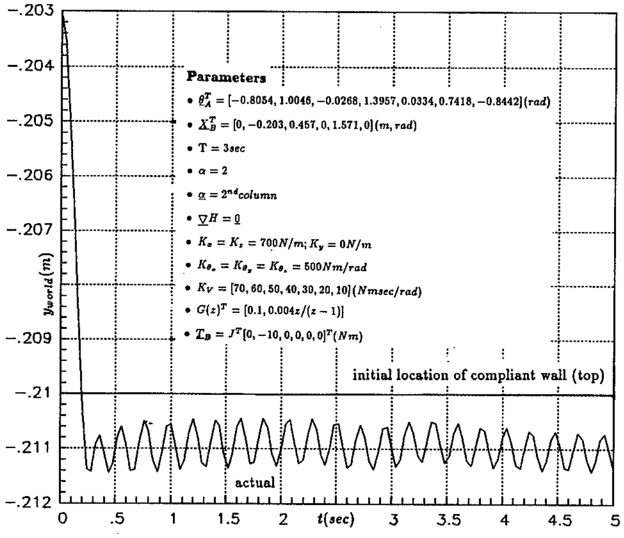


Figure 7 Position ($y_{world}$) actual versus time – scenario number 2 (25Hz, $K_{env} = 10kN/m$)

(mentioned earlier), while all other blocks would be high speed custom blocks suitable for implementation. For the second issue, timing studies based on actual timing of the trajectory generator code (not provided here) indicate that 1 MC68020 $\mu$p could comfortably implement the entire algorithm at rates exceeding 100 Hz.

# 6 Conclusions and Future Work

The active stiffness controller when combined with the gradient projection technique for control of a particular 7 dof manipulator (Robotics Research T-710) appears to work satisfactorily based on preliminary simulation studies for noncontact situations and modest controller frequencies (e.g. $\geq 25Hz$). For the more difficult case where contact is established between the manipulator and the environment or for simple cooperative control, the system is stable when an individual manipulator's environment is somewhat compliant ($K_{env} \leq 1kN/m$) and the controller frequency is sufficiently high (i.e. $\geq 100Hz$). In addition, the computational intensity of the algorithm is quite low and timing studies suggest that 1 MC68020 $\mu$p could implement the algorithm at rates exceeding 100 Hz. Future work could entail: (1) using the torque compensation loop to achieve more stable results when performing contour following of fairly rigid surfaces (i.e. $K_{env} \geq 1kN/m$), (2) a more extensive local stability analysis, (3) include a more realistic actuator and contact dynamic model, and (4) verification on actual hardware. At the time of this writing some of this work is ongoing.

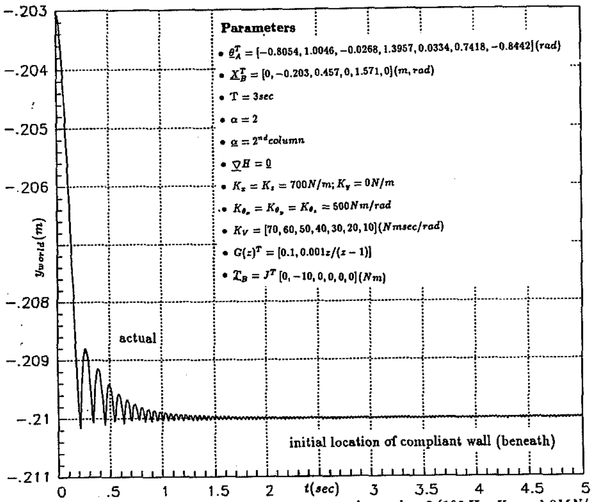


**Figure 8** Position ($y_{world}$) actual versus time - scenario number 2 (100 Hz, $K_{env} = 1.0MN/m$)

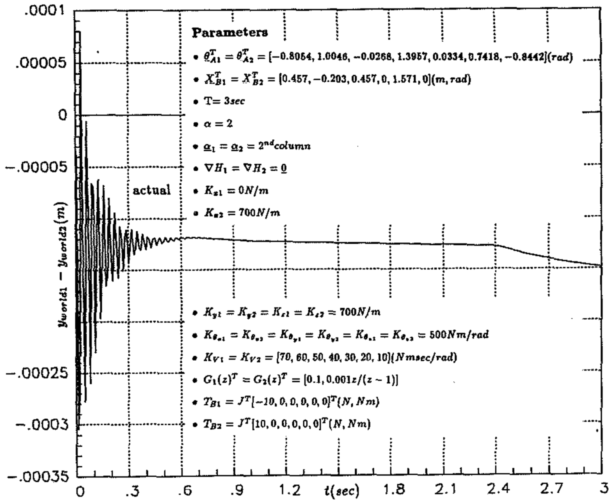


**Figure 9** Relative position error versus time - scenario number 3 ($100Hz$, $K_{spring} = 7,500N/m$)

# 7 Acknowledgements

The authors wish to acknowledge partial support of this work by the Defense Advanced Research Projects Agency (DARPA) under contract no. DAAA19-87-0094 and continued support by FMC Corporation under Independent Research and Development (IR and D) project no. 407. In addition, the authors would like to thank Professor O. Khatib of Stanford University for his consulting comments, Dan Whitney of Draper Laboratory for reviewing the paper, Paul Hasse for use of his graphical manipulator model, and Larry Coopet for performing the timing studies on the trajectory generator code.

# A Robotics Research T-710 Manipulator Kinematic Data (Representative)

- $\underline{h}_{world,base} = [0,0,0]^T(m)$
- $P^{Ref}_{world,base} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$
- $\underline{h}_{base,1} = [0,0,0]^T(m)$
- $P^{Ref}_{base,1} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
- $\underline{h}_{1,2} = [0.080, -0.099, 0]^T(m)$
- $P^{Ref}_{1,2} = \begin{bmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
- $\underline{h}_{2,3} = [0.099, 0, 0.080]^T(m)$
- $P^{Ref}_{2,3} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix}$
- $\underline{h}_{3,4} = [0.067, -0.231, 0]^T(m)$
- $P^{Ref}_{3,4} = \begin{bmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
- $\underline{h}_{4,5} = [0.086, 0, 0.067]^T(m)$
- $P^{Ref}_{4,5} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix}$
- $\underline{h}_{5,6} = [0, -0.244, 0]^T(m)$
- $P^{Ref}_{5,6} = \begin{bmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
- $\underline{h}_{6,7} = [0, 0, 0]^T(m)$
- $P^{Ref}_{6,7} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{bmatrix}$
- $\underline{h}_{7,ee} = [-0.032, -0.292, 0]^T(m)$
- $P^{Ref}_{7,ee} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

# B Robotics Research T-710 Manipulator Dynamic Data (Representative)

- link 1 mass $m_1 = 10$ kg
- link 1 center of gravity location $\underline{r}_{cg_{1,1}} = [0,0,0]^T$ (m)
- link 1 inertia tensor $I_{1,1} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- link 2 mass $m_2 = 10$ kg
- link 2 center of gravity location $\underline{r}_{cg_{2,2}} = [0,0,0]^T$ (m)
- link 2 inertia tensor $I_{2,2} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- link 3 mass $m_3 = 10$ kg
- link 3 center of gravity location $\underline{r}_{cg_{3,3}} = [0,0,0]^T$ (m)
- link 3 inertia tensor $I_{3,3} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- link 4 mass $m_4 = 4$ kg
- link 4 center of gravity location $\underline{r}_{cg_{4,4}} = [0,0,0]^T$ (m)
- link 4 inertia tensor $I_{4,4} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- link 5 mass $m_5 = 4$ kg
- link 5 center of gravity location $\underline{r}_{cg_{5,5}} = [0,0,0]^T$ (m)
- link 5 inertia tensor $I_{5,5} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- link 6 mass $m_6 = 4$ kg
- link 6 center of gravity location $\underline{r}_{cg_{6,6}} = [0,0,0]^T$ (m)
- link 6 inertia tensor $I_{6,6} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- link 7 mass $m_7 = 4$ kg
- link 7 center of gravity location $\underline{r}_{cg_{7,7}} = [0,0,0]^T$ (m)
- link 7 inertia tensor $I_{7,7} =$ diag[0.1,0.1,0.1] (kg$m^2$)
- end effector mass $m_{ee} = 1$ kg
- end effector center of gravity location $\underline{r}_{cg_{ee,ee}} = [0,0,0]^T$ (m)
- end effector inertia tensor $I_{ee,ee} =$ diag[0.1,0.1,0.1] (kg$m^2$)

# References

[1] Whitney, D.E., "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators," ASME Journal of Dynamic Systems, Measurement and Control, Dec. 1972, pp. 303-309.

[2] Baker, D.R. and Wampler, C. W., "Some Facts Concerning the Inverse Kinematics of Redundant Manipulators," IEEE International Conference on Robotics and Automation, 1987.

[3] Dubey, R., Euler, J., and Babcock, S., "An Efficient Gradient Projection Optimization Scheme for a Seven-Degree-of-Freedom Redundant Robot with Spherical Wrist," 1988 IEEE International Conference on Robotics and Automation, April, 1988.

[4] Zhang, Y. and Paul, R., "Robot Manipulator Control and Computational Cost," IEEE Workshop on Special Computer Architectures for Robotics, pp. 28-65, April 1988.

[5] An, C., Atkeson, C., and Hollarbach, J. 1988. *Model-Based Control of a Robot Manipulator.* Cambridge, Mass: MIT Press.

[6] Salisbury, J. K., "Active Stiffness Control of a Manipulator in Cartesian Coordinates," Proc. 19th IEEE Conference on Decision and Control, pp. 95-100, 1980.

[7] Karlen, J. Thompson, J. and Farrell, J., "Design and Control of Modular Kinematically-Redundant Manipulators," Second AIAA/NASA/USAF Symp. on Auto., Robotics and Adv. Comp. for the National Space Program, March 1987.

[8] Paul, R. 1981. *Robot Manipulators: Mathematics, Programming, and Control.* Cambridge, Mass: MIT Press.

[9] Integrated Systems Inc., "MATRIXx/AR (Automation and Robotics) Modeling, Control Design, and Simulation Program – User's Guide, May 1986.

[10] Hennessey, M. P., "Application of the Stiffness Controller and the Gradient Projection Technique to Control of a 7 DOF Manipulator," FMC Internal Report, Dec., 1988.

[11] Jackson, D., "Advanced Robotic Manipulator(s) Performance and Demonstration Test Plan – Draft," prepared for the U.S. ARMY AMCCOM by FMC Corporation, ASC, Minneapolis, MN, July 1988.

# INVERSE DYNAMICS OF A 3 DEGREE OF FREEDOM SPATIAL FLEXIBLE MANIPULATOR

E. Bayo and M. Serna

Department of Mechanical Engineering
University of California
Santa Barbara, CA 93106

## Abstract

A technique is presented for solving the inverse dynamics and kinematics of 3 degree of freedom spatial flexible manipulator. The proposed method finds the joint torques necessary to produce a specified end effector motion. Since the inverse dynamic problem in elastic manipulators is closely coupled to the inverse kinematic problem, the solution of the first also renders the displacements and rotations at any point of the manipulator, including the joints. Furthermore the formulation is complete in the sense that it includes all the nonlinear terms due to the large rotation of the links. The Timoshenko beam theory is used to model the elastic characteristics, and the resulting equations of motion are discretized using the finite element method. An iterative solution scheme is proposed that relies on local linearization of the problem. The solution of each linearization is carried out in the frequency domain.

The performance and capabilities of this technique are tested through simulation analysis. Results show the potential use of this method for the smooth motion control of space telerobots.

## 1. Introduction

The assumption that robot arms are rigid is not valid when considering light manipulators handling heavy payloads with accuracy. In modelling and controlling these manipulators the flexibility of their members must be considered. In particular, residual vibrations must be avoided when positioning the end-point of a robotic arm. Different techniques, too numerous in fact to be mentioned here, have been proposed to achieve this goal. They differ in the mathematical model used to describe the dynamics of the system, and by the sensors and control laws used to achieve the overall motion of the arm while controlling the tip vibrations [1-5].These results show either an initial motion of the tip in the opposite direction of the overall motion (undershoot) or an overshoot of the set-point and residual vibrations at the end of the trajectory. The initial undershoot is characteristic of the step response of a non-minimum phase system, such as a flexible link [1-5]. By relying entirely on feedback schemes, the mentioned techniques implicitly considered causal controls.

The authors introduced a direct approach for the solution of the inverse dynamics of a single-link [6,7], and multi-link [8] flexible manipulators. The proposed method finds the joint torques necessary to produce a desired end effector motion. The new technique for inverse dynamics has been validated by feed-forwarding the computed torques in numerical simulations [6-8], and in experiments [7,8] for both single and multi-link planar flexible robots, yielding excellent results. For a generic acceleration profile the solution of the inverse dynamics yields a torque that must be applied before the end-effector starts actually moving. The inverse dynamics provides a unique non-causal control law [9] which is capable of tracking a specified trajectory with no deflections from it. In particular, it is possible to track any Fourier transformable acceleration profile with zero initial undershoot or overshoot, and this therefore contradicts the belief that a non-minimum phase system can not be inverted for any given trajectory [10]. This only applies if causal solutions are sought, as is the case in Ref. [10].

In this paper that formulation is extended to compute the joint torques (and joint angles) that position the end-effector of a 3 degree of freedom spatial flexible manipulator along prescribed trajectories. The new formulation includes all the nonlinear Coriolis and centrifugal effects. Timoshenko beam theory is used to model the elastic behavior of the system and the resulting equations of motion are discretized using the finite element method. An iterative solution scheme is proposed for finding the desired joint torques, where the solution of each linearization is carried out in the frequency domain. A simulation is performed that demonstrates the capabilities of the present formulation in handling the problems of vibration control and trajectory tracking of spatial manipulators. The technique is therefore of direct use for open-loop control and it also provides the basis for closed-loop control algorithms.

## 2. Kinematics

Let us consider the spatial manipulator depicted in Fig. 1 that has three degrees of freedom and three revolute joints. The objective is to move the end effector along a given trajectory without overshoot and residual elastic oscillations of the tip. These oscillations are due mainly to the transverse elastic displacements of the links. We consider the case in which a general three dimensional motion is decomposed into three parts as shown in Fig. 2. The first and third parts are planar motions and involve the rotations about the $Z_2$ and $Z_3$ axis. The second part of the motion is composed of a rotation about the $Z_1$ axis.

*Kinematics of the first and third motion*

The first and third parts of the motion are planar. Consider the individual flexible link depicted in Fig. 3 of total length Lthat forms part of the spatial robot. A point P at a distance $x$ from the center of the hub has undergone elastic deflections of value $u_x$ and $u_y$, and rotation $\theta$.

These are defined with respect to a nominal position characterized by the moving frame $(\bar{e}_1,\bar{e}_2)$ that rotates at a specified nominal angular velocity and acceleration $\omega_n$ and $\alpha_n$, respectively. This definition of the motions with respect to the nominal frame permits the linearization of the problem from the outset.

As a consequence of the elastic deflections and rotating nominal motion, the point P is subjected to a total linear acceleration $a_p$ and angular acceleration $\alpha_p$. The acceleration of the point P can be set in terms of the nominal translational and angular accelerations at the hub, $a_n$ and $\alpha_n$, angular velocity $\omega_n$ at the hub, and the relative velocity $v_{rel}$ and acceleration $a_{rel}$ of the point P. The latter are due to the elastic deflections with respect to the moving frame. The kinematics of the motion can be established as follows:

$$\boldsymbol{a_p} = \boldsymbol{\omega_n} \times (\boldsymbol{\omega_n} \times \boldsymbol{r}) + \boldsymbol{\alpha_n} \times \boldsymbol{r} + 2\boldsymbol{\omega_n} \times \boldsymbol{v_{rel}} + \boldsymbol{a_n} + \boldsymbol{a_{rel}} \qquad (1)$$

$$\alpha_p = \alpha_n + \ddot{\theta}$$

where $\boldsymbol{r} = (x + u_x)\,\bar{e}_1(t) + u_y\,\bar{e}_2(t)$

The components of the relative velocity are $\dot{u}_x$ and $\dot{u}_y$, and those of the relative acceleration are $\ddot{u}_x$ and $\ddot{u}_y$. Performing the vectorial operations involved in Eq. (1) the following components of the accelerations are obtained:

$$a_x = -\omega_n^2 u_x - \alpha_n u_y - 2\,\omega_n \dot{u}_y + \ddot{u}_x - \omega_n^2 x + a_{nx}$$
$$a_y = \alpha_n u_x - \omega_n^2 u_y + 2\,\omega_n \dot{u}_x + \ddot{u}_y + \alpha_n x + a_{ny} \qquad (2)$$
$$\alpha_p = \alpha_n + \ddot{\theta}$$

*Kinematics of the second motion*

Fig. 4 shows the deformation of the manipulator in this second part of the motion which consists in a rotation about the global $y$ axis. The position vector for the point P now becomes.

$$r = (x + u_x)\,\overline{e}_1 + (y + u_y)\,\overline{e}_2 + u_z\,\overline{e}_3 \tag{3}$$

Note that the moving frame is now $(\overline{e}_1, \overline{e}_3)$ which rotates about the $y$ axis, hence the vectors $\omega_n$ and $\alpha_n$ contain only one non-zero component. Again performing the vectorial operations involved in Eq. (1) the following six components of the accelerations are obtained.

$$\begin{aligned} a_x &= -\omega_n^2\, x - \omega_n^2\, u_x + \alpha_n\, u_x + 2\,\omega_n\, \dot{u}_z + \ddot{u}_x\,, & \alpha_x &= \ddot{\theta}_x \\ a_y &= \ddot{u}_y\,, & \alpha_y &= \alpha_n + \ddot{\theta}_y \\ a_z &= -\,\omega_n^2\, u_z - \alpha_n\, x - \alpha_n\, u_x - 2\,\omega_n\, \dot{u}_x + \ddot{u}_z\,, & \alpha_z &= \ddot{\theta}_z \end{aligned} \tag{4}$$

Note that this part of the motion has a full three dimensional behavior, and that in this case there is coupling between the $x$ and $z$ components of the motion through the elastic deformations.

## 3. Equations of Motion

Once the displacement field has been defined, the principle of virtual displacements can be used to directly set up the dynamic equations of motion. We rely on the Timoshenko beam theory, which includes the effects of shear deformation and rotatory inertia. For the general three dimensional case the equations become:

$$\int_0^L \left( \overline{m}\,[a_x, a_y, a_z] \begin{bmatrix} \delta u_x \\ \delta u_y \\ \delta u_z \end{bmatrix} + \overline{m}\,[\eta_x\alpha_x, \eta_y\alpha_y, \eta_z\alpha_z] \begin{bmatrix} \delta\theta_x \\ \delta\theta_y \\ \delta\theta_z \end{bmatrix} \right) dx + I_h(\alpha_n + \ddot{\theta}_h)\delta\theta_h + M_t a_t \delta u_t + \tag{5}$$

$$\int_0^L \left( E\left[I_y\,\theta_y', I_z\,\theta_z'\right] \begin{pmatrix} \delta\theta_y' \\ \delta\theta_z' \end{pmatrix} + GA\left[k_z\,(\theta_z - u_y'), k_y\,(\theta_y - u_z')\right] \begin{pmatrix} \delta(\theta_z - u_y') \\ \delta(\theta_y - u_z') \end{pmatrix} + EAu_x'\delta u_x' + GJ\theta_x'\delta\theta_x' \right) dx =$$

$$\boldsymbol{T}\,(\delta\theta_h - \delta\theta_t)$$

where $\eta_x$, $\eta_y$ and $\eta_z$ are the radii of gyration of the section, $\overline{m}$ the mass per unit length, I the moment of inertia, A the area, E the Young modulus, G the shear modulus and shear coefficient k. A tip masses are represented by $M_t$, and the hub inertias by $I_h$. $\delta u_x$, $\delta u_y$, $\delta u_z$, $\delta\theta_x$, $\delta\theta_y$ and $\delta\theta_z$ represent a set of virtual elastic displacements. $\boldsymbol{T}=(T_0, T_1, T_2)$ are the unknown torques to be applied at the different joints so that the prescribed end-effector motion is obtained. The equations of motion for the planar case (motions 1 and 3) can be obtained as a particular case of Eq. (5), and they are described in detail in [8]. Note that the accelerations at the joints will have two components $\alpha_n$ and $\ddot{\theta}$. The first are the nominal accelerations and the second are the acceleration due to the elastic deflections.

The displacement field of Eq (5) can be discretized using the finite element method. A complete set of interpolation functions are defined within each element:

$$u_j(x,t) = \sum_{i=1}^{n} H_i(x)\, u_j^i(t) \quad \text{and} \quad \theta_j(x,t) = \sum_{i=1}^{n} H_i(x)\, \theta_j^i(t) \tag{6}$$

where $H_i$ are interpolation functions whose order depends on the number of nodes, $n$, in the elements; $u^i_j$ and $\theta^i_j$ indicate the components $j$ of the nodal deflections. Substituting Eq (4) and (6) in the virtual work expression, Eq (5), and following standard procedures for the formation and assemblage of element matrices [11], the equations of motion of the manipulator may be expressed by a set of time varying differential equations:

$$\mathbf{M}\ddot{\mathbf{v}} + [\,\mathbf{C} + \mathbf{C}_c(\alpha_n,\omega_n)]\,\dot{\mathbf{v}} \;+\; [\,\mathbf{K} + \mathbf{K}_c(\alpha_n,\omega_n)]\,\mathbf{v} \;= \mathbf{T} - \mathbf{F}(\alpha_n,\omega_n) \qquad (7)$$

**M** and **K** are the conventional finite element mass and stiffness matrices, respectively; $\mathbf{C_c}$ and $\mathbf{K_c}$ are the time varying Coriolis and centrifugal stiffness matrices, respectively. The matrix **C** has been added to represent the internal viscous damping of the material. The vector **T** contains the unknown torques at the joints. **F** the known forces produced by the rotating frame effect.

The set of finite element equations (7) may be partitioned as follows:

$$\mathbf{M}\begin{bmatrix}\ddot{\theta}_h\\ \ddot{v}_i\\ \ddot{v}_t\end{bmatrix} + [\mathrm{C}+\mathrm{C}_c(\alpha_n,\omega_n)]\begin{bmatrix}\dot{\theta}_h\\ \dot{v}_i\\ \dot{v}_t\end{bmatrix} + [\mathrm{K}+\mathrm{K}_c(\alpha_n,\omega_n)]\begin{bmatrix}\theta_h\\ v_i\\ v_t\end{bmatrix} = \begin{bmatrix}1\\0\\0\end{bmatrix}T - \begin{bmatrix}F_h(t)\\ F_i(t)\\ F_t(t)\end{bmatrix} \qquad (8)$$

where $\theta_h$ are the elastic rotations at the joints, $v_t$ represent the elastic deflections $(x,y,z)$ at the end-effector, and $v_i$ are all the other internal finite element elastic degrees of freedom of the manipulator. The force vector **F** is partitioned in the same manner.

## 4. Inverse Dynamics

The essence of the inverse dynamics is to find the torques $T$ at the robot joints that will cause the end of the manipulator to move along a specified nominal trajectory. The problem can be quantitatively stated as the finding of the torques $T$ in Eq. (8) under the condition that elastic deflections at the end-effector $v_t(t)$ be zero.

*Planar motions*

The first and third motions are planar. In this case the best way to solve the inverse dynamics is by establishing a recursive procedure as explained in detail in reference [8]. This procedure consists in finding first the torque corresponding to the last link under the condition that the elastic displacement at the tip is zero, the next step is to compute the reactions at the hub which will be transmitted to the next link in the chain. These reactions may be obtained simply by equilibrium considerations. The procedure continues with the next link in the chain in the same manner as before with the reaction forces included. Note that this process is conceptually similar to the recursive Newton-Euler scheme for inverse dynamics of rigid robots. The method continues with the rest of the links until the torque on the first link is determined. This way of proceeding assures that the end of each link moves along the desired nominal trajectory without oscillating from it. Once the torques $T$ have been obtained, the motions at any point of the links specified by $v_i$ or the angles $\theta_h$ (inverse kinematics) follow from Eq (8) by a direct analysis.

*Spatial Motion*

In the case of three dimensional manipulators with elastic properties in all directions, there are elastic displacements that can not be controlled by the corresponding joint torque. These elastic displacements will influence the motions of subsequent links, introducing perturbations in their nominal motion. The nominal position of each link will be modified by the elastic displacement at the end of the previous link. The recursive procedure mentioned above now requires an iteration process that will account for the displacement corrections starting with the last link. For the spatial motion at hand the manipulator does not change its configuration (except for the global rotation), and rather than using the recursive procedure, it is easier to find the solution for the three joint torques $\boldsymbol{T}$ directly from Eq. (8) under the condition that three elastic deflections at the end-effector $v_t(t)$ be zero.

The forcing terms in the right hand side of Eq. (6) depend on the velocities and acceleration profiles of the nominal motion of the link. These will have to be Fourier transformable for the problem to have a solution. Again once the torques $\boldsymbol{T}$ have been found, the motions at any point of the robot specified by $v_i$ and the joint angles $\theta_h$ (inverse kinematics) can be readily obtained from Eq.(8) by a direct analysis. It becomes obvious at this point how the inverse dynamics and kinematics are closely coupled in the analysis of flexible robots.The torques $\boldsymbol{T}$ can be found in the frequency domain by means of an iterative numerical procedure similar to that used by Bayo et al [7-8]. In order to set up the iterative process Eq. (8) may be restated as follows:

$$\mathbf{M}\ddot{\mathbf{v}} + \mathbf{C}\dot{\mathbf{v}} + \mathbf{K}\mathbf{v} = \mathbf{1}\boldsymbol{T} - \mathbf{F} - \mathbf{C}_c(\alpha_n,\omega_n)\dot{\mathbf{v}} - \mathbf{K}_c(\alpha_n,\omega_n)\mathbf{v} \qquad (9)$$

where all the time invariant terms have been left in the L.H.S. of the equation and the time variant ones have been moved to the R.H.S. The vector $\mathbf{1}$ contain unit values for the degrees of freedom corresponding to the joint rotations and zero for the rest, as shown in Eq. (8).

The iteration process is initiated by solving Eq.(9) for the unknown torques $\boldsymbol{T}$ in the frequency domain (see [7-8] for a detailed description of this process). The first iteration is done in the absence of the last two terms involving $\mathbf{C_c}$ and $\mathbf{K_c}$ in the RHS. The first iteration will yield a displacement vector $v^1(t)$ which in turn will be used to compute $\mathbf{K_c}$ and $\mathbf{C_c}$, and the last two terms in Eq. (9). The process is then repeated with the new force vector under the constraint that $v_t(t)=0$. The iteration procedure may be stopped when the norm $\|\boldsymbol{T}_i - \boldsymbol{T}_{i-1}\|$ for the solution of two consecutive iterations, is smaller than a prescribed tolerance.

## 5. Simulation Analysis

Let us consider as an example a manipulator with the following characteristics:

First link: $L=0.2$ m, $A=1.90\text{x}10^{-4}$ m$^2$, $I_1=I_2= 3.5\text{x}10^{-9}$ m$^4$,
$J=2.0\text{x}10^{-8}$ m$^4$, $M_t=2.0$ Kg, $I_h=0.002$ m$^4$

Second link: $L=0.6$ m, $A=1.20\text{x}10^{-4}$ m$^2$, $I_1=I_2= 2.2864\text{x}10^{-10}$ m$^4$,
$J=4.57\text{x}10^{-10}$ m$^4$, $M_t=1.0$ Kg, $I_h=0.0012$ m$^4$

Third link: $L=0.6$ m, $A=0.40\text{x}10^{-4}$ m$^2$, $I_1=I_2= 8.4683\text{x}10^{-12}$ m$^4$,
$J=1.69\text{x}10^{-11}$ m$^4$, $M_t=0.15$ Kg, $I_h=0.00048$ m$^4$

They share the following properties: $E=7.11\text{x}10^{10}$ N/m$^2$, mass density = 2715 Kg/m$^3$, shear coefficient $k=1/2$ and a damping ratio of 0.002. The first frequencies of the bending modes in rd/sec of the second and third links under pinned-free conditions are: for the second link $\omega_1=0$ (rigid body mode), $\omega_2=157.53$, $\omega_3=487.55$; and for the third link $\omega_1=0$, $\omega_2=54.73$, $\omega_3=181.98$ and $\omega_4=389.93$. The natural frequencies of first link are so high that it is considered as being rigid. The manipulator is modeled with 11 finite elements with a total of 67 degrees of freedom.

The first and third motions consist in straight-line tip trajectories (each one of 1 second of duration) generated according the acceleration profile shown in Fig. 5 which corresponds to a Gaussian velocity profile [12]. The second motion consists in a circular tip trajectory whose acceleration is shown in Fig. 5 also and which last 3 seconds. The solution of the inverse dynamics is not limited by the type of trajectories chosen, as long as their acceleration profiles are

Fourier transformable [9]. As shown in [7] and [12], other type of trajectories can be perfectly used. However the trajectory corresponding to a Gaussian velocity profile is selected because it diminishes the high frequency content of the input function.

The torques that need to be applied at each actuator to yield the required tip trajectory are calculated according to the procedures described above. As examples Fig. 6 shows the torque $T_2$ (flexible torque) for the first motion, compared to the torque needed to produce the same tip motion on an infinitely rigid link of the same mass (rigid torque). Fig. 7a and 7b show the torques $T_0$ and $T_2$ for the second motion. It is important to note that in order to track the desired trajectory the flexible torques need to be applied before the tip actually moves, as also noted in [7-8]. In addition, it may be seen also that the flexible torques have a smaller peak value, and are more spread over time than the rigid one. As a result of the inverse kinematics, Fig. 8 compares the nominal angle at joint 2 with the calculated one for the first part of the motion. These could obviously be used in a collocated feed-back control scheme.

All the calculations were performed using a SUN 160 workstation. The total time required for the computation of these torques for all the three motions was approximately 20 seconds. The computational burden is mostly due to the solution of the complex sets of equations and the use of the fast Fourier transform. The computed torques produce the desired tip trajectory without overshoot or residual oscillations. A forward dynamic analysis and graphics simulation are performed yielding the results shown in Fig.9 and 10. It may be observed how both links bend along their motion. However, their ends follow the specified nominal path. Finally Fig. 12 illustrates the motion of the end-effector when the rigid torque at joint 0, corresponding to the second motion, is input to the system. Although the rigid torque is smooth the tip still vibrates, and this points out the fact that smooth trajectories or torques are not enough to eliminate residual vibrations.

## 6. Conclusions and Acknowledgement

A procedure has been presented for the inverse dynamics and kinematics of a three degree of freedom spatial manipulator. The proposed algorithm calculates the torques required to move the end effector through a specified trajectory while avoiding tip oscillations. The tip trajectory acceleration profile should be Fourier transformable. The technique is demonstrated through simulation yielding very good results. The algorithm converges very rapidly in all the cases considered.

The use of the fast Fourier transform, solution of complex equations and the necessity to iterate to obtain the required solution preclude this method from being used in real time control. However, it can still be used in open-loop control and other close-loop control strategies. Currently, under investigation are time domain algorithms to be implemented in multi-processors computer architectures which will substantially reduce the computational time.

The authors would like to acknowledge the support of this work by NATO, grant number 0877-87, and by the National Science Foundation, grant number 8421415, through the Center for Robotics Systems in Microelectronics.

## References

1. R. H. Cannon and E. Schmitz, "Initial Experiments on End-Point Control of a Flexible One-Link Robot," *Int. J. Robotics Res.*, Vol. 3, No. 3, pp. 62-75 (1984).
2. Y. Sakawa, F. Matsuno, and S. Fukushima, " Modeling and Feedback Control of a Flexible Arm," *J. Robotic Syst.*, Vol. 2, No. 4. pp. 453-472 (1985).
3. F. Harashima, T. Ueshiba ,"Adaptative Control of Flexible Arm Using the End-Point Position Sensing". Japan-USA Symposium on Flexible Automation. Osaka, Japan. Vol. 1, 225-229 (1986).
4. P. Kotnik, S. Yurkovik and Ü. Özgüner, "Acceleration Feedback Control for a Flexible Robot Arm," *J. Robotic Syst.*, Vol. 5, No. 3, pp. 181-196 (1988).

5. D.Wang and M. Vidyasagar, "Control of a Flexible Beam for Optimum Step Response," *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1567-1572 (1987).

6. E. Bayo, "A Finite-Element Approach to Control the End-Point Motion of a Single-Link Flexible Robot," *J. Robotic Syst.*, Vol. 4, No. 1, pp. 63-75 (1987).

7. E. Bayo, R. Movaghar and M. Medus, "Inverse Dynamics of a Single-Link Flexible Robot. Analytical and Experimental Results". *Int. J. of Robotics and Automation.* Vol. 3, No. 3, pp.150-157, (1988).

8. E. Bayo, M. Serna, P. Papadopoulos, J. Stubbe "Inverse Dynamics and Kinematics of Multi-Link Elastic Robots. An Iterative Frequency Domain Approach" Report # UCSB-ME-87-7 Mechanical Engineering Department, University of California Santa Barbara, Dec 1987. To appear in the *International Journal of Robotics Research.*

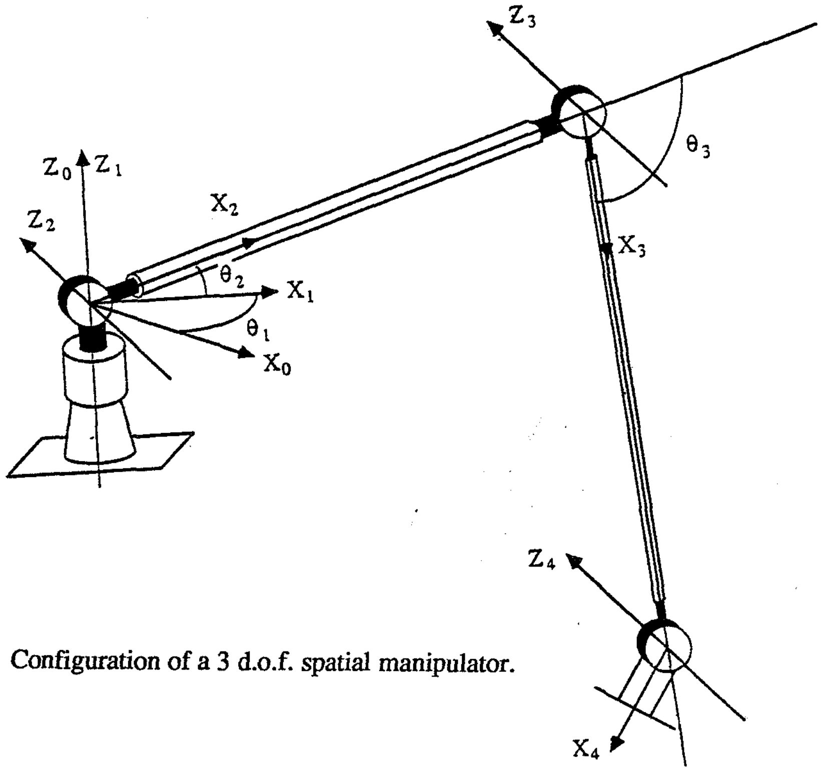
9. Moulin H. and Bayo E. "On the Accuracy of End-Point Trajectory Tracking for Flexible Arms by Non-Causal Inverse Dynamic Solutions," Report # UCSB-ME-88-9 Mechanical Engineering Department, University of California Santa-Barbara, Nov 1988.

10. A. De Luca and B. Siciliano, "Joint-Based Control of a Nonlinear Model of a Flexible Arm," Proc. American Control Conference. Atlanta. (July 1988).

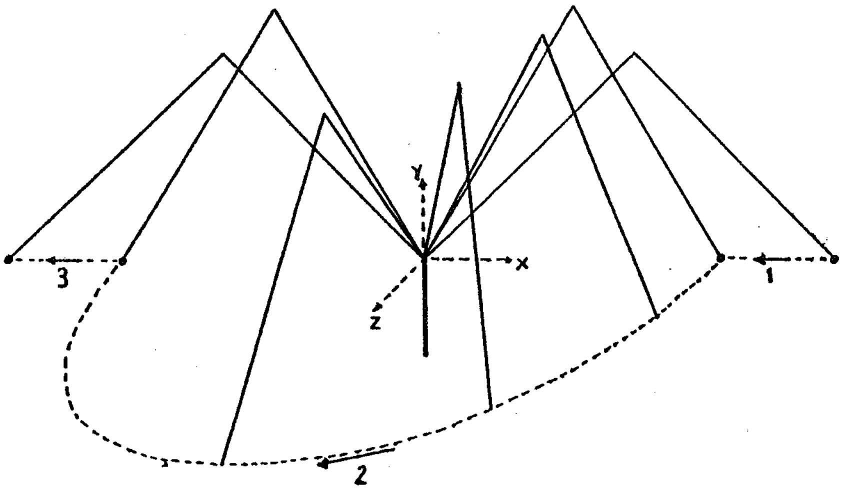
11. K. J. Bathe, *Finite Elements Procedures in Engineering Analysis*, Prenctice-Hall, Englewood Cliffs, NJ, 1982.

12. Bayo, E. and Paden, B. "On Trajectory Generation of Flexible Robots". *Journal of Robotic Systems.* Vol. 4 No. 2, pp. 239-245. Spring 1987.

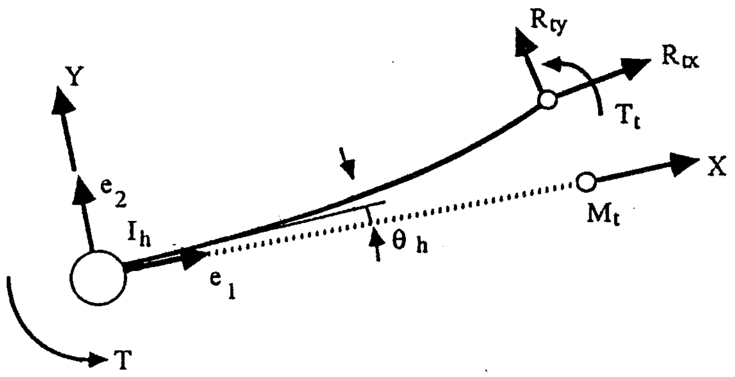
## figures



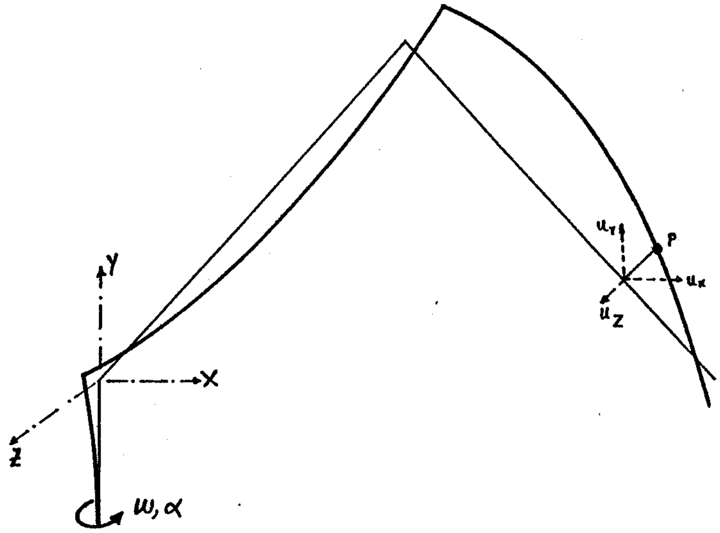
1. Configuration of a 3 d.o.f. spatial manipulator.



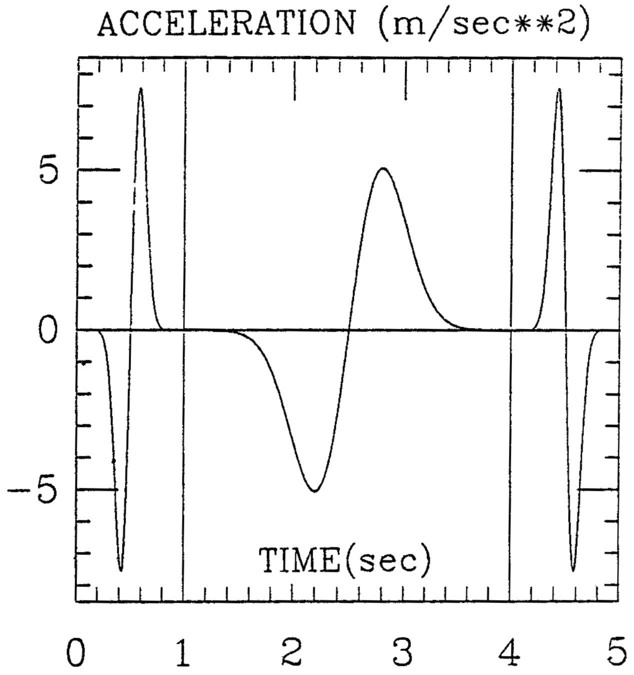
2. The three parts of the motion.



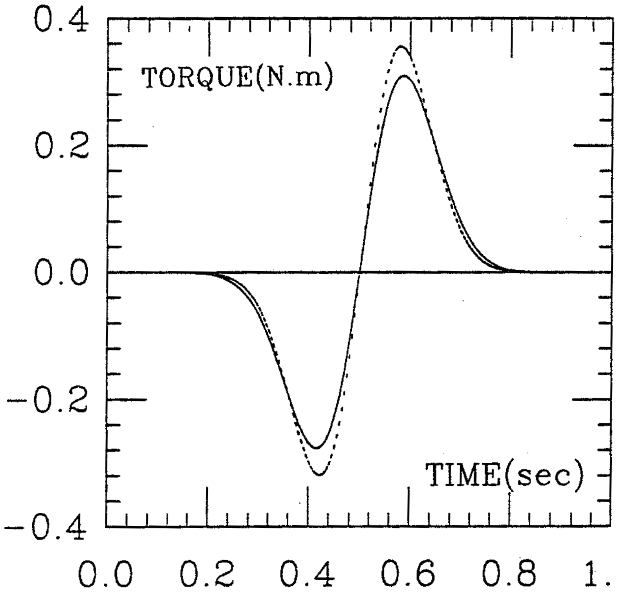
3. Deformation of a link in planar motion.



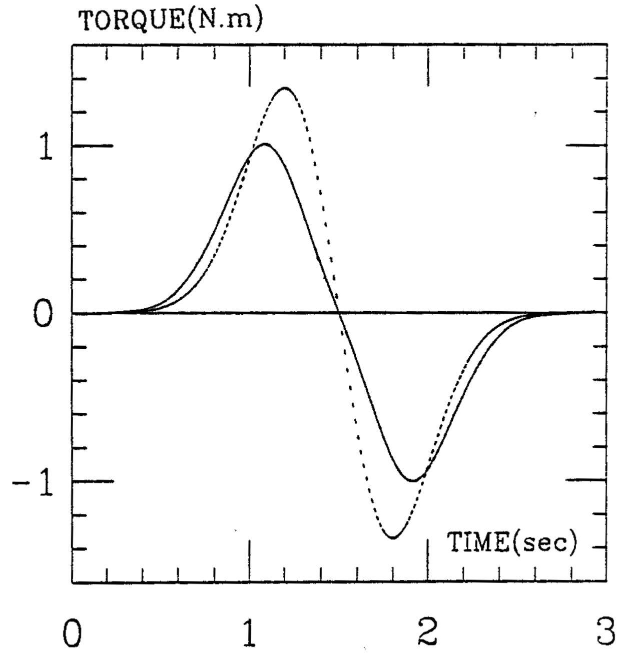
4. Deformation of the robot in the spatial motion.



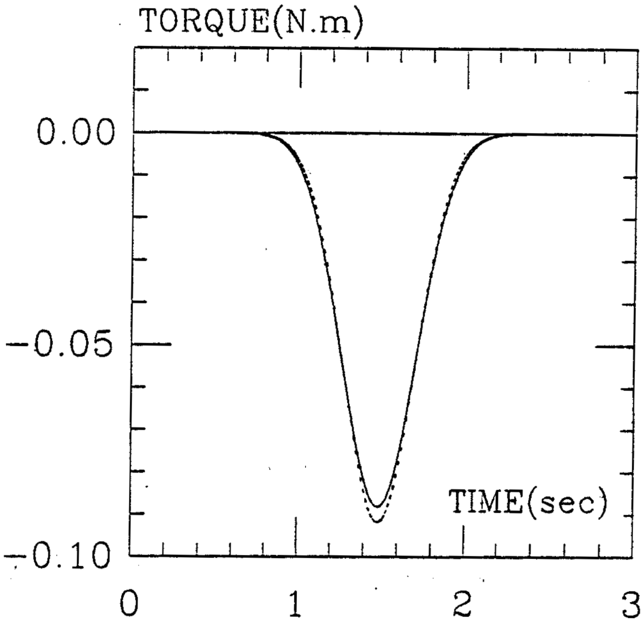
5. Acceleration profiles for the three motions.



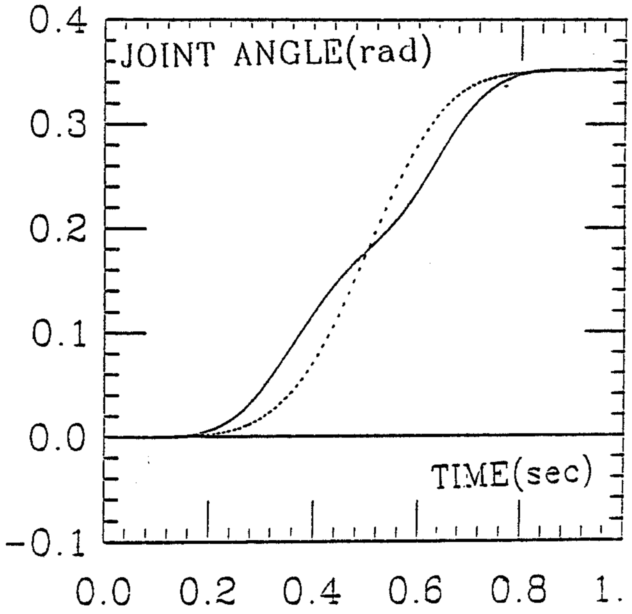
6. Joint 2. First part of the motion.
Rigid torque (dotted) vs. flexible torque (solid).



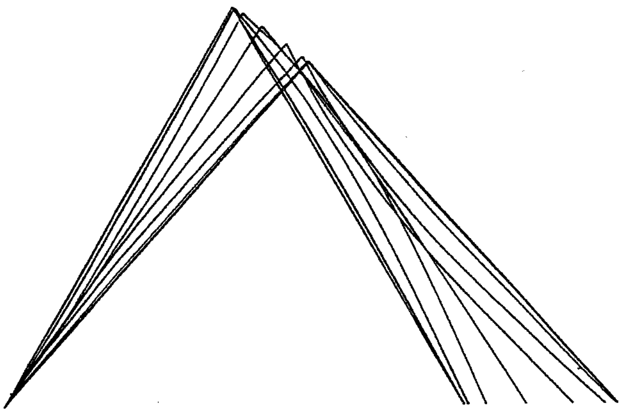
7a Joint 0. Second part of the motion.
Rigid torque (dotted) vs. flexible torque (solid).



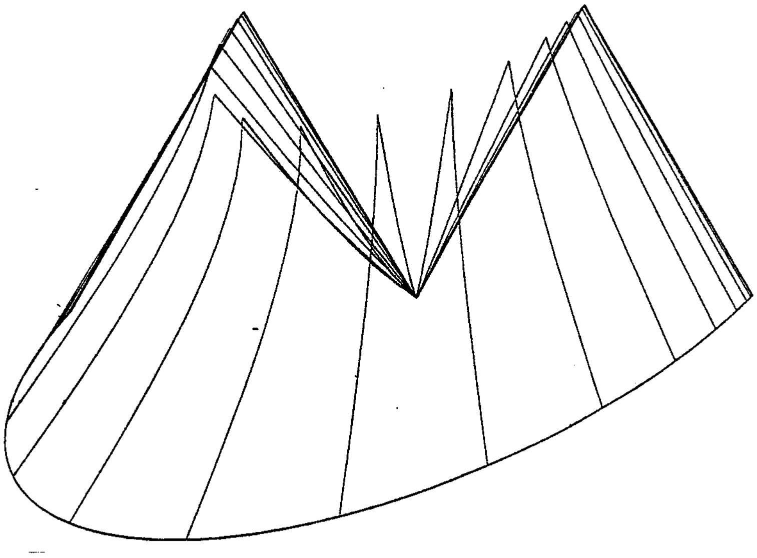
7b Joint 2. Second part of the motion.
Rigid torque (dotted) vs. flexible torque (solid).



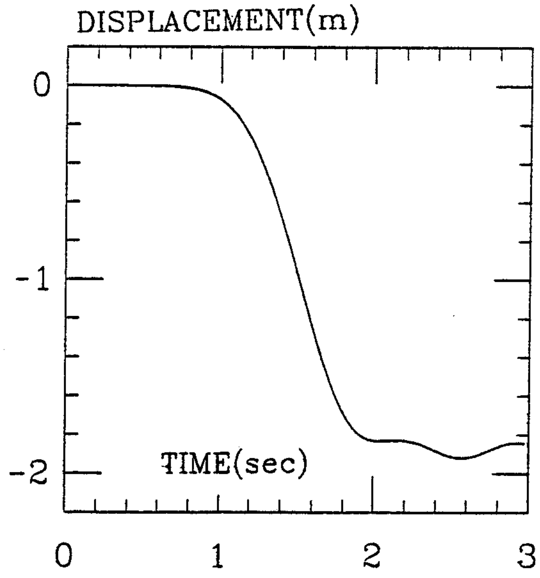
8 Joint 2. First part of the motion.
Nominal (dotted) vs. calculated (solid) joint angles.



9. Simulation of the first part of the motion.



10. Simulation of the second part of the motion.



11. Vibration at the end-effector when using the rigid torque. Second part of the motion.

N90-29879

# A CONTROL APPROACH FOR ROBOTS WITH FLEXIBLE LINKS AND RIGID END-EFFECTORS

Enrique Barbieri
Tulane University
Electrical Engineering Department
New Orleans, LA 70118

Ümit Özgüner
The Ohio State University
Electrical Engineering Department
Columbus, OH 43210

## ABSTRACT

Multiarm flexible robots with dexterous end-effectors are currently being considered in such tasks as satellite retrieval, servicing and repair where a two-phase problem can be identified: Phase I, robot positioning in space; Phase II, object retrieval. In this article we only consider some issues in Phase I regarding modelling and control strategies for a robotic system comprised of a long flexible arm and a rigid three-link end-effector. The control objective is to maintain the last (rigid) link stationary in space in the presence of an additive disturbance caused by the flexible energy in the first link after a positioning maneuver has been accomplished. In our formulation, several configuration strategies can be considered, and optimal decentralized servocompensators can be designed. Preliminary computer simulations are included for a simple proportional controller to illustrate the approach.

## I. Introduction

One of the requirements of current and future robot manipulators is that they be lightweight to permit rapid operation at a minimum energy cost. A typical application involves the (perhaps rapid) slewing of the shuttle arm in a satellite holding and retrieval task. Numerous studies have been done to find adequate controllers for robots consisting of one or more flexible/rigid links, including linear and nonlinear feedback, adaptive techniques, feed forward control, servocompensators, and optimal control [1-6].

One robot configuration that has not received attention is that of a long, lightweight (flexible) first link, followed by a dexterous end-effector consisting of rigid links numbered 2 onwards. The problem considered is that of keeping the end-effector fixed in space in orientation after a slew operation, which has been accomplished using time-optimal or minimal energy constraints. Due to the inherent flexibility of link 1, its tip will vibrate causing disturbances that affect (rigid) links 2 through n. The question we are addressing is twofold: 1) Can joints 2 through n move to keep link n fixed? What are the kinematic configurations that can accomplish this vibration compensation?, and 2) What control strategy must be followed by joints 2 on, for this compensation to be effective?

In answering these questions, we may consider the last link fixed and look at the reverse problem for a fictitious rigid manipulator comprised of links n - 1 through 2 (the last link is considered the base). Thus, this is now a problem of analyzing the space that the tip of the fictitious rigid manipulator can move, and matching it to the vibrations of a single link manipulator. A number of different solutions and configurations can be considered depending on the type of joints that the rigid manipulator has. The manipulator considered here is shown in Figure 1. The system consists of four links labeled L1 through L4. The first link is very long and flexible (L1 >> L2, L3, L4), and the last three links represent a rigid robot arm with an end-effector. The hub joint is revolute to allow planar or single-axis slewing maneuvers; the second joint is also revolute, the third joint is both revolute and prismatic, and the fourth joint is revolute.

The following assumptions/simplifications are made:

* The flexible disturbance is additive.

* The end-effector robot can be modelled as a lumped point mass.

The modelling restriction may be relaxed in a later study and a more detailed model can be obtained [7-9].

Definition 1:

The rigid configuration is the orientation that the arm and end-effector would assume if the arm were rigid.

The main goal of this paper is to solve the control problem of system orientation about the rigid configuration in the presence of flexible disturbances. One particular configuration strategy is considered in the following section.

## II. A Configuration Strategy

The problem formulated in the Introduction is now solved within the framework of a particular configuration strategy. To that end, consider the relegation of control effort to Joint J2 so that link L2 remains parallel to the rigid configuration. We will label this strategy as the absolute flexible-to-rigid configuration strategy. Given this orientation constraint, our task is to determine the appropriate control commands for joints J2, J3, J4.

Figure 1 illustrates the rigid configuration which is also the "steady-state" orientation after the flexible disturbance has disappeared. We attach the coordinate frames {1}, {2}, {3}, and {4} to each of the links as shown in the figure. Reference frame {0} is an inertial coordinate system. Using these frames, a point ${}^{1}P_{1}$ in {1}, for example, can be expressed in {2} as ${}^{2}P_{1}$ by the relation:

$$\begin{bmatrix} {}^{2}P_{1} \\ 1 \end{bmatrix} = {}^{2}_{1}T \begin{bmatrix} {}^{1}P_{1} \\ 1 \end{bmatrix}$$

where the homogeneous frame transformation ${}^{2}_{1}T$ is given by

$$ {}^{2}_{1}T = \begin{bmatrix} {}^{2}_{1}R(\theta_{12}) & {}^{2}P_{1org} \\ 0 & 1 \end{bmatrix} \; ; \; {}^{2}_{1}R(\theta_{12}) = \begin{bmatrix} \cos(\theta_{12}) & \sin(\theta_{12}) \\ -\sin(\theta_{12}) & \cos(\theta_{12}) \end{bmatrix} $$

where ${}^{2}P_{1org}$ is a vector that expresses the origin of {1} in frame {2}.

Assume that a suitable rigid configuration has been determined with the corresponding solution set

$$ \cdot \left( \Psi^{r}_{01}, \Psi^{r}_{12}, (\Psi^{r}_{23}, x^{r}), \Psi^{r}_{34} \right) $$

where, as illustrated in Figure 1, $\Psi^{r}_{ij}$ is the angle between frame i and frame j, and $x^{r}$ is the displacement of the prismatic joint.

We next define "flexible coordinate frames" {1F}, {2F}, and {3F} which coincide exactly with frames {1}, {2}, and {3}, respectively, once the flexible disturbance has disappeared. We also define the flexibility pair ($\beta$, $Y_{TIP}$) where $\beta$ is a small flexibility angle measured locally at the tip of link L1 and $Y_{TIP}$ is the corresponding local transverse displacement. Then,

$$ \Psi_{01} = \Psi_{0,1F} = \Psi^{r}_{01} + \beta $$

and for the configuration strategy under consideration,

$$ \Psi_{02} = \Psi_{0,2F} = \Psi^{r}_{02} $$

from which it follows that

$$ \Psi_{12} = \Psi_{1,2F} = \Psi^{r}_{12} - \beta. \qquad (1) $$

To determine the flexible displacement $x = x^{f}$ of the prismatic joint, we observe that

$$ {}^{3F}P_{4org} = \begin{bmatrix} x^{f} \\ 0 \end{bmatrix} = \begin{bmatrix} x \\ 0 \end{bmatrix} = -{}^{3F}_{4}R \begin{bmatrix} p \\ q \end{bmatrix} \qquad (2) $$

$$ \text{where} \quad \begin{bmatrix} p \\ q \end{bmatrix} = {}^{4}_{3}R \; {}^{3}_{2}R \; {}^{2}_{1}R \begin{bmatrix} L_{1}(\cos(\beta)-1) \\ Y_{TIP} \end{bmatrix} - {}^{4}_{3}R \begin{bmatrix} x^{r} \\ 0 \end{bmatrix} $$

$$ \text{and} \quad \Psi_{43} = \Psi_{4,3F} = \begin{cases} \mathrm{ATAN}(\frac{q}{p}) & \text{if } p < 0 \\ \pi + \mathrm{ATAN}(\frac{q}{p}) & \text{if } p > 0 \text{ and } q < 0 \\ -\pi + \mathrm{ATAN}(\frac{q}{p}) & \text{if } p > 0 \text{ and } q > 0 \end{cases} \qquad (3) $$

Finally, the angular set point of joint J3 is found to be

$$\Psi_{32} = \Psi_{3F,2F} = \Psi^{r}_{32} - [\Psi_{43} - \Psi^{r}_{43}] \qquad (4)$$

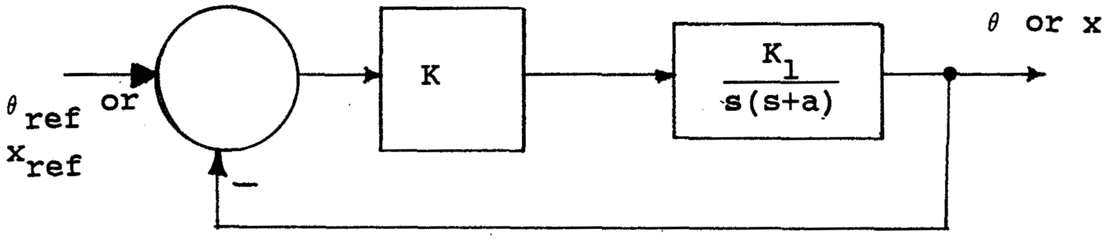
Equations 1-4 are the solution to the regulation problem for this particular configuration strategy. Note that the joint coordinates can be found from the following relations:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = {}^{0}_{1}T \begin{bmatrix} L_1 \\ 0 \\ 1 \end{bmatrix} ; \begin{bmatrix} x_3 \\ y_3 \\ 1 \end{bmatrix} = {}^{0}_{1}T\, {}^{1}_{2}T \begin{bmatrix} L_2 \\ 0 \\ 1 \end{bmatrix} \qquad (5a)$$

$$\begin{bmatrix} x_4 \\ y_4 \\ 1 \end{bmatrix} = {}^{0}_{1}T\, {}^{1}_{2}T\, {}^{2}_{3}T \begin{bmatrix} x \\ 0 \\ 1 \end{bmatrix} ; \begin{bmatrix} x_5 \\ y_5 \\ 1 \end{bmatrix} = {}^{0}_{1}T\, {}^{1}_{2}T\, {}^{2}_{3}T\, {}^{3}_{4}T \begin{bmatrix} L_4 \\ 0 \\ 1 \end{bmatrix} \qquad (5b)$$

## III. Simulations

In this section we include some representative simulations that illustrate the results obtained thus far. For simplicity, we assume that all motors are identical and can be modelled as shown in Figure 2.



$$H(s) = \frac{\omega_o^2}{s^2 + 2\,\xi\omega_o\, s + \omega_o^2}$$

Figure 2. Motor Model

where $\theta$ is the angle of a revolute joint and x is the linear displacement of a prismatic joint. The gain K can be chosen so that a suitable response is obtained. In the simulations we set the damping ratio $\xi = 0.707$ and $\omega_o$ is a design parameter. This simple proportional controller is used to illustrate the results and should be viewed only as a representative design. A procedure to construct a suitable finite dimensional model of the slewing beam can be found in [10]. A slewing profile can be generated by applying a torque input to the model with hub rate and angle feedback [11]. The physical parameters used, structural modes and closed-loop modes are listed in Table 1. We have chosen the first eight modes to constitute a "truth" model for the flexible beam. A typical simulation of hub angle $\Psi_{01}$ and tip position $Y_{TIP}$ is shown in Figure 3 for two and eight modes. Notice that no attempt is made to decrease the structural vibrations and a "poor" slewing profile is deliberately generated. Table 2 lists the initial and final rigid system configurations.

Equations 1-4 are used to compute the required motor set-points based on a reduced order model of the beam that retains two or four modes. The resulting joint angles and the true hub angle and tip position (from the 8-mode model) are then used to compute the position of the links (see Equation 5). One representative simulation is given in Figure 4. An expanded view of the error in the coordinates of the last link is shown in Figure 5. There, we observe that the y-coordinate of the tip ($y_5$) has a maximum error of about 2 cm. Table 3 compiles the results for several simulation runs and different choices of motor bandwidth and number of modes used in the reduced-order model of the flexible link. Increasing the motor's bandwidth reduces the overall regulation error to some extent. The major contribution to the error is seen to be the inexact account of the total deviation of link L1 from its rigid position. This is a consequence of employing a controller based on a reduced-order model of the flexible beam.

## IV. Conclusions and Further Research

We have presented a control strategy for a flexible manipulator with rigid end-effector. Our preliminary results are promising, and current research efforts include analysis of other configuration strategies, extension to a similar robotic system with a three-dimensional workspace, coordination of multiple robotic arms, and implementation of an optimal decentralized servocompensator [11,12] for each motor for reference tracking and flexible disturbance rejection.

## V. References

[1] H. Seraji, "Design of Adaptive End-Effector Controllers for Manipulators", in Proceedings of the 4th IFAC Symposium on Control of Distributed Parameter Systems, Los Angeles, CA, June 30-July 2, 1986.

[2] B. Gebler, "Feed-Forward Control Strategy for an Industrial Robot with Elastic Links and Joints", in Proceedings of the 3th IEEE Conference on Robotics and Automation, Rayleigh, NC, April 1987, pp.923-928.

[3] H. J. Buchner and H. Hemami, "Servocompensation of Disturbances in Robotic Systems", International Journal of Control, 1988, Vol.48, NO.1, pp.273-288.

[4] S. Yurkovich, A. Tzes and F. Pacheco, "Experiments in the Identification and Control of Flexible Link Manipulators", in these Proceedings, (to appear).

[5] J. Juang, L. G. Horta and H. H. Robertshaw, "A Slewing Control Experiment for Flexible Structures", Journal of Guidance and Control, Vol. 9, NO. 5, pp.599-607, Sep-Oct 1986.

[6] E. Barbieri and Ü. Özgüner, "Rest-to-Rest Slewing of Flexible Structures in Minimum Time", in the Proceedings of the 27th IEEE Conference on Decision and Control, Austin, TX, December 1988, pp.1633-1638.

[7] G. Naganathan and A. H. Soni, "Non-linear Flexibility Studies for Spatial Manipulators", in Proceedings of the 1986 IEEE International Conference on Robotics and Automation, pp.373-378.

[8] G. Naganathan and A. H. Soni, "An Analytical and Experimental Investigation of Flexible Manipulator Performance", in Proceedings of the 1987 IEEE International Conference on Robotics and Automation, pp.767-773.

[9] T. J. Tarn, A. K. Bejczy and X. Ding, "On the Modelling of Flexible Robot Arms", Robotics Laboratory Report, SSM-RL-87-08, Department of Systems Science and Mathematics, Washington University, St. Louis, Missouri, 63130.

[10] E. Barbieri and Ü. Özgüner, "Unconstrained and Constrained Mode Expansions for a Flexible Slewing Link", ASME Transactions, Journal of Dynamic Systems, Measurements and Control, Vol. 110, December 1988, pp.416-421; also in the Proceedings of the American Control Conference, Atlanta, GA, June 1988, pp.83-88.

[11] E. Barbieri, Ü. Özgüner and S. Yurkovich, "Vibration Compensation in Optical Tracking Systems", AIAA Journal of Guidance, Control and Dynamics, March 1989, (to appear).

[12] A. İftar and Ü. Özgüner, "A Linear-Quadratic Optimal Controller for the Servomechanism Problem", in Proceedings of the 25th IEEE Conference on Decision and Control, Athens, Greece, December 1986, pp.1784-1785.

Table 1. Physical Link Parameters and Modes

| | |
|---|---|
| Link 1 | $L_1 = 4.0m$ ; $\rho = 0.4847$ Kg/m ; $I = 3.3339 \times 10^{-11} m^4$<br>$E = 6.8944 \times 10^9$ N/m$^2$ ; $A = 1.5875 \times 10^{-4} m^2$<br>Tip Mass = 0.2 Kg<br>$L_2 = L_3 = 0.5$ m ; $L_4 = 0.2$ m |
| Structural Modes (Hz) | 0 ; 0.0884 ; 0.2635 ; 0.4828 ; 0.8151<br>1.3090 ; 1.9535 ; 2.7426 ; 3.6846 |
| Closed Loop Modes | $-1.7723 \times 10^{-4} \pm j\ 0.1177$ ; $-6.4417 \times 10^{-3} \pm j\ 0.7731$<br>$-6.2638 \times 10^{-2} \pm j\ 2.2021$ ; $-5.1936 \times 10^{-1} \pm j\ 3.9712$<br>$-3.6426 \times 10^{-1} \pm j\ 5.2491$ ; $-7.4596 \times 10^{-2} \pm j\ 8.2481$<br>$-2.6528 \times 10^{-2} \pm j\ 12.281$ ; $-1.1691 \times 10^{-2} \pm j\ 17.234$<br>$-7.4112 \times 10^{-3} \pm j\ 23.152$ |

Table 2. Initial and Final Configurations
($\Psi_{ij}$ in degrees ; x,y in meters)

| $\Psi_{01}$ | $\Psi_{12}$ | $\Psi_{23}$ | x | $\Psi_{34}$ | $x_2$ | $y_2$ | $x_3$ | $y_3$ | $x_4$ | $y_4$ | $x_5$ | $y_5$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 55 | 45 | -80 | 0.2 | -30 | 2.29 | 3.27 | 2.21 | 3.77 | 2.39 | 3.84 | 2.59 | 3.81 |
| 25 | 20 | -40 | 0.1 | -20 | 3.63 | 1.69 | 3.98 | 2.04 | 4.08 | 2.05 | 4.27 | 2.00 |

Table 3. Error Range in Link 4 Displacement
* Time Interval = 15-30 sec
* First entry corresponds to 2 mode controller
* Second entry corresponds to 4 mode controller

| BW | Error Range (cm) | | | |
|---|---|---|---|---|
| (Hz) | $x_4$ | $y_4$ | $x_5$ | $y_5$ |
| 1 | ±3 ; 0,3.5 | ±6 ; 0,7 | ±3 ; 0,3.5 | ±6 ; 0,7 |
| 2 | ±2.5 ; 0,2 | ±4.5 ; 0,4 | ±2.5 ; 0,1.9 | ±5 ; 0,4 |
| 5 | -3,2 ; 0,1.2 | -4,6 ; 0,2.25 | -3,2 ; 0,1.1 | -4,6 ; 0,2.4 |
| 10 | -3,2 ; 0,1 | -4,6 ; 0,2 | -3,2 ; 0,0.9 | -4,6 ; 0,2.1 |
| 30 | -3,2 ; 0,1 | -4,6 ; 0,2 | -3,2 ; 0,0.9 | -4,6 ; 0,2.1 |

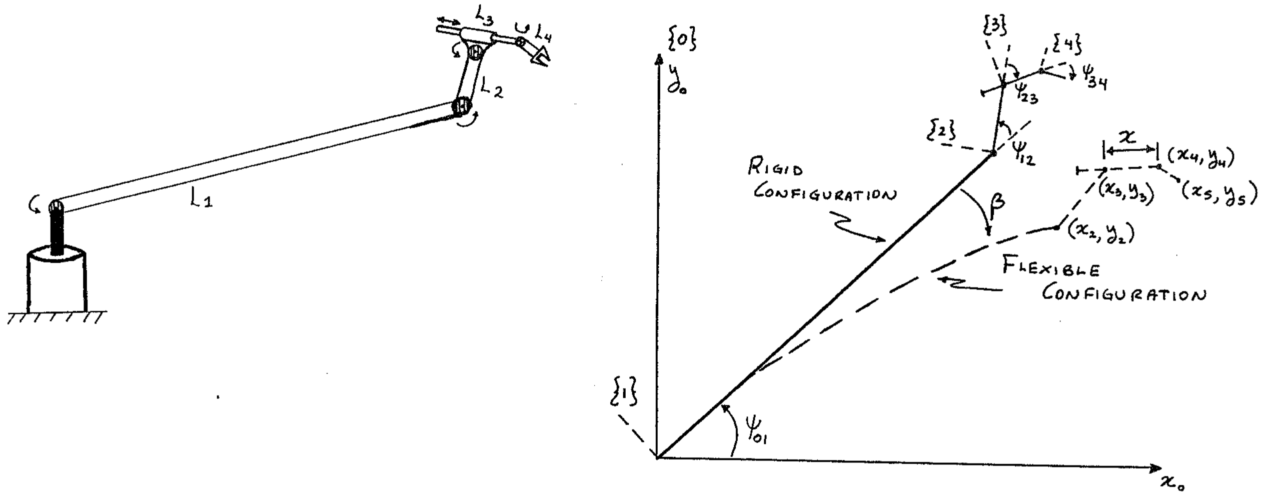


Figure 1. The Flexible Manipulator.

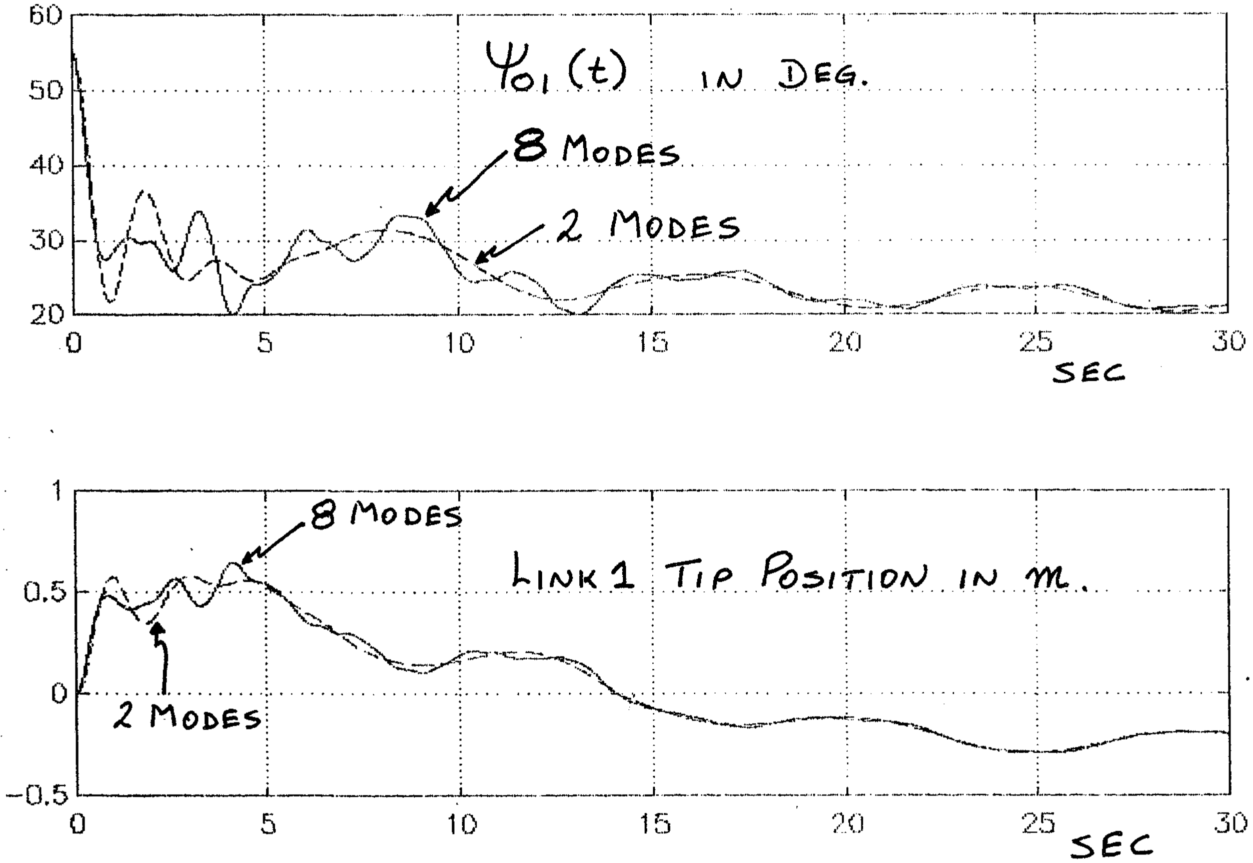


Figure 3. Hub Angle and Local Tip Position of Link 1.

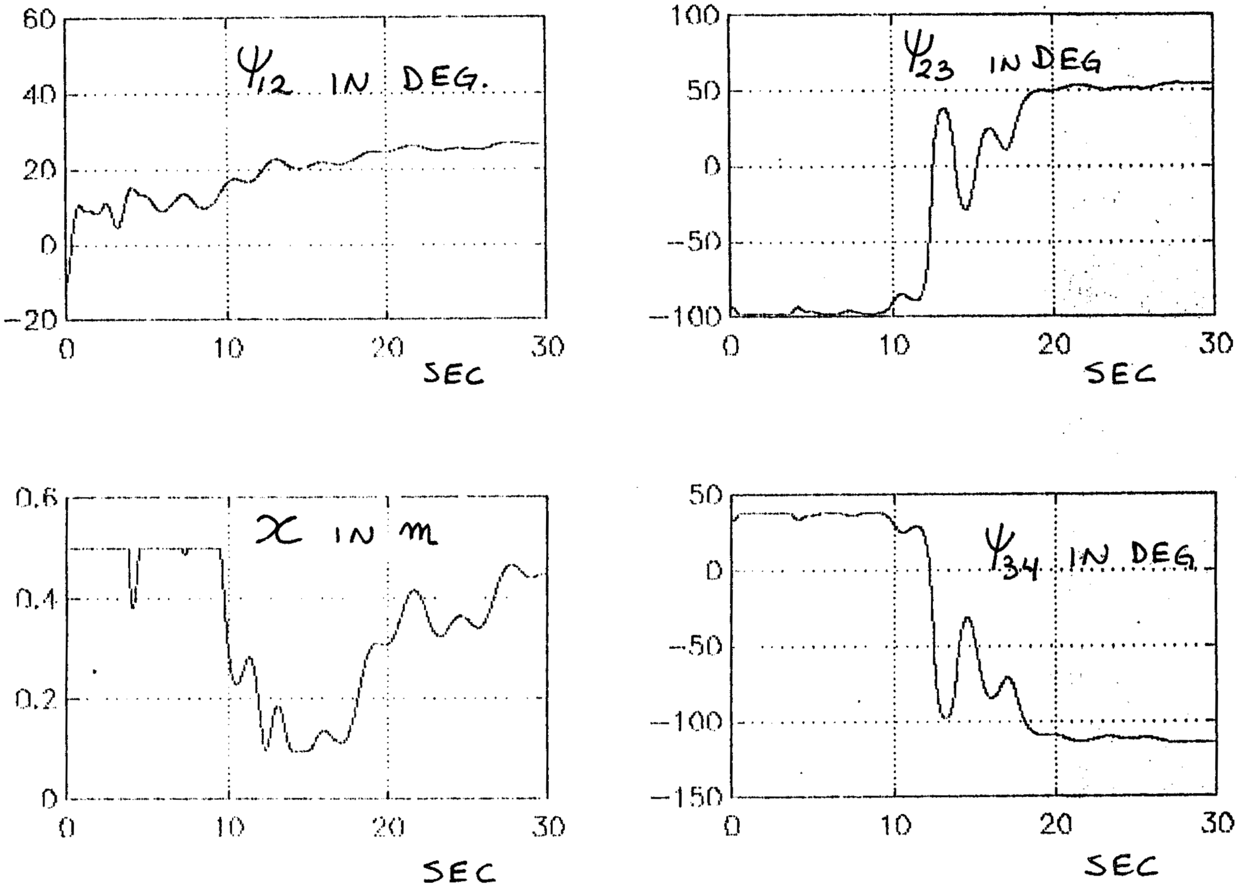


Figure 4. Joint Angles and Prismatic Displacement for a 4 mode Controller and 30 Hz Bandwidth.

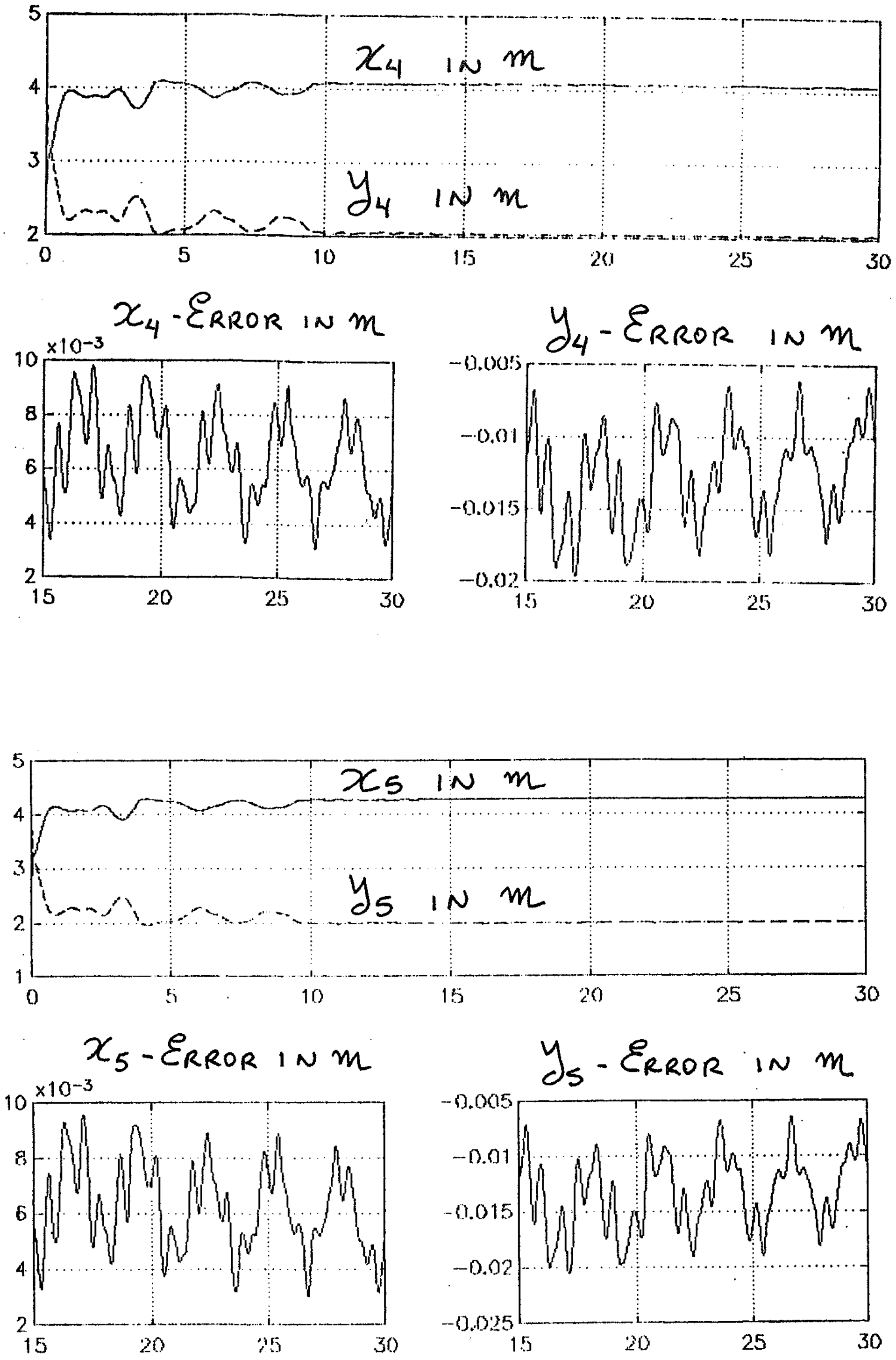


Figure 5. Coordinates of Link 4 and
Expanded View of Errors.

# SPECIAL TOPICS IN TELEOPERATION

# Preshaping Command Inputs to Reduce Telerobotic System Oscillations

**Neil C. Singer**
Research Assistant
Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA

**Warren P. Seering**
Associate Professor
Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA

## Abstract

This paper presents the results of using a new technique for shaping inputs to a model of the space shuttle Remote Manipulator System (RMS). The shaped inputs move the system to the same location that was originally commanded, however, the oscillations of the machine are considerably reduced. First, an overview of the new shaping method is presented. Second, a description of Draper Laboratories' RMS model is provided. Third, the problem of slow joint servo rates on the RMS is accommodated with an extension of the shaping method. Lastly, the results and sample data are presented for both joint and three-dimensional cartesian motions. The results demonstrate that the new shaping method performs well on large, telerobotic systems which exhibit significant structural vibration. The new method will be shown to also result in considerable energy savings during operation of the RMS manipulator.

## Introduction

Control of machines that exhibit flexibility becomes important when designers attempt to push the state of the art with faster, lighter machines. Many researchers have examined different controller configurations in order to control machines without exciting resonances. The input commands to these closed loop systems are "desired" trajectories. Often they are step inputs or trajectories that the machine cannot closely follow—especially if the commands are from human operators. The controllers treat the inputs as disturbances and try to eliminate the resulting oscillations. In fact, the energy which goes into a system in the form of unknown disturbances is typically small when compared with the energy inserted by the servo systems. Because we know quite precisely the character of the energy from the servos, we should be able to regulate it so that it does not cause vibration.

The approach of command shaping is designed to reduce the problems for the controller by altering the shape of the desired trajectory (teleoperator commands). The new, shaped trajectory is close to the original trajectory but does not cause vibration. In this paper a brief overview of vibration reduction control techniques is presented first. Second, input-shaping techniques are discussed. Third, a new method of residual vibration reduction is outlined. Fourth, Draper laboratories' software model of the space shuttle RMS (called the DRS) is discussed. This discussion will be used to motivate some extensions to the new method so that it may be used on teleoperated systems with slow servo rates. The remainder of this paper will then present the results of a series of experiments that were performed on the DRS.

## Vibration Reduction

Many researchers have examined feedback approaches to the control of flexible systems. Cannon and Schmitz [5], and Hollars [10] have examined the feedback of endpoint position measurements from a manipulator. Book [4] and Alberts [1] have examined feedback of strain gage measurements. Yurkovich [13] has examined acceleration feedback techniques for residual vibration reduction. Another approach is to include additional damping into the structure with additional actuators. Plump, Hubbard, and Bailey [18] examined the use of piezoresistive polymer films. Crawley [7] examined the use of a distributed array of piezoelectric devices for actuation on a structure. A more complete review of the control of flexible machines literature is given in Singer [19].

### Feedforward or Command Shaping

The earliest form of command preshaping was the use of high-speed cam profiles as motion templates. These input shapes were generated so as to be continuous throughout one cycle (ie. the cycloidal cam profile). Another early form of setpoint shaping was the use of posicast control by O.J.M. Smith [22]. This technique involves breaking a step of a certain magnitude into two smaller steps, one of which is delayed in time. This results in a response with a reduced settling time.

Optimal control approaches have been used to generate input profiles for commanding vibratory systems. Junkins, Turner, Chun, and Juang have made considerable progress toward practical solutions of the optimal control formulation for flexible systems [12][11][6]. Gupta [9], and Junkins and Turner [12] also included some frequency shaping terms in the optimal formulation. The derivative of the control input is included in the penalty function so that, as with cam profiles, the resulting functions are smooth.

Farrenkopf [8] and Swigert [23] demonstrated that velocity and torque shaping can be implemented on systems which modally decompose into second order harmonic oscillators. They showed that inputs in the form of the solutions for the decoupled modes can be added so as not to excite vibration while moving the system. Another technique is based on the concept of the computed torque approach. The system is first modeled in detail. This model is then inverted — the desired output trajectory is specified and the required input needed to generate that trajectory is computed [2][17].

Another approach to command shaping is the work of Meckl and Seering [14] [15]. They investigated several forms of feedforward command shaping. One approach they examined is the construction of input functions from either ramped sinusoids or versine functions. This approach involves adding up harmonics of one of these template functions in order to approach a time-optimal input. The harmonics that have significant spectral energy at the natural frequencies of the system are discarded. Aspinwall [3] proposed a similar approach which involves creating input functions by adding harmonics of a sine series. Singer and Seering [20] investigated an alternative approach of shaping a time optimal input by acausally filtering out the frequency components near the resonances.

## Brief Introduction to the New Shaping Method

A full derivation and analysis of this method can be found in Singer and Seering [21, 19]. Essentially, this technique involves generating an impulse input sequence (a command consisting entirely of impulses). The criterion for generating this sequence is that it should move an idealized system (a system with the same resonant frequency and damping ratio as the system that is intended to be controlled) without vibration. The reason that the system is considered idealized is because impulses are not physically realizable. The signals that are eventually given to the real system will not contain impulses and will, therefore, be realizable. An example of such an idealized sequence is shown in figure 1. If these two impulses are used as input to the ideal system, the oscillations of the system cancel and the system moves without vibration.

Singer [19] shows that the two-impulse input shown in figure 1 can be obtained by satisfying the equations

$$\begin{aligned} V_1 &= \sum_{j=1}^{N} A_j e^{-\zeta\omega(t_{end}-t_j)} \sin\left(t_j\omega\sqrt{1-\zeta^2}\right) = 0 \\ V_2 &= \sum_{j=1}^{N} A_j e^{-\zeta\omega(t_{end}-t_j)} \cos\left(t_j\omega\sqrt{1-\zeta^2}\right) = 0 \end{aligned} \tag{1}$$

where $A_j$ is the amplitude of the $j$th impulse, $t_j$ is the time of the $j$th impulse, and $t_{end}$ is the time at which the sequence ends (time of the last impulse), for the case when $N = 2$. The first impulse time and amplitude ($t_1$ and $A_1$) are not free variables. The time of the first impulse is fixed at zero and its amplitude can be arbitrarily chosen — linearity guarantees that the solution will scale with the value of $A_1$. This leaves two equations with two

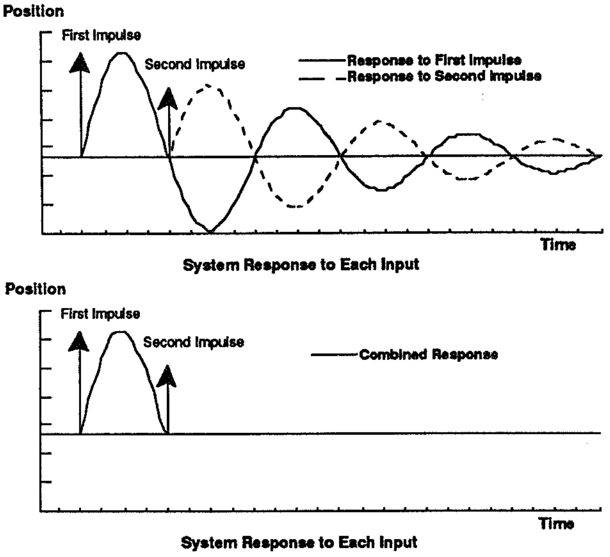


Figure 1: The two impulses shown, when given to an idealized system produce the two impulse responses shown. By superposition, the net response is that of the bottom plot.

unknowns ($A_2$ and $t_2$) which is solved in figure 1. Note that the amplitudes of the impulses were normalized so that they sum to unity.

Figure 2, however, shows that system response is extremely sensitive to variations in the natural frequency of the system. If a small uncertainty in the natural frequency exists (because of hardware considerations or nonlinearities) a great deal of residual vibration may be induced. On the plot of figure 2 this appears as a large error percentage for small excursions in the nondimensional natural frequency. A five percent level is indicated on the plot as a reference. An oscillation of less than five percent is often considered a fully "settled" system. The vibration error curve that is shown is a plot of

$$\frac{\sqrt{V_1^2 + V_2^2}}{C}$$

where $C$ is a system-specific constant.

This two impulse sequence lacks robustness. However, some additional constraints can be added when generating an impulse input sequence for the idealized system. The first constraint to be added is a requirement that the residual vibration error (the percentage of the move distance that becomes residual vibration) change slowly for uncertainties in the natural frequency and damping ratio of the system.

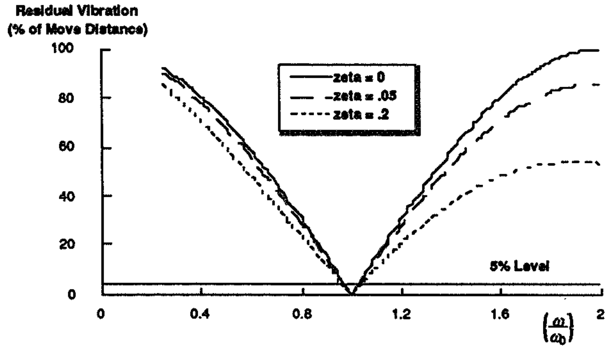


Figure 2: Plot of residual vibration amplitude (expressed as a percentage of the move distance) vs. the system's actual natural frequency. The impulse sequence is designed with the assumption that the system's natural frequency is unity (nondimensionalized).

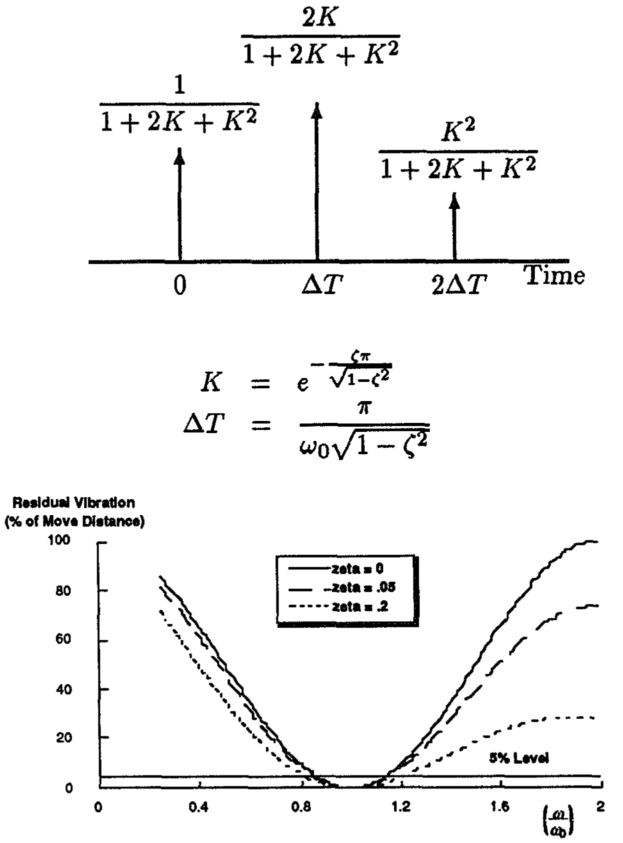
Figure 3 shows the resulting impulse sequence and the corresponding vibration error curve. Note that the slope of the vibration error curve at the anticipated natural frequency of the system ($\omega/\omega_0 = 1$) is zero. This can be interpreted as a form of robustness. As the frequency of oscillation varies from $\omega/\omega_0 = 1$, the vibration that is incurred at the end of the move does not significantly increase. The mathematical constraint for the additional robustness is given by

$$\begin{aligned} \sum_{j=1}^{N} A_j t_j e^{-\zeta\omega(t_{end}-t_j)} \sin\left(t_j\omega\sqrt{1-\zeta^2}\right) &= 0 \\ \sum_{j=1}^{N} A_j t_j e^{-\zeta\omega(t_{end}-t_j)} \cos\left(t_j\omega\sqrt{1-\zeta^2}\right) &= 0 \end{aligned} \tag{2}$$

By adding these two equations, a total of four equations are now solved simultaneously. Because four unknowns must be present, $N$ can be increased to 3, thus adding one additional impulse and two additional unknowns ($A_3$ and $t_3$). The solution of this system of equations is shown in figure 3.

This approach can be carried further by adding additional constraints and generating sequences with greater robustness and/or sequences for systems with multiple modes.[21]

The next step is to use this impulse sequence as a shaping template for input functions. Just as the single impulse is the elemental building block for arbitrary functions, the impulse sequence can be used as a building block for arbitrary vibrationless functions. The method for using the impulse sequence



$$K = e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}$$
$$\Delta T = \frac{\pi}{\omega_0\sqrt{1-\zeta^2}}$$

Figure 3: **Top:** Three-impulse input—designed to have a vibration-error expression which is both zero and tangent at the expected system natural frequency, $\omega_0$. $\zeta$ is the expected damping ratio. **Bottom:** Residual vibration amplitude (expressed as a percentage of the move distance) vs. the system's actual natural frequency. The impulse sequence is designed with the assumption that the system's natural frequency is $\omega_0$.

is to convolve it with any desired system input or trajectory. The resulting system response is similar to the requested trajectory but results in little or no residual vibration. This fact is offered without proof here, but is documented in [21, 19]. It is important to note that the signals that are sent to the system do not contain impulses once the convolution is performed. The signals are now physically realizable (assuming that the requested trajectory is physically realizable). At this point the restriction on needing an idealized system is dropped and the real system may be commanded. In addition, since the sequence consists of just three impulses, computation of the command signal is trivial.

## The DRS Space Shuttle Manipulator Model

Next, a detailed model of the Space Shuttle Remote Manipulator System (RMS) was adapted for this research. C. S. Draper Laboratories developed this complex model which they call the DRS (Draper Remote-manipulator Simulation). NASA uses the DRS to verify and test payload operations on the actual shuttle. The Draper shuttle manipulator model includes many of the complicating features of the hardware shuttle manipulator such as stiction/friction in the joints; nonlinear gearbox stiffness; asynchronous communication timing; joint freeplay; saturation; and digitization effects. The simulation was verified with actual space-shuttle flight data. Excellent agreement was obtained both for steady-state and for transient behavior. Approximately ten man-years of programming was invested in this model in order to assure that it accurately represents the actual shuttle hardware. It consists of approximately 14,000 lines of FORTRAN code (with 11,000 additional lines of comments).

The model was executed with twenty-two degrees of freedom. These include three rotational degrees of freedom for the space shuttle, five vibrational modes in each of the two long links, freeplay at the swingout joint and grapple point (between the arm and the payload), and seven degrees of freedom of the arm. The five vibrational modes in each long link are comprised of a first and second bending mode in two perpendicular directions, and one torsional mode. The four bending modes are modeled using an assumed cubic mode shape (figure 4).

This model was ideal as a test facility. It provided a repeatable, realistic environment for testing vibration suppression techniques. New concepts could be easily implemented in software without risking hardware. Additionally, new techniques could be inserted into the model at any location. On real hardware it is often difficult to implement a new concept because specialized hardware would either have to be altered or constructed.

First, a series of frequency tests were performed on the DRS model in order to understand the nature of its geometric nonlinear behavior. As the RMS moves throughout its workspace, its period of oscillation changes. An example of a map of natural frequency vs. joint angle is shown in figure 5. Note that the frequency of the first mode changes by approximately a factor of two over the workspace (when no payload is present). In addition the frequency shift is shown to be smooth and continuous. This fact is beneficial because the robustness of the new technique can accommodate reasonable shifts in frequency.

One obstacle to the use of the input shaping technique presented above on the DRS is the slow, .08 second servo rate of the RMS. The new technique assumed that both the amplitude and time of the impulse sequence could be precisely set. The next section discusses the effect of digitization and presents an alternative approach for generating a robust input shaping sequence.

## Digital Implementation

The derivation presented above assumed that the timing of the impulses (the times at which the requested input is repeated into the system) could be specified exactly. If the system is digital, the spacing of the impulses is at fixed intervals — multiples of the sampling rate. Figure 6 demonstrates this problem assuming that a three impulse input is used. The middle impulse falls directly in between two sampling intervals. This causes a timing error. This section evaluates how well this technique fares when the sampling induced error $\delta$ becomes large.

The vibration error due only to digitization can be calculated. The expression for the vibration amplitude that is induced by the digitization of the system is

$$\text{Error} \approx \frac{\delta t}{4\Delta T} \quad ,$$

where $\delta t$ is the sampling period, and $\Delta T$ is the half-period of the damped natural frequency of the sys-

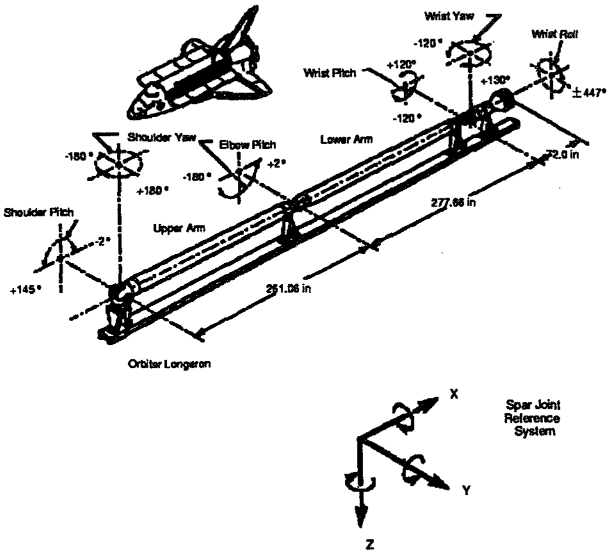


Figure 4: Space shuttle remote manipulator system joint reference coordinates.

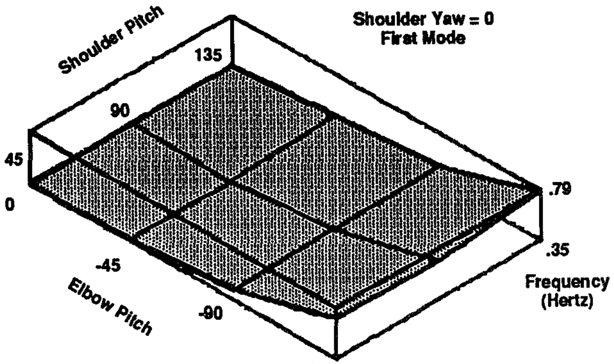


Figure 5: The first mode of the unloaded RMS as a function of shoulder pitch and elbow pitch. Shoulder yaw is fixed at 0° (the arm is moving in a vertical plane which includes the longeron).

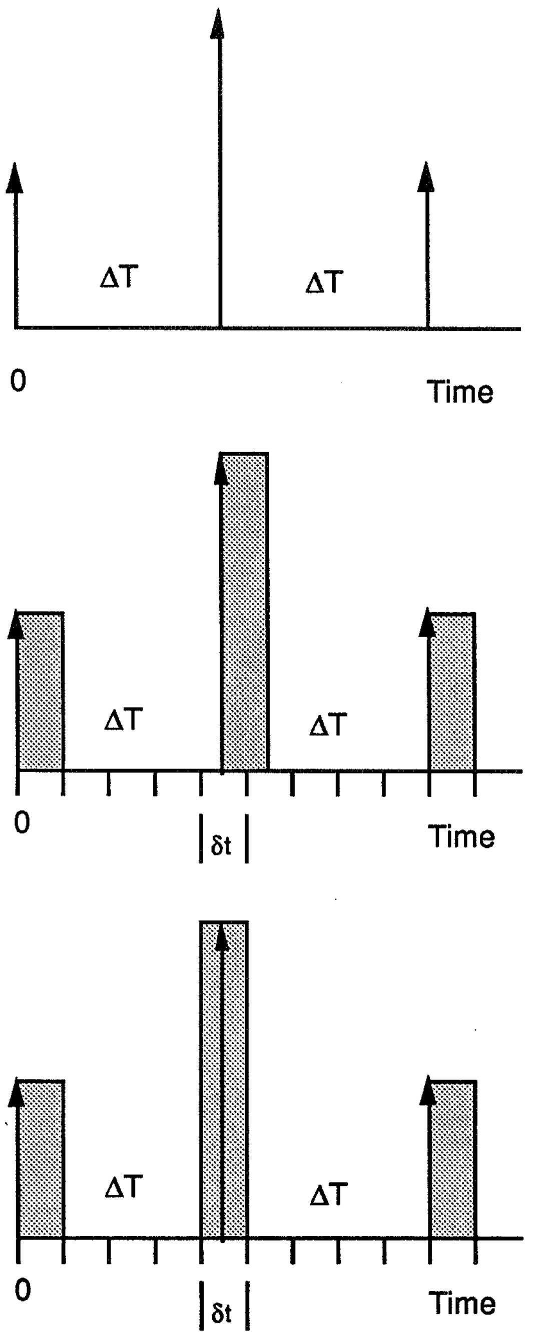


Figure 6: The problem of shaping inputs to digital systems. Top is the desired sequence. Middle: The digital timing of the system requires that the impulses do not all line up with the sampling intervals. Bottom: If the closest digital approximation is used (rounding to the nearest sampling interval), the impulse sequence is essentially translated as shown.

tem (the impulse spacing). This expression is derived in [19]. If this fraction is small for a particular digital system, then the digitization of the system can be ignored, and the impulses can be moved to the nearest sampling interval without inducing a significant vibration penalty. Small values for the error are typically less than 5% – 10% (corresponding to a 5% – 10% vibration of a digitized simple harmonic oscillator commanded with a step).

### Sequences for Digital Systems

Once it has been determined that the error due to digitization is unacceptably large, a new form of the input sequence must be generated. This input sequence is constructed from impulses which occur at integral multiples of the sampling interval.

Figure 7 shows a sequence generated for the space shuttle RMS by solving the four equations (1 and 2) which were used to generate the three-impulse sequence shown in figure 3 with the additional constraint that forced the impulses to occur at multiples of the sampling period. Because the same design criteria are met, this five impulse sequence has the same vibration-reducing and robustness properties of the three impulse sequence derived above. The additional impulses adjust for the timing constraints. Essentially, several impulses that have been constrained to be on sampling intervals are adding to form the impulse that we would like to have had which is not on a sampling interval.

## Solving for the Sequence

Because there are more unknowns in this solution than constraint equations, a minimization routine was used to generate the impulse sequence of figure 7. First, an impulse amplitude ($A_j$) was assigned to each sampling interval of the digital system. The length of the system was chosen (in figure 7 the length is 22 – 1.76 seconds at .08 second sampling) The values for the $A_j$ are then determined using a simplex algorithm. The second derivative expression of equation 1 with respect to $\omega$, given by:

$$\begin{aligned} \sum_{j=1}^{N} A_j(t_j)^2 e^{-\zeta\omega(t_{end}-t_j)} \sin\left(t_j\omega\sqrt{1-\zeta^2}\right) &= 0 \\ \sum_{j=1}^{N} A_j(t_j)^2 e^{-\zeta\omega(t_{end}-t_j)} \cos\left(t_j\omega\sqrt{1-\zeta^2}\right) &= 0 \end{aligned} \quad (3)$$

is then minimized subject to the following constraints:

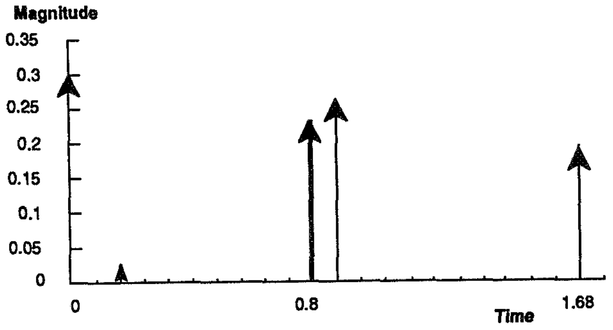


Figure 7: Robust digital sequence. This sequence meets the constraint that requires that the system have no residual vibration when the input has ended. Additionally, this sequence meets the robustness constraint that requires the rate of change of the vibration with respect to changes in natural frequency be zero. Therefore, small uncertainties in the parameters of the system (ie. natural frequency) do not cause an appreciable increase in residual vibration.

- The sinusoidal part of the vibration amplitude expression equals zero (the first equation of 1).
- The cosinusoidal part of the vibration amplitude expression equals zero (the second equation of 1).
- The sinusoidal part of the first derivative $(\frac{d}{d\omega})$ expression equals zero (the first equation of 2).
- The cosinusoidal part of the first derivative $(\frac{d}{d\omega})$ expression equals zero (the second equation of 2).
- The magnitude of the impulse amplitudes must be less than a limit ($A_j <=$ Limit)
- The sum of the impulse amplitudes are unity.

Note that many of the amplitudes will be zero. The length of sequence ($N$ in the equations above) is reduced until the constraint equations can no longer be satisfied. In the space shuttle example, 22 was the shortest sequence length for which a solution was possible. The resulting solutions are theoretically exact. If the system were to be exactly as modeled, the response to the input would be totally without vibration. The digital timing of the system is already included in the derivation, therefore, the digitization does not alter the vibration-reducing effects.

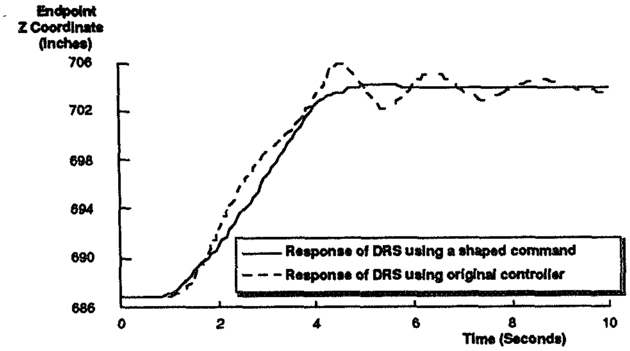


Figure 8: Comparison between the RMS controller and a controller that shapes inputs with a three-impulse equivalent sequence.

## Results on the DRS Model

The shaped command of the previous section was next tested on the computer model of the Space Shuttle Remote Manipulator System (RMS). Figure 8 shows a comparison between the response of the DRS using the current shuttle RMS controller and the response of the DRS using an input that was shaped by the sequence of figure 7 as a velocity input. The requested (unshaped) input was a step to maximum velocity followed by a step back to zero velocity. The residual vibration is reduced by more than one order of magnitude (a factor of 25) for the unloaded shuttle arm. Comparable results were obtained for a variety of moves tested.

## Multiple Joint Actuation

### Linear Systems

One important question that must be addressed in using this technique is the effect of simultaneously shaping inputs to two separate joints of a machine. The technique would be of limited utility if it could only be used on single joint machines. This last section will discuss the effect of shaping several machine inputs simultaneously.

Because the resonances of the system are configuration dependent and are independent of the joint that is to be actuated, only one shaping sequence is used on all of the joints of a system. Additionally, Singer [19] shows (for linear systems) that the shaping of inputs for one axis of a machine cannot interfere with the shaping at any other axis.

## Vibrationless Cartesian Motion from Non-Cartesian Machines

When the system that is to be controlled is a cartesian machine, the technique presented above applies. However, often the system is not cartesian and, therefore, straight line motion must be achieved by computation of joint trajectories for cartesian motion. The problem that will be addressed in this section is the effect of shaping on the overall endpoint trajectory of the machine.

Two main approached have been examined. The first was to determine the straight line joint trajectories that would be required assuming that the signals were not to be shaped. Next, these joint trajectories were shaped so that they become vibration-reducing. The advantage of this approach is that the vibration control is the best possible (keeping all other factors constant). The disadvantage is that the trajectories are not theoretically exact straight line trajectories. However, the original "straight-line" trajectories are not perfectly straight either [16]. Intermediate points are computed on a straight-line trajectory and non-straight, joint-interpolated motion is used between these points. Therefore, cartesian trajectories in practice are only as straight as the available computation allows (Paul [16] states that joint-interpolated motion requires roughly 1% of the computation of cartesian motion). Shaping the trajectory does not significantly alter the cartesian nature of the input, especially as more intermediate points are used. Additionally, since the shaped trajectory does not have the unwanted vibration in the output, the actual endpoint position will most likely be closer to "straight" than the unshaped trajectory. (If the vibration was not causing problems, shaping would never have been considered for that system!)

The second approach is to shape the cartesian trajectories and then convert them to joint trajectories. This approach guarantees that the trajectories will be as straight as possible (keeping all other factors constant). The drawback of this approach is that the vibration reduction is slightly degraded.

For the DRS space shuttle manipulator model, the first configuration was used. The joint trajectory was calculated from the commanded cartesian motion from the teleoperator. Next, the joint trajectories were shaped at each joint with the same shaping sequence. Figure 9 shows a cartesian move on the space shuttle arm. The joint trajectories are calculated first and then are shaped. The plot shows the motion in the y-direction. Figure 10 shows the x and z direction motion for the same cartesian move. The command is only in the y-direction. These plots demonstrate that even without the preshaping, the shuttle's "cartesian" motion is not straight, and the vibration amplitude effects the straightness of the motion to a much larger extent than the alteration of the joint trajectories caused by shaping.

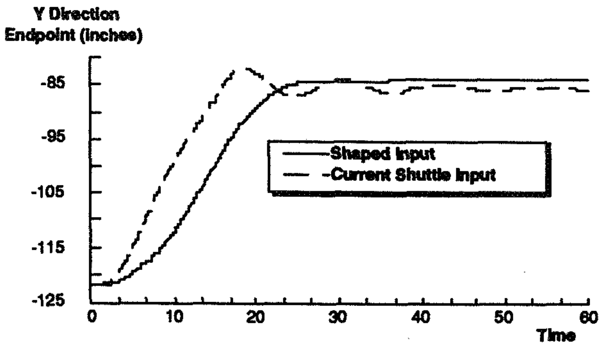


Figure 9: Cartesian motion of the shuttle manipulator. The command to the shuttle was a straight-line motion (step) in the y direction. The dashed line is the unaltered RMS controller. The solid line is a shaped input. The data is shown is motion in the y direction. The next figure shows the x and z motion during the same move.

Figure 11 shows a comparison of the energy consumed by the shuttle manipulator during the two moves. A 20% savings in energy was realized by not inducing vibration in the arm. This energy savings has significant implications for space systems like the shuttle and space station. Since energy in space is expensive (the shuttle, for example must carry its own fuel) the energy savings alone may justify shaping of the command input for the reduction of vibration.

## References

[1] **Alberts, T.E., Hastings, G. G., Book, W.J., and Dickerson, S.L.,** "Experiments in Optimal Control of a Flexible Arm with Passive Damping", *Fifth VPISSU/AIAA Symposium on Dynamics and Control of Large Structures*, June 12, 1985,. Blacksburg, VA,, pp 423-435

[2] **Asada, Haruhiko; Ma, Zeng-Dong; and**

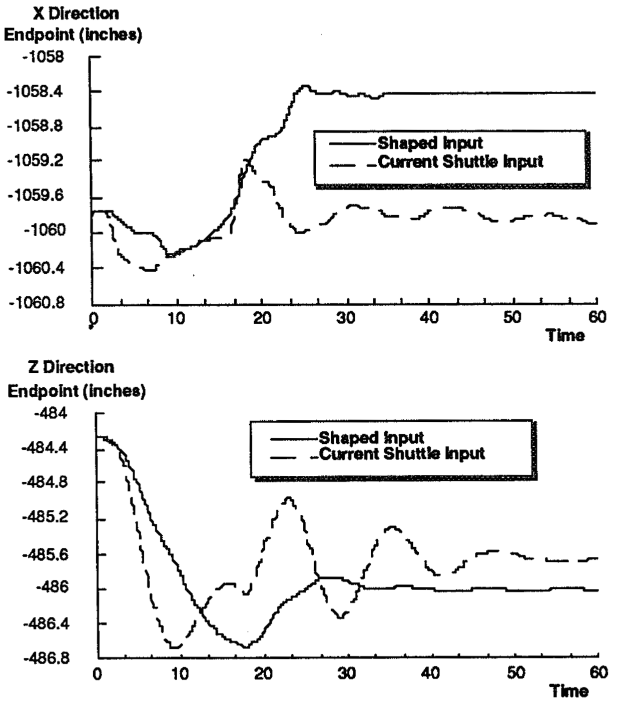


Figure 10: Cartesian motion of the shuttle manipulator. The command to the shuttle was a straight-line motion (step) in the y direction. The dashed line is the unaltered RMS controller. The solid line is a shaped input. The motion perpendicular to the commanded motion is shown. The previous figure shows the motion in the commanded direction. Note that neither move (with or without shaping) is extremely "straight" on the DRS.

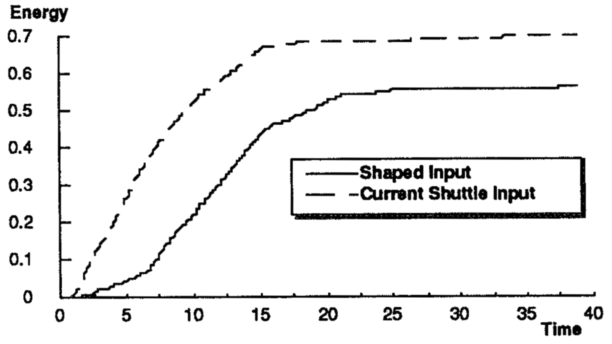


Figure 11: Energy plot for straight-line motion. This figure demonstrates the energy savings by using the new shaping technique.

**Tokumaru, Hidekatsu,**
"Inverse Dynamics of Flexible Robot Arms for Trajectory Control.", *Modeling and Control of Robotic Manipulators.* , 1987 ASME Winter Annual Meeting.. pp 329–336

[3] **Aspinwall, D. M.,**
"Acceleration Profiles for Minimizing Residual Response", *Journal of Dynamic Systems, Measurement, and Control,* March 1980,. Vol. 102, No. 1,, pp 3–6

[4] **Book, Wayne J.; and Cetinkunt, Sabri,**
"Near Optimal Control of Flexible Robot Arms on Fixed Paths", *Proceedings of the 1985 IEEE International Conference on Robotics and Automation,* April 1985,. St. Louis, MO,, pp 1522–28

[5] **Cannon, Robert H. and Schmitz, Eric.,**
"Initial Experiments on the End-Point Control of a Flexible One Link Robot", *The International Journal of Robotics Research,* Fall 1984,. Vol. 3, No. 3.,

[6] **Chun, Hon M.; Turner, James D.; and Juang, Jer-Nan,**
"Disturbance-Accommodating Tracking Maneuvers of Flexible Spacecraft", *Journal of the Astronautical Sciences,* April-June 1985,. Vol. 33, No. 2,, pp 197-216

[7] **Crawley, Edward F., de Luis, J.,**
"Experimental Verification of Distributed Piezoelectric Actuators for use in Precision Space Structures", *Structures, Structural Dynamics and Materials Conference,* May 19–21, 1986,. San Antonio, TX,, *AIAA Paper No. 86-0878,* pp 116-124

[8] **Farrenkopf, R.L.,**
"Optimal Open-Loop Maneuver Profiles for Flexible Spacecraft", *Journal of Guidance and Control,* Vol. 2, No. 6,, November–December 1979,. pp 491–498

[9] **Gupta, Narendra K.,**
"Frequency-Shaped Cost Functionals: Extension of Linear-Quadratic-Gaussian Design Methods", *Journal of Guidance and Control,* Vol. 3, No. 6,, Nov.-Dec 1980,. pp 529–35

[10] **Hollars, Michael G. and Cannon, Robert H.,**
"Experiments on the End-Point Control of a Two-Link Robot with Elastic Drives", *AIAA Paper 86-1977,* 1986..

[11] **Juang, Jer-Nan; Turner, James D.; and Chun, Hon M.,**
"Closed-Form Solutions for Feedback Control with Terminal Constraints", *Journal of Guidance and Control,* January–February 1985,. Vol. 8, No. 1,, pp 39–43

[12] **Junkins, John L.; Turner, James D.,**
"Optimal Spacecraft Rotational Maneuvers", Elsevier Science Publishers, New York,, 1986..

[13] **Kotnik, P. T.; Yurkovich, S.; and Ozguner, U., ,**
"Acceleration Feedback for control of a flexible Manipulator Arm", *Journal of Robotic Systems,* Vol. 5, No. 3,, June 1988,. pp (to appear)

[14] **Meckl, P. and Seering, W.,**
"Minimizing Residual Vibration for Point-to-point Motion", *ASME Journal of Vibration, Acoustics, Stress, and Reliability in Design,* Vol. 107, No. 4,, October, 1985,. pp 378-382

[15] **Meckl, Peter H., and Seering, Warren P.,**
"Controlling Velocity–Limited Systems to Reduce Residual Vibration", *Proceedings of the 1988 IEEE International Conference on Robotics and Automation,* April 25–29, 1988,. Philadelphia, Pennsilvania.,

[16] **Paul, Richard P.,**
"Robot Manipulators", MIT Press,. Cambridge, MA,, 1982,. pp 139–43

[17] **Pfeiffer, F., Gebler, B.,**
"A Multistage-Approach to the Dynamics and Control of Elastic Robots", *Proceedings of the 1988 IEEE International Conference on Robotics and Automation,* April 25–29, 1988,. Philadelphia, Pennsilvania,, pp 2-8

[18] **Plump, J. M.; Hubbard, J. E.; and Bailey, T.,**
"Nonlinear Control of a Distrubuted System: Simulation and Experimental Results", *Journal of Dynamic Systems, Measurement, and Control,* June, 1987,. Vol. 109, pp 133-139

[19] **Singer, Neil C. ,**
"Residual Vibration Reduction in Computer Controlled Machines", Massachusetts Institute of Technology, PhD Thesis, January, 1989..

[20] **Singer, Neil C.; Seering, Warren P. ,**
"Using Acausal Shaping Techniques to Reduce Robot Vibration", *Proceedings of the 1988 IEEE International Conference on Robotics and Automation,* Philadelphia, PA,, April 25–29, 1988,. pp 1434–7

[21] **Singer, Neil C.; Seering, Warren P. ,**
"Preshaping Command Inputs to Reduce System Vibration", **AIM No.,1027**, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts. May, 1988..

[22] **Smith, O.J.M.,**
"Feedback Control Systems", McGraw-Hill Book Company, Inc., New York,, 1958..

[23] **Swigert, C.J.,**
"Shaped Torque Techniques", *Journal of Guidance and Control,* September-October 1980,. Vol. 3, No. 5,, pp 460-467

# Performance Constraints and Compensation For Teleoperation With Delay

J. S. McLaughlin
B. D. Staunton
The Aerospace Corporation
El Segundo, CA 90245

Abstract:

A classical control perspective is used to characterize performance constraints and evaluate compensation techniques for teleoperation with delay. Use of control concepts such as open and closed loop performance, stability, and bandwidth yield insight to the delay problem. Teleoperator performance constraints are viewed as an open loop time delay lag and as a delay-induced closed loop bandwidth constraint. These constraints are illustrated with a simple analytical tracking example which is corroborated by a real-time, man-in-the-loop tracking experiment. The experiment also provides insight to those controller characteristics which are unique to a human operator. Predictive displays and feedforward commands are shown to provide open loop compensation for delay lag. Low pass filtering of telemetry or feedback signals is interpreted as closed loop compensation used to maintain a sufficiently low bandwidth for stability. A new closed loop compensation approach is proposed that uses a reactive (or force feedback) hand controller to restrict system bandwidth by impeding operator inputs.

## I) INTRODUCTION

This paper illustrates teleoperation performance constraints caused by time delays and discusses delay compensation methods. Use of control concepts such as open and closed loop performance yields insight to the delay problem. Teleoperator performance constraints are viewed as an open loop time delay lag and as a delay-induced closed loop bandwidth limitation. The work is motivated by real-time ground control of spacecraft, including remote piloting, docking, refueling, and servicing with broader application to any system faced with delay-induced performance degradation.

A typical architecture for ground control of spacecraft operations is pictured in Figure 1. A satellite is to perform a mission such as docking or servicing and is controlled in real-time by operators at a ground console through a communications link. Time delays result from signal time-of-flight and queing and processing at each of the nodes. For Earth-orbiting satellites, the node delays are dominant. This problem is represented in the following by separating controller commands from the plant by an uplink delay and the plant measurements from the controller by a downlink delay (Figure 2).

Time delays affect teleoperator performance, ultimately destabilizing the system. The significance of these effects depends on the magnitude of the delay with respect to the bandwidth of the tasks involved. The emphasis of this paper is on the closed loop performance limitations caused by the delay. This paper offers a perspective from which to clarify performance issues and evaluate compensation approaches. System design considerations have also emerged from this perspective.

In Section II, classical control concepts are used to give a heuristic explanation of the performance degradation due to delay. An analytical example is introduced in Section III to quantify the stability constraint for a simple system. This effect is illustrated by a "stability boundary" in a plot of loop delay versus bandwidth showing the upper bound on closed loop bandwidth caused by the delay. A man-in-the-loop experiment is described in Section IV, whose results are consistent with the analytical findings. The experiment also provides insight to the controller characteristics of a human operator. Methods to compensate for delay-induced performance degradation are discussed in Section V. In particular, the utility of dynamic predictors, telemetry filtering, and predictive displays is addressed. In addition, a new closed loop compensation approach is proposed that uses a reactive (or force feedback) hand controller to restrict system bandwidth by impeding operator inputs.

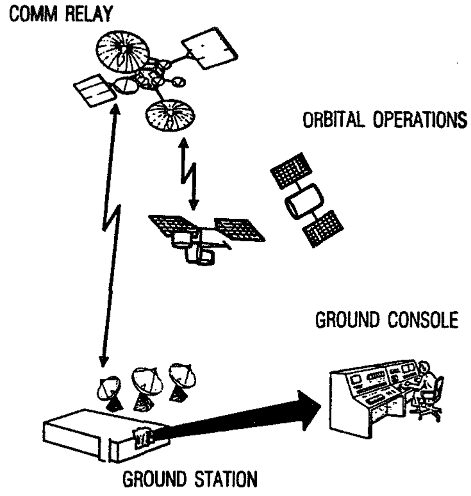


Figure 1: Ground Control of Spacecraft

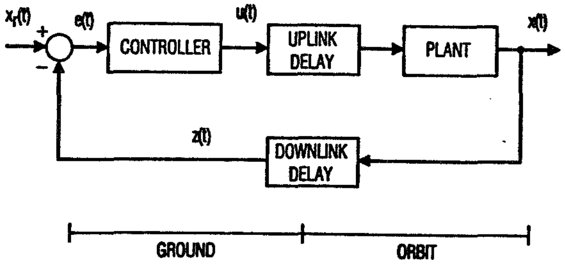


Figure 2: Idealized Block Diagram of Tracking with Delay

## II) EFFECTS OF DELAY

The purpose of this section is to focus on a closed loop performance issue in teleoperation with delay and describe a way to characterize it. This is accomplished by making a distinction between open and closed loop control in teleoperation and providing intuitive explanations for the effects of time delay on each. From this, a bandwidth constraint is formulated and related to physical principles.

It has been useful in this work to consider open and closed loop commanding separately when investigating the effects of delay. An open loop command depends only on knowledge of the initial state (or measurement) of the system and is *independent* of the current measurement of the system. A closed loop command is generated from the current state through feedback, usually as corrections to errors from the desired state of the system.

In open loop control, models of the system and environment are used to compute open loop commands which drive the system in a desired fashion. An unmodeled delay introduces position errors resulting from lags in the system response. This can be corrected for by including the time delay in the model used to generate the commands. In addition, unmodeled disturbances introduce position errors. In fact, *all* errors from the desired path are a result of inaccuracies in the models used to generate the open loop commands. Open loop control cannot compensate for unknown characteristics of the system.

Closed loop control is used to improve tracking performance in the presence of model uncertainties and/or disturbances. A closed loop command is generated by a control law that operates on system errors determined by measurements of the state of the system. In the experiment described below, the control law is a human operator generating commands while watching a video display . Closed loop stability is intuitively related to error correction: if the errors are changing quickly relative to the delay, then using a relatively ″old″ measurement to make a correction is ineffective and could compound the errors (i. e., become unstable). The controller is only able to make stable corrections for errors that change slowly with respect to the loop delay. In other words, the bandwidth of the controller is limited. This in turn will limit overall system tracking or disturbance rejection performance.

This intuitive argument on the destabilizing effect of time delays can be related to the fundamental principle of the causality of physically realizable systems, which means that physically realizable systems cannot know the future. A closed loop controller that alleviates the bandwidth limitation will have some component that requires knowledge of a future state and therefore cannot be implemented. A controller that *predicts* the future state based on a model of the system and knowledge of the current state is an open loop controller and will not improve the closed loop bandwidth limitation.

## III) STABILITY BOUNDARY: ANALYTICAL EXAMPLE

This section describes the procedure for arriving at quantitative delay-induced performance constraints in the form of a stability boundary. The effect of time delays is considered for deterministic, continuous, linear, time-invariant plants and full state feedback (FSFB), constant gain controllers. Time delays in the up and downlinks result in a transcendental characteristic equation which is solved and the dependency on total loop delay shown. The distribution of delays between the up and downlink segments are shown to have no effect on systems with this type of controller.

These systems can be represented in state space form as:

$$\dot{x}(t) = Ax(t) + Bu(t - T_u) \tag{1}$$

$$u(t) = -Kx(t - T_d) \tag{2}$$

where A is the plant dynamics matrix, B is the control distribution matrix, K is the feedback gain matrix, x is the state vector, and u is the control signal. Use of FSFB assumes that the entire state vector is available for control. $u(t\text{-}T_u)$ in equation (1) indicates that the plant dynamics are influenced by a control signal that has been delayed by $T_u$ seconds in the uplink. $x(t\text{-}T_d)$ in equation (2) shows that the current control signal is computed from state measurements delayed by $T_d$ seconds in the downlink.

Using Laplace transforms, these equations can be represented in the frequency domain as:

$$sx(s) = Ax(s) + Be^{-sT_u}u(s) \tag{3}$$

$$u(s) = -Ke^{-sT_d}x(s) \tag{4}$$

where s is the independent frequency parameter and $e^{-sT}$ is the Laplace transform of a pure time delay of T seconds.

Some matrix algebra results in:

$$[sI - (A - BKe^{-s(T_u + T_d)})]x(s) = [0] \tag{5}$$

where I is the identity matrix, [0] is the zero matrix of appropriate dimensions, and multiplication of the exponential delay terms results in the sum of the delays in the exponent. Note that for the case of FSFB, only the *total* loop delay is significant, not the location of the respective delays.

In order for a non-zero solution of x(s) to exist, the coefficient matrix in equation (5) must be singular over all frequencies s. This can be stated by requiring the determinant of the matrix to be equal to zero:

$$\det[sI - (A - BKe^{-s(T_u + T_d)})] = 0 \tag{6}$$

Equation (6) is known as the characteristic equation for the feedback system represented in equations (1) and (2). Stability of the system is guaranteed if the real part of all roots of the characteristic equation are negative (Reference 1). Although the plant dynamics matrix A is finite dimensional, the system is infinite dimensional because of the time delay (as seen by Taylor series expansion of the exponential term in (6) which results in a polynomial in s with *infinitely* many terms). This complicates the solution of the characteristic equation and thus the determination of stability.

The characteristic equation for a simple second order system is solved here to illustrate delay-induced performance constraints. The system is representative of a positioning device with unit mass load and constant gain FSFB through a delay T. All stable roots of the characteristic equation are found (see Appendix) in order to compute an upper bound on system bandwidth as:

$$r < \frac{\cos^{-1}(\zeta)}{T\sqrt{1 - \zeta^2}} \quad , \quad \zeta \in (0, 1) \tag{7}$$

where $\zeta$ is a damping parameter that governs transient response behavior such as overshoot and settling time. The bandwidth is defined here as the magnitude r of the smallest root of the characteristic equation in (6). (The traditional definition of the -3 db crossover frequency in the closed loop transfer function differs slightly from that used in this paper depending on the damping of the fundamental response.)

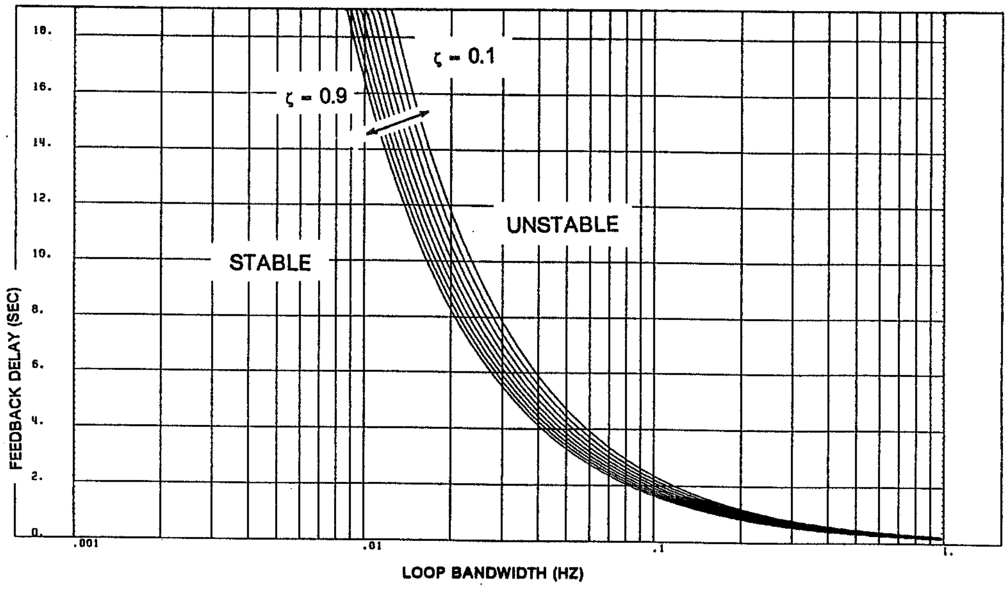


Figure 3: Stability Boundary for Second Order Example

Equation 7 is an inverse relationship between maximum system bandwidth and loop delay that is plotted in Figure 3 as a family of stability boundaries parameterized by ζ. For a given delay, controller parameters should be selected to ensure that the system bandwidth is low enough to fall within the stable region in Figure 3. Other criteria will influence the control design as well; this example serves to illustrate only the effects of time delay on closed loop performance.

## IV) MAN-IN-THE-LOOP EXPERIMENT

A man-in-the-loop tracking experiment was conducted to test the closed loop stability boundary result of the analytical example. This section describes the experiment set-up and data gathering procedure, presents the significant results, and compares the findings to those from the analytical example.

The experiment involves the simulated motion of two vehicles, a graphic display of one (the target) as viewed from the second (the tracker), and a human operator. The operator sees the target from the perspective of the tracker. The operator inputs commands through a keyboard while watching the display which is driven by a real-time simulation of the tracking dynamics and delays. The total delay includes uplink and downlink delays, and an additional delay due to an inherent lag in the operator's response.

The target moves side-to-side in a one-dimensional motion perpendicular to the target/tracker line-of-sight. The target motion is oscillatory with random peak amplitudes and a frequency upper bound that is successively increased to find the stability boundary. Random amplitudes are used to prevent the operator from predicting the motion. Steps are taken to eliminate operator use of a fixed reference point (the zero crossing point of the target oscillatory motion).

The operator commands the tracking vehicle to follow the target while watching the relative vehicle positions in the display. Two keyboard keys are used to apply discrete velocity control in one dimension. To avoid corruption of the stability results, steps are taken to provide the operator with sufficient control authority for tracking all target motion frequencies.

The bandwidth constraint for this man-in-the-loop system is inferred from the maximum target frequency the operator is able to successfully track. The target motion frequency and uplink and downlink time delays are prescribed for each run which lasts 100 to 300 seconds, depending on the period of target motion. The data gathering procedure allows for a number of practice runs for each time delay/motion frequency case; the best run is saved. Position-time histories of both vehicles are recorded and plotted after each run.

Interpreting the tracking results in terms of stability can be difficult. Unlike an analytical controller, an experienced human operator does not go unstable in the classical sense. When faced with target motion too fast for the given time delays, the operator chooses to issue very few commands and waits for the target to return rather than chasing it (this is why it is important to eliminate the fixed reference point). The operator learns to lower his bandwidth sufficiently for stable tracking, eventually resorting to a move and wait strategy. In this process, tracking is significantly degraded. The experiment results are therefore interpreted as one of three qualitative tracking assessments: "unsatisfactory" tracking is used when the operator ceases to follow the target (described above), "satisfactory" tracking is used to indicate good performance in the traditional sense (small error), and "marginal" tracking is applied to cases in between.

Figure 4 shows example position-time history plots of the target and three separate tracking runs. The maximum target frequency is 0.03 hertz frequency and the three runs have delays of 0, 2, and 4 seconds. Note that tracking performance gets worse as delays are increased. The three cases are interpreted as "satisfactory," "marginal," and "unsatisfactory" tracking.

Time delay/target frequency cases were attempted over the envelope defined by the analytical results. Figure 5 shows results in terms of "unsatisfactory," "marginal," and "satisfactory" tracking, overlayed on a plot of the stability boundary from the analytical example. The experiment results are consistent with the analytical example: the teleoperator is unable to track outside the stable region. The three cases from Figure 4 are denoted in Figure 5 by triangles.

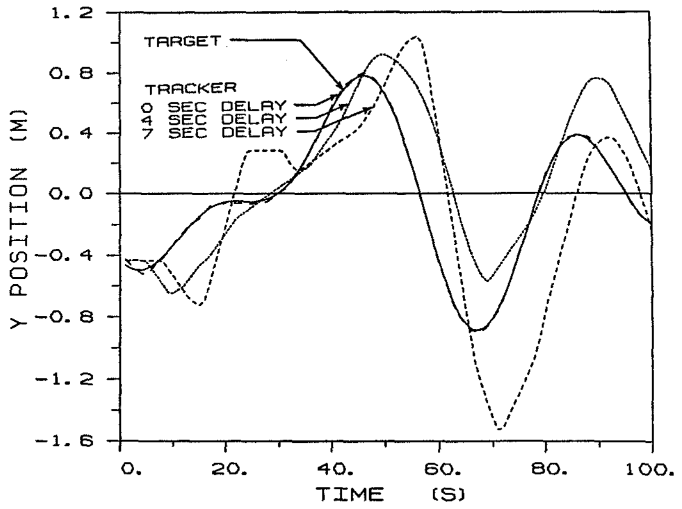


Figure 4: Man-in-the-Loop Tracking With Delay

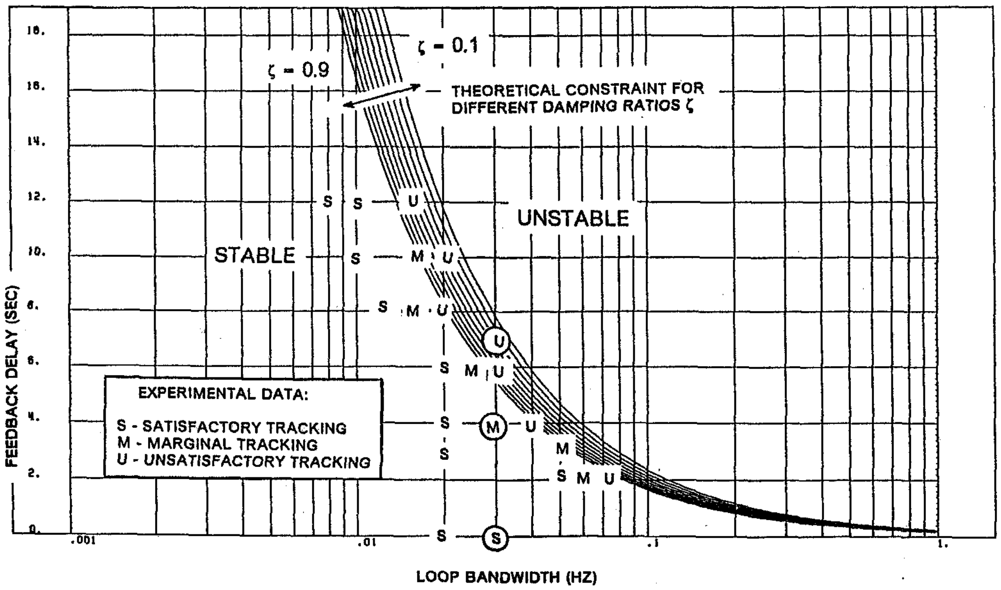


Figure 5: Stability Boundary With Man-in-the-Loop

## V) DELAY COMPENSATION

The purpose of this section is to discuss delay compensation schemes that alleviate the performance degradation described above. Delay compensation approaches are described while maintaining the distinction between open and closed loop that has been useful in characterizing performance constraints. The goal of closed loop compensation is to maintain a stable closed loop bandwidth as close to the stability boundary as prudent while the aim of open loop compensation is to speed up the large motion segments of a task.

### *Closed Loop Compensation*

A bandwidth constraint means that closed loop compensation must ensure the system bandwidth is maintained in the stable region and as close to the boundary as prudent to maximize performance. One way of limiting bandwidth is to use some sort of low-pass filtering on the up or downlink telemetry.

An idea introduced here is to use an active hand controller to lower the system bandwidth by impeding operator inputs. This effectively lowpass filters operator commands with enough impedance to ensure stability given the time delay. It is anticipated that the operator would not "fight" the stick to try and overcome the resistance but rather "feel" the effect of the time delay through the responsiveness of the stick and modulate his inputs accordingly. The desired result is to replace a move-and-wait strategy with slow, smooth commands generated from what is essentially a continuous "display" of time delay information (through stick impedance). The authors are considering experiments using hardware much like that used in force-reflecting hand controllers to investigate this form of delay compensation.

An alternative to impeding operator inputs is to display time delay information graphically. Experience with the man-in-the-loop tracking experiment suggests that knowledge of the time delay is beneficial. When motions are large, the delay is easily perceived from the lag between command and response. A graphical display of the time delay would provide the operator with delay information even when motions are small.

An additional consideration is that delays in a communications loop can change with time. In the tracking experiments, the operator appeared to be able to learn the (constant) delay over several trials with a marked improvement in performance. A delay that changes with time is likely to impede this learning, possibly with the operator retreating to some upper bound on the time delay variation.

The telemetry filter lowpass bandwidth and the hand controller impedance can be functions of the current time delay, known through time-tagged telemetry. This enables closed loop performance to be maximized even in the presence of time variable delays.

### *Open Loop Compensation*

Although closed loop performance is limited by the bandwidth constraint, performance gains can be had with open loop (or feedforward) compensation in large motion segments of a task. Most manipulation tasks can be broken into large and small motion segments such as the slew and grapple segments in pick and place tasks. Here, fine position control is usually required only during the small motion segments where the manipulator has to be precisely positioned to grapple or place an object. During the large motion segments, position control requirements are relaxed and other performance objectives such as smoothness (to avoid disturbances) and minimum time are important.

Open loop compensation can be used to produce smooth and fast commands in these instances where exact position control is not required and a move-and-wait strategy is inefficient. This compensation requires models of the system and environment as well as knowledge of initial conditions to generate the commands.

A popular idea for delay compensation is to use predicted values of the current state in the control law instead of the delayed measurements.

Consider the plant equation (1) with no uplink delay and an additional input disturbance term f(t):

$$\dot{x}(t) = Ax(t) + Bu(t) + f(t) \tag{8}$$

Use a FSFB, constant gain controller operating on a predicted value of the state $\tilde{x}(t)$:

$$u(t) = -K\tilde{x}(t) \tag{9}$$

The state prediction $\tilde{x}(t)$ is generated by propagating ahead from a delayed state measurement x(t-T), including the effects of the control over the time interval (t-T,t):

$$\tilde{x}(t) = e^{AT}x(t - T) + \int_{t-T}^{t} e^{A(t-\tau)}Bu(\tau)d\tau \tag{10}$$

The prediction is actually the solution over the interval (t-T,t) of the differential equation in (8) without the disturbance f. The first term in (10) uses the state transition matrix to propagate the transient response over the interval and the second term propagates the forced response over that interval. For the purposes of showing the open loop behavior, this idealized predictor assumes exact plant knowledge, uncorrupted measurements and exact computation of the integral. Note that only the delayed state x(t-T) and the control history $\{u(\tau):\tau \in (t - T,t)\}$ are available to the predictor; the disturbance f(t) is unknown by definition.

Differentiate (10) to produce a differential equation for stability analysis:

$$\dot{\bar{x}}(t) = e^{AT}[Ax(t - T) + Bu(t - T) + f(t - T)] + A[\bar{x}(t) - e^{AT}x(t - T)] + Bu(t) - e^{AT}Bu(t - T) \tag{11}$$

where the plant dynamics (8) have been substituted for $\dot{x}(t - T)$ in the first term of (11), and (10) has been used to replace the integral left over after differentiation of the second term in (10).

Cancelling terms, noting that A and $e^{AT}$ commute, gives:

$$\dot{\bar{x}}(t) = A\bar{x}(t) + Bu(t) + e^{AT}f(t - T) \tag{12}$$

Substitute (9) into (8) and (12), and combine into matrix form to get:

$$\frac{d}{dt}\begin{bmatrix} \bar{x}(t) \\ x(t) \end{bmatrix} = \begin{bmatrix} A - BK & [0] \\ -BK & A \end{bmatrix}\begin{bmatrix} \bar{x}(t) \\ x(t) \end{bmatrix} + \begin{bmatrix} [0] \\ [I] \end{bmatrix} f(t) + \begin{bmatrix} e^{AT} \\ [0] \end{bmatrix} f(t - T) \tag{13}$$

Note that the composite system dynamics matrix in (13) is triangular and that the open loop plant dynamics matrix A is on the diagonal, indicating that this form of compensation is open loop. Also note that disturbances affect the plant instantly, but have a delayed affect on the predictor since the predictor can only "see" the disturbance through the delayed state measurements. This predictor is an open loop command generator implemented by driving a "reference" model with the desired closed loop behavior. This scheme works best when the disturbance f is small relative to the control term Bu (e. g., during large motions), but cannot alleviate the closed loop bandwidth constraint.

In teleoperation, open loop compensation can be implemented with a predictive display which enables the operator to generate smooth, fast commands (References 2,3). The predictive display compensates for time delays by immediately displaying the results of the operator's commands, effectively creating a local, fast closed loop on the ground that generates a smooth, fast command sequence for the remote system (Figure 6). The loop is closed around an operator and a computer model of the manipulator and hence performance is determined by the quality of the model. The use of a predictive display has been demonstrated by Sheridan (Reference 3) to be effective in manipulator slewing applications. In this application, a predictive graphical overlay was used on a delayed video image of the manipulator.

It is possible that the predictor could be continuously updated by using an observer or estimator operating on the delayed feedback. However, these updates can only correct for slow, low bandwidth estimate errors because of the bandwidth constraint for the closed loop estimation process.

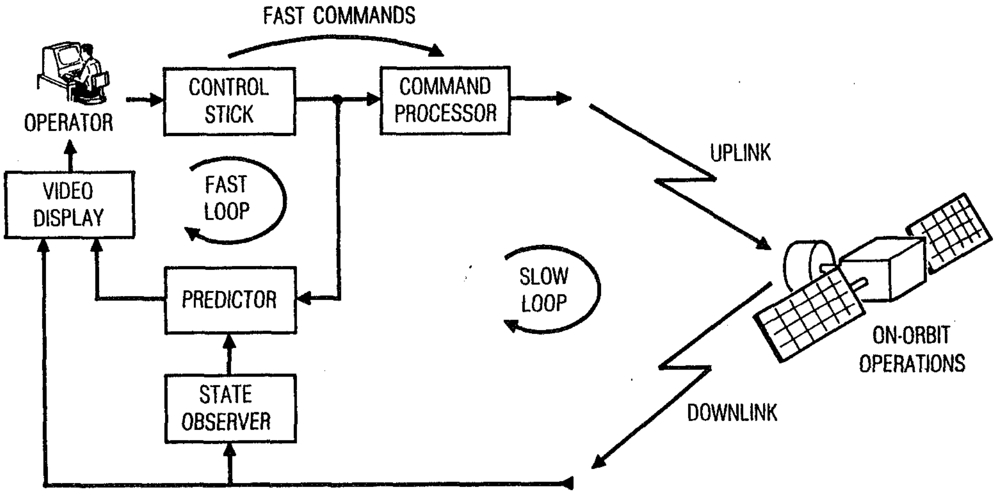


Figure 6: Predictive Display Block Diagram

## VI) SUMMARY

The objective of this work is to characterize performance constraints and discuss compensation schemes in teleoperation with time delays. The distinction between open and closed loop in defining performance constraints and compensation schemes is essential. This distinction appears to be overlooked in current research which has motivated the present effort. An important performance constraint induced by delays is an upper bound on closed loop bandwidth. This constraint is related to the causality principle of physically realizable systems and and is illustrated with an example and man-in-the-loop experiment. This constraint is pictured as a stability boundary in plots of loop delay versus closed loop bandwidth. Open loop compensation is discussed as a way of improving large motion segments of a task, including the use of predictive displays in teleoperation. Closed loop compensation is described as maximizing bandwidth within the constraint imposed by time delays. A concept for the use of a resistive hand controller in delay compensation is introduced.

Top level conclusions can be drawn from the type of performance constraints described above. First, system architects must define requirements that reflect the delay-induced performance limitations in ground-based control of spacecraft. This realization influences the partitioning of tasks between those under direct ground control and those under local autonomous control. Telerobotics and hierarchical control concepts where a local computer assumes more of the control responsibilities could provide significant benefits here. Also, an emphasis on minimizing lags in communications systems would have a direct benefit in improving closed loop capabilities. A second conclusion concerns improving performance in the presence of delays. Proper use of both closed loop and open loop compensation schemes can maximize admissible bandwidth and speed up large motions.

A significant amount of work remains in this area. A more rigorous analytical definition of the bandwidth constraint and extension to higher order and more complex systems should be addressed. The relationship to man-in-the-loop needs further work, including investigations of man's inherent response lag and implementation of open and closed loop compensation.

## ACKNOWLEDGEMENTS

This work was sponsored by W. I. Shanney of the Aerospace Corporation in support of the Air Force Satellite Control Network. The graphic display used in the experiment was developed by G. A. Irmen of The Aerospace Corporation.

## REFERENCES

(1) Mori, T., et al.; "Analysis of Time Delay Systems: Stability and Instability", Proc. of 25th Conf. on Decision and Control, Dec. 1986, pp. 895-898.

(2) Conway, L., et al.; "Tele-Autonomous Systems: Methods and Architectures for Intermingling Autonomous and Telerobotic Technology", Proc. of IEEE Conf. on Robotics and Automation, 1987.

(3) Sheridan, T. B., et al; "MIT Research in Telerobotics", Proc. of the Workshop on Space Telerobotics, July 1987, Vol. 2, pp.403-412.

## APPENDIX: SECOND ORDER SYSTEM

This example is of a positioning system driving a unit mass load with measurement delay of T seconds. The system matrices for equations (1) and (2) are:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad K = [k_p \ k_d]$$

Where $k_p$ and $k_d$ are scalar proportional and derivative gains, respectively.

Plugging these values into (5) yields:

$$\begin{bmatrix} s & -1 \\ k_p e^{-sT} & s + k_d e^{-sT} \end{bmatrix} x(s) = [0] \tag{A1}$$

The characteristic equation for this example is then:

$$e^{sT} s^2 + k_d s + k_p = 0 \tag{A2}$$

Represent the complex variable s in polar coordinates:

$$s = re^{i\theta}, \;\; r > 0, \;\; i = \sqrt{-1,} \tag{A3}$$

and use in (A2) to get:

$$e^{rTe^{i\theta}} r^2 e^{2i\theta} + k_d r e^{i\theta} + k_p = 0 \tag{A4}$$

Then use the identity:

$$e^{i\theta} = \cos(\theta) + i\sin(\theta) \tag{A5}$$

in (A4) and manipulate:

$$e^{rT(c\theta + is\theta)} r^2 (c2\theta + is2\theta) + rk_d(c\theta + is\theta) + k_p = 0 \tag{A6}$$

$$\Rightarrow \quad e^{rTc\theta}[\cos(rTs\theta) + i\sin(rTs\theta)]r^2(c2\theta + is2\theta) + rk_d(c\theta + is\theta) + k_p = 0 \tag{A7}$$

The real part of (A7) is:

$$r^2 e^{rTc\theta}[\cos(rTs\theta)c2\theta - \sin(rTs\theta)s2\theta] + rk_d c\theta + k_p = 0 \tag{A8}$$

and the imaginary part is:

$$r^2 e^{rTc\theta}[\cos(rTs\theta)s2\theta + \sin(rTs\theta)c2\theta] + rk_d s\theta = 0 \tag{A9}$$

With some trig identities, these can be reduced to:

$$r^2 e^{rTc\theta}\cos(rTs\theta + 2\theta) + rk_d c\theta + k_p = 0 \tag{A10}$$

$$r^2 e^{rTc\theta}\sin(rTs\theta + 2\theta) + rk_d s\theta = 0 \tag{A11}$$

Combining (A10) and (A11) into a matrix equation yields:

$$\begin{bmatrix} 1 & rc\theta \\ 0 & rs\theta \end{bmatrix} \begin{bmatrix} k_p \\ k_d \end{bmatrix} = -r^2 e^{rTc\theta} \begin{bmatrix} \cos(rTs\theta + 2\theta) \\ \sin(rTs\theta + 2\theta) \end{bmatrix} \tag{A12}$$

If $s\theta \neq 0$, then the coefficient matrix on the left hand side of (A12) can be inverted to solve for the controller gains:

$$\begin{bmatrix} k_p \\ k_d \end{bmatrix} = -\frac{r}{s\theta} e^{rTc\theta} \begin{bmatrix} rs\theta & -rc\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(rTs\theta + 2\theta) \\ \sin(rTs\theta + 2\theta) \end{bmatrix}, \quad s\theta \neq 0 \tag{A13}$$

Multiplied out and using a trig identity, (A13) becomes:

$$\begin{bmatrix} k_p \\ k_d \end{bmatrix} = \frac{r}{s\theta} e^{rTc\theta} \begin{bmatrix} r\sin(rTs\theta + \theta) \\ -\sin(rTs\theta + 2\theta) \end{bmatrix} \tag{A14}$$

Stability requires the roots to be in the Left Half Plane (LHP) which restricts $\theta$ to $\theta \in (\frac{\pi}{2}, \frac{3\pi}{2})$. The fact that complex roots occur in conjugate pairs permits a further restriction of $\theta$ to the upper left quadrant:

$$\theta \in (\frac{\pi}{2}, \pi) \quad \Rightarrow \quad s\theta > 0 \tag{A15}$$

where only the first harmonic (smallest magnitude root) is considered.

Positive feedback gain $k_p > 0$ is assumed and (A15) is used to develop the following inequalities from (A14):

$$k_p > 0 \quad \Rightarrow \quad \sin(rTs\theta + \theta) > 0 \tag{A16}$$

$$\Rightarrow \quad (rTs\theta + \theta) \in (0, \pi) \tag{A17}$$

$$\Rightarrow \quad \theta < (\pi - rTs\theta) \tag{A18}$$

(A18) with (A15) gives:

$$\theta \in (\frac{\pi}{2}, \pi - rTs\theta) \tag{A19}$$

Now introduce the damping parameter $\zeta$, related to $\theta$ by:

$$\cos(\theta) = -\zeta \quad \Rightarrow \quad \cos(\pi - \theta) = \zeta \tag{A20}$$

$$\sin(\theta) = \sqrt{1 - \zeta^2} \tag{A21}$$

Use (A20) and (A21) in (A18) and rearrange to get:

$$r < \frac{\cos^{-1}(\zeta)}{T\sqrt{1-\zeta^2}} \quad , \quad \zeta \in (0,1) \qquad \text{(A22)}$$

This is the bandwidth constraint for the second order example with the root magnitude r is constrained by an inverse function of delay. This is plotted in Figure 2 where the units of r have been changed from rad/sec to hz. Note that the boundary moves in as the damping parameter $\zeta$ is increased.

N90-29882

# FLIGHT TELEROBOTIC SERVICER CONTROL FROM THE ORBITER

Texas M. Ward
Don L. Harlan

Lockheed Engineering and Sciences Company
2400 NASA Rd. 1
P. O. Box 58561
Houston, Texas 77258

## ABSTRACT

This paper presents the research and work conducted at the Johnson Space Center (JSC) on the development of a testbed for a display and control panel for the Flight Telerobotic Servicer (FTS). The FTS is being developed by the Goddard Spaceflight Center for NASA. Research was conducted on both software and hardware needed to control the FTS as it has been described through the Phase B studies (ref. 1) conducted by the NASA centers. A breadboard was constructed and placed into a mockup of the aft station of the Orbiter spacecraft. This breadboard concept was then evaluated using a computer graphics representation of the Tinman FTS.

Extensive research was conducted on the software requirements and implementation. The hardware selected for the breadboard was "flight like" and in some cases fit and function evaluated. The breadboard team studied some of the concepts without pursuing in depth their impact on the Orbiter or other missions. Assumptions were made concerning payload integration because the Payload Integration Plan had not been developed to define certain constraints. Requirements ,like workstation cooperation, fail safe fail operational, redundancy, and autonomous control were kept in mind, but were not major drivers during the design.

Evaluations and assistance was obtained from the Man-Systems Telerobotics Laboratory at NASA's Johnson Space Center. Reviews were held with the NASA Goddard Spaceflight Center to make sure the designers were interpreting the FTS requirements properly. The lessons learned are presented in this paper and cover most of the problems that will be encountered during the development of a flight qualified control panel. Many of these problems are major in scope and will require extensive engineering and research to solve.

## INTRODUCTION

The Flight Telerobotics Servicer is a robotic device that is planned to be used for space station construction and satellite repair. Early studies at the Johnson Space Center were conducted to produce a general type of display and control workstation that could be used on the ground, in the space station, or in the orbiter. During the early development phase the major emphasis was on the software, to make it portable and common for all workstations. Starting in early 1988, the development team began looking at "flight like" hardware, and how to get all the functions needed for the FTS displays and controls into the aft station of the orbiter and near a window. The development team also had to look at simultaneous operations with the Remote Manipulator System (RMS) so the only readily available space was located on panels A6 and A7. If the display and control panel could fit in this area then it would also fit in locations L10 and L11 which are on the port side of the Orbiter flight deck. There is not room at A6 and A7 for the processor so the team was looking at the possibility of putting the processor under L10 through L12. Sizing and cooling turn out to be big issues on the hardware. The Orbiter Interface Control Document (ref. 1) was consulted on what services are available and this was matched against the requirements set forth in the FTS Phase C/D Request For Proposal (ref. 2) for the FTS written by NASA. Little time was spent on the Orbiter mission kits that would be required as this would be a waste of time until the location of the panel was defined by the Orbiter program office.

Designing a panel that would give the operator maximum visibility into operations external to the Orbiter without using multiple monitors led us into researching flat panel technology, split screen imaging and sharing the orbiter video system. There was a conflict with the Remote Manipulator System operations when an attempt was made to share the Closed Circuit Television monitors on the aft deck. The physics of connecting any FTS cameras into the Orbiter Closed Circuit Television system became a research project that was not pursued. The space available behind panels A6 and A7 did not lend itself to using a very large CRT. A 9 inch color monitor and two 5 inch monitors were selected for the final testbed configuration. This created a heat problem. With each color CRT requiring about 150 watts of cooling there is a need to circulate air behind the panels.

Tentative plans were made to mount the processor(s) behind panels L13 or L14. There is sufficient space back there and cooling is convenient on the Orbiter. No significant time was spent looking at the possible attach points or cable routing while the Standard Switch Panel of other control panels were mounted in panels L10 and L11 above. When the size of the processor(s) is determined then this problem needs to be studied in more detail. Some thought was given to mounting the control panel processor(s) out in the payload bay. This is workable but would use more Orbiter payload wiring.

According to the Space Shuttle Systems Payload Accommodations, JSC 07700 volume XIV, (ref. 1) there are connectors behind panels A6-A2 and A7-A2 that we could use to route signals to the Payload Station Distribution Panel (PSDP). From there the developer could tie into the processor and route signals to the payload bay and the FTS hardware called the High Level Control Module (HLCM). All the wires would be covered under the definition of a "Mission Kit" and not require Orbiter modifications. So this part of the study although only a paper research looks as though it is workable for the flight item.

## ERGONOMICS

The workstation breadboard was designed to allow for efficient human/machine interactions. Several layouts of workstation display and Control devices were studied with Human Factors engineers to insure the incorporation good human engineering practices in the design of the breadboard. These included an Alpha workstation, a portable workstation and two versions of an Orbiter panels A6-A2 and A7-A2 workstation.

## THE JSC DISPLAY AND CONTROL LABORATORY

The laboratory used to develop the hardware and software covered in this research paper is located in Building 16 at the Johnson Space Center. The laboratory is run by the Teleoperator Systems Branch of the Systems Development and Simulation Division. The laboratory is used to develop Orbiter and space station display and control breadboards. It is also used for robotic studies. The laboratory has a variety of computers, mock-ups, and robot arms.

## ALPHA PANEL

Early in the program the JSC team developed a panel called the Alpha panel. This panel includes an electroluminescent display with touch screen. Also on the panel is an assortment of lights, switches and programmable display pushbuttons. All this is tied into two IBM PC-AT computers through off the shelf interface boards. Software developed for this Alpha panel is used on the final Orbiter panel configuration. A data base software was developed first then the other modules were added as needed to interface with the hardware and a robotic simulation of the FTS. A software package was purchased that would run on a simulated interrupt schedule because the PCs do not support multitasking capability. The two PC computers were tied together using Ethernet TCP/IP hardware and software. The simulation computer, which is a Silicon Graphics IRIS 4D computer, was also tied into this Ethernet network. Simulations were generated on the IRIS to support each of the data and command functions, including a visual representation of the FTS on the end of the RMS in the Orbiter payload bay. Any function that would be in the HLCM was simulated in the IRIS.

A considerable amount of time was spent trying to procure a suitable data base software package to use in the PCs. Most vendor packages either had too much or too little capability, so it was decided to develop a custom data base in-house. The data base was easy to define as the programmers had previous experience in simulation data bases. The biggest task was to build support software for that data base. The software was done in C computer language.

## PORTABLE WORKSTATION

The Man-Systems Department at JSC developed a portable workstation concept that could be stowed in the lower flight deck. This workstation contained one 17 inch CRT that would use windowing to give the operator multiple views. This concept was not evaluated by this team in a breadboard configuration. Some research was done into how such a control panel would be interfaced to the Orbiter. An interface panel could be built and mounted in the area of the Standard Switch Panels, (L10 thru L14), on the port aft side of the flight deck. This would allow the portable control panel to be connected into the PSDP for power and signal distribution. Cooling would be performed by dumping the heat load into the cockpit using a small fan.

## ORBITER WORKSTATION

### DISPLAYS, GENERAL

Due to the workstation's primary location on the aft flight deck, the operator can view FTS operations from the aft windows only on a periodic basis. Therefore, the breadboard design provides a primary display, two secondary displays, and access to the Remote Manipulator System displays. The JSC Orbiter workstation breadboard is mounted in an aft station mockup which includes two simulated Orbiter closed circuit television monitors. This allows the operator to control the FTS without the need of continuous direct viewing of the robot.

At this time the FTS simulation provides four camera views which include views from the head of the robot, a view from both the left and right wrists, and an auxiliary view. Three of the four views can be displayed at a time on any of the three monitors. This allows the operator to select the primary or secondary views according to the needs of a particular task.

### PRIMARY DISPLAY

The primary display on the breadboard will be a 9 inch diagonal color monitor located on panel A6-A2. The 9 inch monitor is connected to a video digitizing board on one of the PC computers. This monitor provides a display of both text and graphics. The information displayed on this monitor include:

a. FTS health and status conditions.
b. Joint rates and positions.
c. Safety messages and system diagnostics.
d. Graphics generated review of task sequences.
e. Camera video scenes.

### SECONDARY DISPLAYS

The secondary displays on the Orbiter workstation include two 5 inch monitors which display camera views only. They are connected to a video distribution unit that has inputs from both the remote cameras and the IRIS computer scene generation output.

### FLAT PANEL DISPLAYS

Due to the limited depth of panels A6-A2 and A7-A2 in the Orbiter, the installation of CRTs at this location is not possible. Therefor, the monitors used in the breadboard design are of functional use only. However, flat panel technology is rapidly developing and the use of color liquid crystal displays and or color plasma displays for space applications is in the foreseeable future. These type of displays do not require as much mounting depth as conventional cathode ray tube monitors, making them better display devices for the FTS orbiter workstation at this location. The display and control laboratory has a 6.25 X 6.25 inch liquid crystal color display. Present plans are to install it in place of the 9 inch monitor for demonstrations and evaluation. The laboratory also has some small color plasma displays, however these are not of sufficient quality to use on our breadboards.

## PROGRAMMABLE SWITCHES

The workstation employs a 4 X 5 matrix of programmable switches developed by Litton Systems. Each individual switch consists of a 16 by 35 array of miniature light emitting diodes, which can be pre-programmed to display text or graphic messages. Since the switches are programmable, a single switch can control several devices or functions. This effectively reduces the amount of panel space required to control the FTS. The orbiter workstation's programmable switches control such functions as:

a. Camera control, pan, tilt, and zoom.
b. FTS operation mode selections.
c. FTS position control system.
d. FTS work site lighting.

As development of the FTS continues, it may be necessary to include additional functions which are not presently included in breadboard design. Programmable switches will allow these additional functions to be implemented without redesigning the workstation.

## HAND CONTROLLERS

At first two 6 Degree of Freedom hand controllers were installed on the Alpha control panel to complete the hand controller command loop. These controllers were force torque controllers. They were analog devices so they were connected to one of the PCs through an analog to digital conversion board. The commands were transferred to the IRIS computer via the laboratory Ethernet. These controllers turned out to be difficult to use because of the lack of feedback. Fingertip control switches were added to the hand controllers to activate all the necessary functions for gripper or end effector control. These switches were easy to activate, but put torque on the hand controllers causing them to issue unwanted commands.

The second set of hand controllers to be installed were 6 degree of freedom mini-master controllers. These controllers have a RS-232 interface. They were used as a stepping stone to become familiar with the new interface and the mini-master concept while waiting for the 7 degree of freedom controllers to be built.

The Orbiter panel was upgraded to 7 degree of freedom mini-master controllers developed by Schilling, Inc. These control units are a miniature replica of the Tinman manipulator arm with approximately the same range of motion as the robotic arm itself. Since the velocity and angular motion initiated at the master unit is duplicated by the slave arm, the operator is able to efficiently maneuver the robot arms. The Schilling master control unit has three separate discrete switches located on the end effectors. These switches provide the capability to open or close the gripper, and to freeze the slave arm in its current position. Once the freeze function is activated, any subsequent movement of the control unit does not affect the arms. This allows the operator to easily freeze the slave arms in order to do another task such as camera or lighting control. The operator will, however, have to take his hands off the controllers to operate other switches in the present workstation configuration.

The hand controllers are mounted in a separate panel that is attached to the Orbiter panel structure between the A6-A2 and A7-A2 panels. This configuration is not expected to be a final configuration because it is not a goal of the development team to develop hand controller configurations for actual flight.

## BACKUP SWITCHES

The functions for the switches that were selected for the backup control were taken from the RMS displays and controls. Care was taken to minimize the number of switches to conserve space on the panel. When the FTS is operating using the umbilical the number of switches and switch functions will be a driving factor on the number of wires in the umbilical. There is a problem where the FTS was disconnected from its umbilical and was controlled from a communications link. The backup switches will only be hard wired into the front end of the communications network which have not been defined. The end point connections of these wires would be determined by the design of the robot. In the Orbiter testbed, the backup switches simply set a discrete or flag in the computer that is read, stored, displayed and integrated with the command data block that is sent to the simulated robot. If the switch needs to provide direct drive power to a joint motor or gripper motor in the robot, then it will take about 34 wires that are capable of carrying the required current load in the present Orbiter breadboard configuration.

## CAUTION AND WARNING INDICATORS

Caution and warning indicators were installed on a common area of the workstation. These indicators are sunlight readable and have "dead legends" until activated. At this time caution and warning messages to be displayed are being researched as evaluations of the display and control of the FTS continues.

An audible alarm will also be implemented in the workstation. This alarm will sound if a condition arises which requires the operator's immediate attention.

## PROBLEM AREAS

One of the big problems that the FTS project in general has run up against is the establishment of a RF communications link between the control panel and the robot when it is not on an umbilical. Because the plans include up to 4 video links as well as a high data rate for force reflective hand controllers the requirements exceed those planned by the program. If there is a requirement for the robot to be controlled by the space station workstation and have the Orbiter workstation operating in cooperative mode one workstation transmitter would have to be disabled. This also requires the cooperative workstation to now interpret RF commands from the space station workstation and display them, act on them, or at least store them in its data base. The software in the orbiter workstation would have to make an intelligent decision based upon the type of command the space station workstation has issued.

During a design review of the panel it was pointed out that one of the mission rules, 12-5 Orbiter Avoidance Maneuvers Constraint, stated that a pilot shall always be available to perform a collision avoidance maneuver during loaded RMS operations. If that pilot had to be located in the aft station to operate the Digital Auto Pilot (DAP) while the FTS is on the RMS, then there is a "people" space problem. If the pilot can be located in the pilot seat or be the RMS operator then locating the FTS panel in A6 and A7 will not pose a violation of the mission rule. Logically of course the pilot needs an aft station out the window view to do any collision avoidance maneuver with a payload in or near the payload bay. During the development of the Orbiter panel it was concluded that the emergency stop capability of the FTS would allow the FTS operator to hit the button and take over the DAP controls for an evasive maneuver if required. Further investigation of this problem will have to be conducted with the Mission Operations Directorate at JSC before the FTS panel can be mounted in A6-A2.

## FOLLOW ON WORK AND APPLICATIONS

### TASK EVALUATIONS

The latest version of the Orbiter FTS workstation will be used for task evaluations. There are plans to install the Orbiter panel in a high grade mockup so that it can be used by other groups for studies. Collision detection with the FTS mounted on the end of the RMS is a study that can be done with this system by adding the collision detection to the IRIS computer.

### PROGRAMMABLE DISPLAY PUSHBUTTON DEVELOPMENT

As new requirements for functions to install into the Programmable Display Pushbuttons are defined they can easily be integrated into the Orbiter panel.

### RELOCATION

The latest configuration of the Orbiter FTS panels are installed on rails so that they can be removed from the mockup easily and relocated into other locations.

## VOICE RECOGNITION

The software was designed so that voice recognition command system could be added to the design if the need arises or financing is provided.

## COOPERATIVE WORKSTATION

The team that has developed the workstation is now in the process of developing a FTS workstation for the space station. Once this workstation is developed some studies can be initiated on workstation cooperation.

## ACKNOWLEDGEMENT

Support for this work was provided by the National Aeronautics and Space Administration through contract NAS9-17900 to Lockheed Engineering and Management Services Company.

## REFERENCES

1 Final Report Flight Telerobotic Servicer (FTS) Tinman Concept In-house Phase B Study Vol I & II, Goddard Spaceflight Center, FS-GSF-0042, September 9, 1988.

2 Flight Telerobotic Servicer Phase C/D Request For Proposal, Goddard Spaceflight Center, November 11, 1988.

3 NASA, Space Shuttle Systems Payload Accommodations, NSTS 07700 Volume XIV, Revision 1, Johnson Space Center, Houston, 1986.

# Teleoperation Experiments with a Utah/MIT Hand and a VPL DataGlove

D. Clark, J. Demmel, J. Hong
G. Lafferriere, L. Salkind, X. Tan

Robotics Research Laboratory
Courant Institute
New York University
New York, NY 10012

**Abstract**

We describe a teleoperation system capable of controlling a Utah/MIT Dextrous Hand using a VPL DataGlove as a master. Additionally the system is capable of running the dextrous hand in robotic (autonomous) mode as new programs are developed. We present the software and hardware architecture used and describe the experiments performed. We further analyze the communication and calibration issues involved and investigate applications to the analysis and development of automated dextrous manipulations.

## 1 Introduction

The research in Dextrous robotic hands has been fueled by two main factors: the interest in more versatile manipulators and the availability of truly dextrous end effectors like the Utah/MIT hand [5] and the Salisbury hand [7]. Research has ranged from low level control [13] to high level task planning or grasping issues [8,12]. However, because of the large number of degrees of freedom to be controlled implementations of complex manipulations tasks have been few. The great similarity between the human hand and the Utah/MIT hand suggested that a master-slave pairing should be possible if we could measure the human joint angles accurately and in real time. The VPL DataGlove provides just that capability.

At the Robotics Research Laboratory of New York University we have set up a hardware and software architecture suitable for teleoperating the Utah/MIT Hand with a VPL DataGlove. The Utah/MIT Hand is attached to a PUMA 560, whose position is also teleoperated. Thus, as the human wearer of the DataGlove moves his hand and fingers, the Utah/MIT Hand and PUMA mimic him closely. This work includes a sophisticated calibration procedure for the DataGlove and a supervisory real time operating system which made communication between the different system components possible. The current implementation allowed us to perform the following tasks:

- stacking blocks
- picking up a milk carton and pouring milk into a cup
- picking up the cup using the handle and pouring the milk into another cup
- picking up a nut and screwing it into a bolt.

Moreover the system also allows the operation of the robotic hand in an autonomous mode providing for a smooth transition between the different modes. This lets the operator transfer control to preprogrammed motions when appropriate. The system will be described in the following sections.

## 2 Other related work

Our system is purely positioned controlled and uses only the operator's visual feedback. We have not analyzed other feedback needed by the operator to facilitate the manipulation task nor have we developed a force reflecting master [1,6,11,14]. Our work is, however, unique in providing the operator with a nearly anthropomorphic end effector and a very natural interface.

The main emphasis of our work is in dextrous manipulation. Most research in that field is still in the exploratory stages, including building computer systems suitable for real time control [4]. Much of our effort has gone into this as well, as a result of which our system is quite flexible and fast; it will be compared to other systems in detail elsewhere [2,10]. As for accomplishing actual manipulation tasks using the Utah/MIT Hand, to our knowledge[4] our work is among the most successful.

## 3 Hardware

The hardware components of the system consist of: a VPL DataGlove, a Utah/MIT Hand, a PUMA 560, a SUN workstation and several single board 68020 based computers. The communications between the different parts of the system were either through serial lines or through Ethernet (see Figure 1).

The VPL DataGlove (or simply Glove) consists of a cloth glove with fiber-optic cables attached to the back and palm and passing over the finger joints (see Figure 2). The fiber-optic cables in the Glove are treated at the sites where fingers flex so that light escapes when a finger is bent. More light is lost when the movement is greater. Phototransistors convert the light into an electrical signal which is then digitized. Coupled with a Polhemus sensor (which can be mounted on the back of the Glove), the position and orientation of the hand in space (6 degrees of freedom from the Polhemus sensor) as well as 14 finger joint angles (from the DataGlove) can be measured in real time (about 60 times a second).

The Utah/MIT hand (henceforth simply Hand) is a 16 degree of freedom manipulator consisting of 4 anthropomorphic fingers and a palm (see Figure 2). The fingers are arranged so that one of them opposes the other three much as in the human hand. The "thumb" however, has exactly the same design as the other fingers and it must always oppose them, not allowing for a flat palm configuration. Each finger has four joints and each joint is operated by a pair of opposing tendons connected to pneumatic pistons. It has position sensors at each joint and tension sensors for each tendon, mounted on the wrist. This sensory information is used by the controller to servo the joint position and/or joint torque. The actuator package is separated from the hand by an articulated extension linkage (the remotizer) which allows the hand to be conveniently attached to a robot arm. Control signals for the pneumatic pistons come from an analog controller box which is connected to a VME bus via D/A and A/D boards. Several 68020 based Ironics boards on the bus allow implementation of a variety of digital control schemes. A SUN workstation, connected to the VME bus via an HVE bus to bus connector, is used for program development and compilation. Programs are then run on the SUN or (more commonly) downloaded for running on the Ironics boards. In our current setup both the program that maps the Glove angles to Hand angles and a monitoring

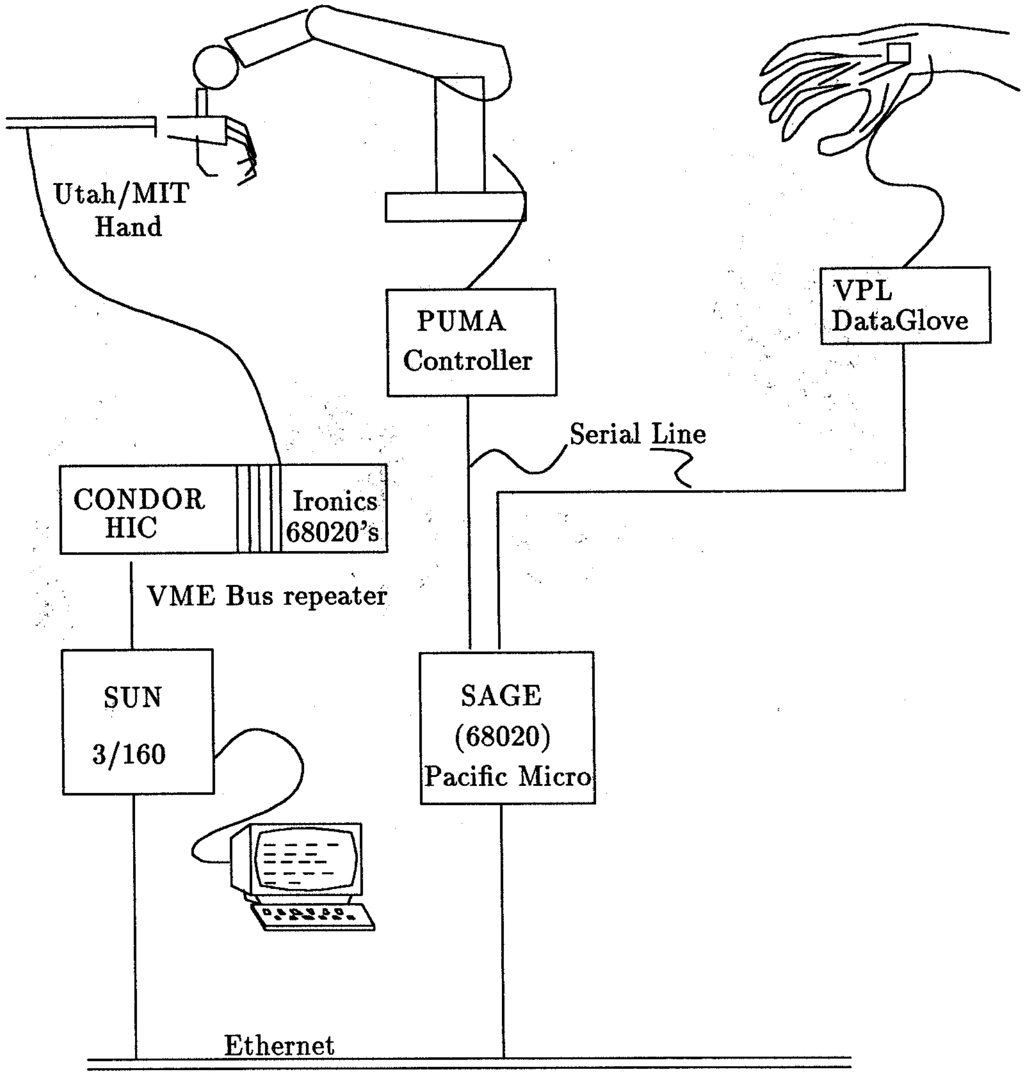


Figure 1: Architecture of Hand Teleoperation System

program run on the SUN while a joint level servo loop and a mode-switching program run on the Ironics.

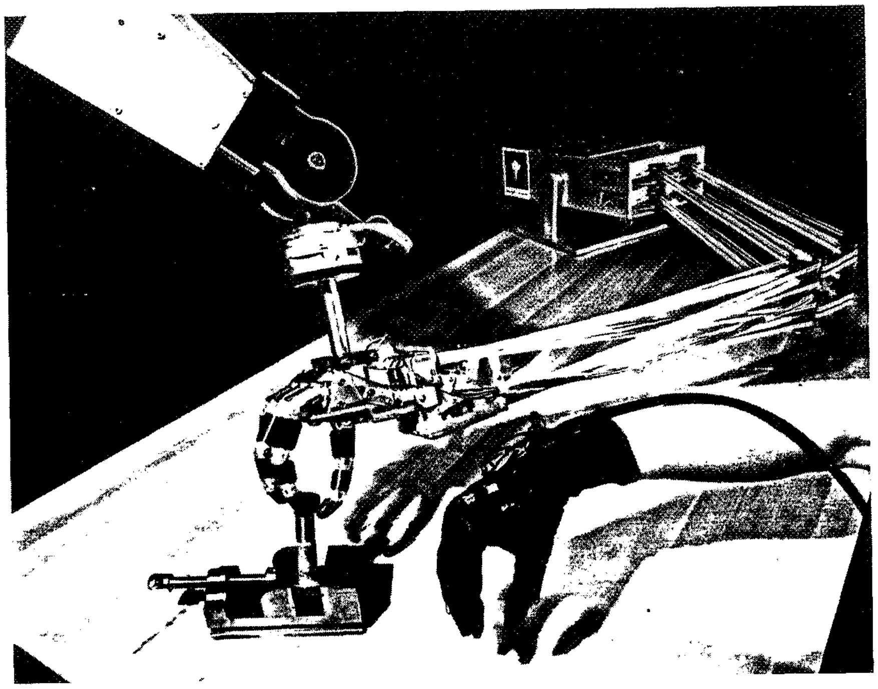


Figure 2: The VPL DataGlove and the Utah/MIT Dextrous Hand attached to a PUMA 560

The Hand is mounted on the end of a PUMA 560 robot arm. To control the position and orientation of the Hand in real-time, a 68020 generates new robot setpoints every 28 msecs based on the current Polhemus sensor readings. These setpoints are then sent over a serial line to the PUMA VAL-II controller using the ALTER interface, which performs independent joint PID control at a 1 msec servo loop rate.

## 4 Software

The software components of the system include SAGE, a real time operating system; Condor, a runtime environment for controlling the Hand; and HIC, a servo loop scheduler for hierarchical controllers.

A major component of the remote teleoperation system is the SAGE operating system, which interfaces the Glove, arm, and Hand. SAGE is designed specifically for real-time robotics supervisory control, and has multi-tasking, memory management, low-overhead synchronization, and network communications capabilities.

Both HIC and Condor run on four 68020 boards mounted on a VME bus. Condor [9] provides the development system, the inter-processor communication, the user interface, the timer interface, and the debugging support for control of the Utah/MIT hand. HIC provides the process or event scheduler and the Periodic Data Queues used for inter-process communication. HIC's scheduler performs the same functions as the Minimal Operating System (MOS) in Condor (and in a similar manner) but has more flexibility.

An essential part of this teleoperation capability was a very complex calibration procedure developed for the DataGlove which mapped the sensor data to Utah/MIT hand joint angles in a way that looked natural to the wearer of the Glove.

### 4.1 Raw joint servo.

To control the Hand from the Glove only the lowest level of control is required of the Hand's controller. This is called the *raw joint servo* since it works entirely in the Hand's native joint position and tendon strain units. Figures 3,4 show the raw joint servo schematically and the communication between the servo and the controlling Sun. The box at the bottom of the figure represents the joint servo. This is a HIC process running on one of the Ironics processors. The three phases of the servo, input — planning — output, are represented by the top four sub-boxes. Notice that there are in fact two separate planners for the two different modes of operation (autonomous and teleoperated). The mode switching done by the switch event (box on center right) is achieved by changing the pointer for the current planner. The switch event is now triggered by a SUN

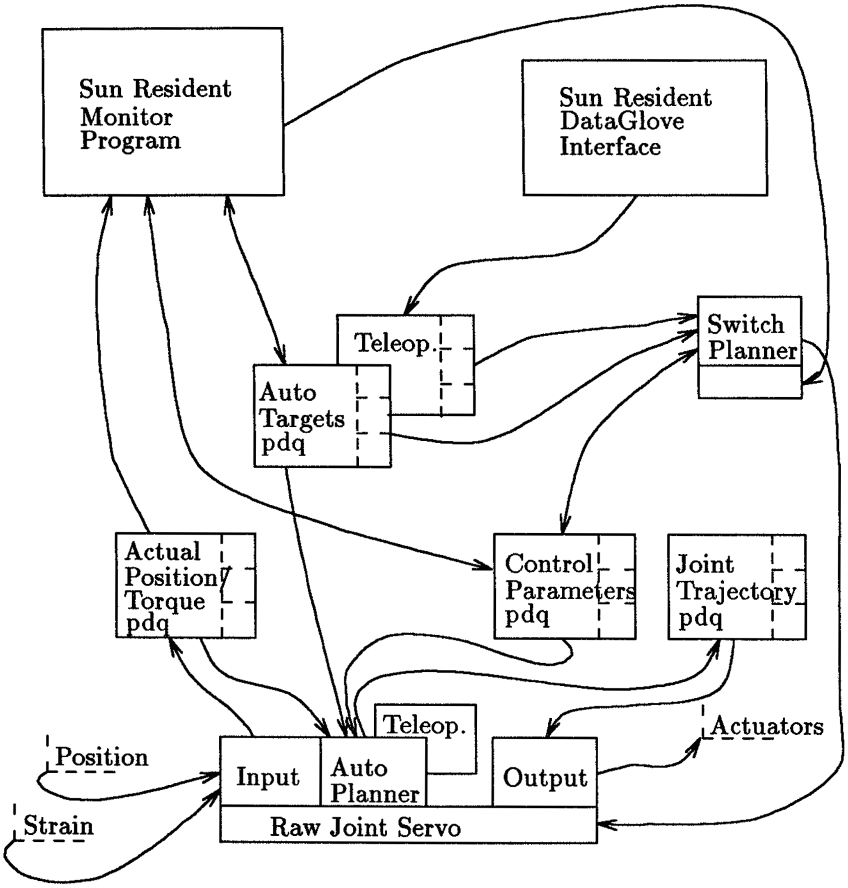


Figure 3: Communication with the Joint Servo (teleoperated mode).

resident monitoring program (but could in fact be triggered from anywhere in the system). Some position offsets are also computed by the switch event in order to make the transition smooth (and are placed in the control parameters buffer).

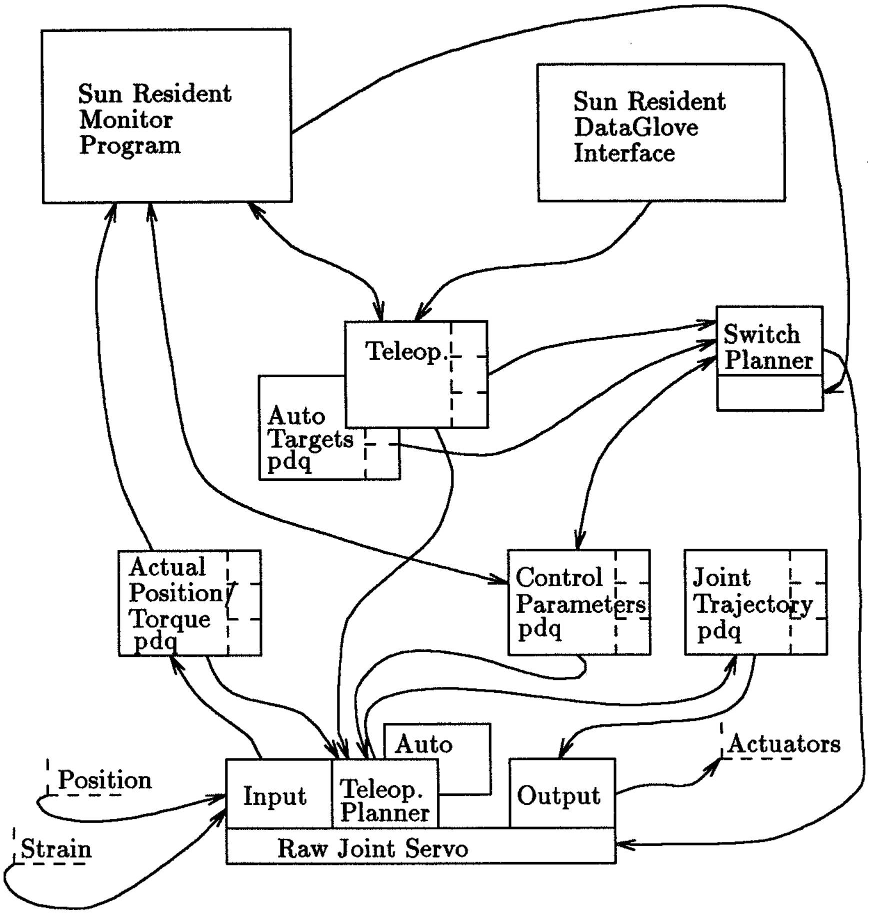


Figure 4: Communication with the Joint Servo (autonomous mode).

Above the servo are the five periodic data queues (pdq's) used by the servo (however only four of them are in use at a given time since the two planners use different target pdq's). The *input* procedure obtains values from the joint position and tendon strain sensors and produces position and torque values which are placed in the *actual position/torque* pdq. The second phase, the *planner*, takes these actual values and the target values and control parameters from the *target position/torque* pdq and the *control parameter* pdq respectively. The planner uses a combination of force and position control to produce trajectories for the joint actuators. The planner places the trajectories in the *trajectory* pdq. The *output* routine picks up the trajectories and applies them to the Hand's actuators. This loop is repeated 250 times per second.

The top two boxes in the figure represent programs running on the Sun3/160 host computer. The monitor program watches the actual, target, and control parameter pdq's and displays their values. To control the Hand with the Glove a second program on the Sun is run which supplies target joint positions based on the sensed position of the operator's joints in the glove.

As higher level servo loops are implemented, controlling, say, the fingertip positions, they will generate the joint level targets in autonomous mode. While no such servo is running, the joint level servo uses the last targets specified.

The pdq's are the unifying data structures in HIC. Their use provides a disciplined way for numerous multirate hierarchical servo loops to communicate in a timely way.

## 5 Calibration and workspace constraints

The calibration problems involved in interfacing the Glove with the Hand are several. First, the kinematic structures of the Hand and the human hand are different. Second, the Glove can only measure some of the joint angles of the human hand. Third, the Glove sensors do not measure individual joints separately but rather there are strong nonlinear correlations among the sensors. The most complex part of the calibration procedure turned out to be the elimination of these correlations.

The calibration procedure is described in detail in [3]; we outline it here. The calibration procedure is decomposed into three levels. In the first level the raw sensor values are converted into joint angles by a parameter identification procedure. After considerable experimentation it was found that the relationship between raw output from the Glove sensors and the joint angles could be modeled satisfactorily by using a function of the form

$$\alpha = ar + b + c\log(r)$$

where $\alpha$ is the joint angle, $r$ is the sensor reading and $a$, $b$, and $c$ are parameters to be identified.

In the second level the correlations between the different calibrated angles were eliminated. This was done by first identifying those joints which were correlated and then eliminating the influence of one of the angles on another one by one. It is important to note that these correlations are not linear.

Finally, in the third level, we used forward kinematics on the human hand and inverse kinematics on the Hand to map the Glove angles to suitable Hand angles which would put the Hand fingertips in the same relative positions as the Glove fingertips.

Because of limitations in the workspace of the remotizer constraints had to be put on the motion of the joints of the PUMA. Finding the appropriate set of constraints required solving the inverse kinematics for the remotizer. Those were approximated using a simple open kinematic chain model. The precise kinematics are much harder to describe because of flexibility of the rods and the complex arrangement of joint and links.

## 6 Experiments

With the setup described above we were able to successfully teleoperate the Hand using the Glove to supply finger angles, and the Polhemus sensor to supply position and orientation for the PUMA 560 carrying the Hand. The following experiments were performed.

1. Stacking blocks. Three two-inch cubic blocks were picked up using two or three fingers and then stacked up. The whole task took a trained operator 1 minute and 45 seconds to perform.
2. Picking up a milk carton and pouring milk into a cup. A regular half gallon milk carton was opened by pushing back the flaps of the (already unglued) spout. Then it was picked up by grasping it from the side. After tilting it enough the milk poured into the cup without spilling out. The task took 2 minutes and 20 seconds to perform, including time to move to carton to a position where it could be grasped more firmly.

3. Picking up the cup using the handle and pouring milk into another cup. Three fingers were moved through the handle while the thumb pushed against the side of the cup. (1 minute and 50 seconds)

4. Picking up a nut and screwing it onto a bolt (see Figure 2). A nut is picked up by the thumb and two fingers. It is then started on the 3/4in bolt and screwed in using two fingers. When successful it took a trained operator under $1\frac{1}{2}$ minutes to perform. However starting the nut on the thread was difficult and 3 out of four times either the nut was dropped or the grip became unstable and the experiment had to be restarted.

These experiments were performed fully in teleoperated mode and only visual feedback was used. Preliminary results using both the autonomous mode and a transition to a purely force controlled mode for releasing a grip indicate that we may greatly improve the performance in experiments such as 1 and 3 above. A force controlled mode can be achieved by setting the position feedback gain (specified in the control parameters data structure) to zero using the monitoring program.

## 7 Applications

The most immediate application we foresee for our current teleoperation setup is as an aid in developing automatic control schemes for dextrous manipulation tasks. Teleoperation can be used to perform the parts of a complex task which have not yet been automated. For example, in stacking up blocks we may teleoperate the hand until a grasp is achieved and then let a preprogrammed routine maintain it while we perform other motions with the whole hand. Then, when more fine positioning needs to be done to properly align the blocks control can again be transferred to the operator. Later when a grasping routine is written the initial teleoperation may be reduced to a simple pregrasping position at which point the autonomous program takes over and the rest proceeds as before. In this way smaller parts of a more complicated task can be tested before the whole task is automated. Our current setup also easily allows a mixed teleoperation/automated mode. An example could be found in writing with the Hand. The compliance of the pen on the paper could be preprogrammed while the shaping of the characters would be teleoperated.

## 8 Conclusions

We described a teleoperation system for the Utah/MIT Dextrous hand using a VPL DataGlove. While the system could be greatly improved in many areas it provides a new facility with which to perform fine manipulation experiments. The ability to switch back and forth between a teleoperated mode and an automated mode both allows the operator to rest while some already preprogrammed motions are performed and helps in the development of complex dextrous manipulations. The teleoperated mode is facilitated by the use of the natural interface provided by the DataGlove.

## Acknowledgments

Work on this paper has been possible by Office of Naval Research Contract #N00014-87-K-0129, National Science Foundation CER Grant #DCR-8320085, Carnegie Mellon University subcontract #CMU-406349-55586 (NSF), NASA grant #NAG 2-493 and by grants from the Digital Equipment Corporation and the IBM Corporation. The second author is a Presidential Young Investigator.

# References

[1] Antal K. Bejczy and Zoltan F. Szakaly. *A laboratory system for generalized bilateral and shared manual-computer control of robot arms.* Publication 88-5, Jet Propulsion Laboratory, April 1988. Preliminary draft, presented at the Workshop on Dextrous Hands, IEEE, Philadelphia, April 1988.

[2] Dayton Clark. *Hierarchical Control System.* Robotics Report 167, New York University, Courant Institute, Robotics Research Laboratory, 1988.

[3] Jiaiwei Hong and Xiaonan Tan. *Teleoperating the Utah/MIT Hand with a VPL DataGlove. I.DataGlove Calibration.* Robotics Report 168, New York University, Courant Institute, Robotics Research Laboratory, New York, 1988.

[4] T. Iberall and S.T. Venkataraman, editors. *Workshop on Dextrous Robot Hands*, IEEE, Philadelphia, April 1988.

[5] Steve C. Jacobsen, John E. Wood, D.F. Knutti, and Klaus B. Biggers. The Utah/MIT dextrous hand: work in progress. In M. Brady and R.P. Paul, editors, *First International Conference on Robotics Research*, pages 601–653, MIT Press, Cambridge, Massachusetts, 1984.

[6] S. Lee, G. Bekey, and A.K. Bejczy. Computer control of space-borne teleoperators with sensory feedback. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 205–214, St. Louis, MO, March 1985.

[7] Matthew T. Mason and J. Kenneth Salisbury, Jr. *Robots Hands and the Mechanics of Manipulation.* MIT Press, Cambridge, Massachusetts, 1985.

[8] B. Mishra, J.T. Schwartz, and M. Sharir. *On the Existence and Synthesis of Multifinger Positive Grips.* Robotics Report 89, Courant Institute, NYU, New York, November 1986.

[9] Sundar Narasimhan, David M. Siegel, David Taylor, and Steven M. Drucker. *The Condor Programmer's Manual – Version II.* Artificial Intelligence Laboratory, MIT, Cambridge, Massachusetts, 1987.

[10] Lou Salkind. *The SAGE Operating System.* Robotics Report, New York University, Courant Institute, Robotics Research Laboratory, 1988.

[11] Lawrence W. Stark, Won S. Kim, and Frank Tendick. Cooperative control in telerobotics. In *Proc. IEEE Robotics and Automation Conference*, pages 593–597, Philadelphia, PA, April 1988.

[12] Rajko Tomovic, George E. Bekey, and Walter J. Karplus. A strategy for grasp synthesis with a multifingered hand. In *Proceedings of the International Conference on Robotics and Automation*, pages 83–89, IEEE, Raleigh, North Carolina, April 1987.

[13] S.T. Venkataraman and Theodore E. Djaferis. Multivariable feedback control of the JPL/Stanford hand. In *Proceedings of the International Conference on Robotics and Automation*, pages 77–82, 1987.

[14] Jean Vertut and Philippe Coiffet. *Teleoperation and Robotics: Applications and Technology.* Volume 3B of *Robot Technology*, Prentice Hall, 1986.

# INSTRUCTION DIALOGUES: TEACHING NEW SKILLS TO A ROBOT

C. Crangle and P. Suppes
IMSSS, Ventura Hall, Stanford University, Stanford, CA 94305.
COLLEEN@SUWATSON.BITNET

## 1. Introduction

This paper describes extended dialogues between a human user and a robot system. The purpose of each dialogue is to teach the robot a new skill or to improve the performance of a skill it already has. Our particular interest is in natural-language dialogues but the techniques we illustrate in this paper can be applied to any high-level language. The primary purpose of this paper is to show how verbal instruction can be integrated with the robot's autonomous learning of a skill.

The learning techniques we apply are based on a set of concepts developed within mathematical learning theory and thoroughly tested in human learning ([8] - [11]). These techniques relate directly to skill-performance tasks in which the subject (human or robot) learns to make responses along a continuum of values. A great many tasks we would want a robot to perform fall into this category. Whenever the robot is required to position its end-effector, for instance, some specific point on a line, or on a surface, or in 3-dimensional space, must be selected.

It is a feature of most high-level task specifications that they underspecify in that they fail to express many of the details the robot requires to carry out the task. Nowhere is this more evident than in natural-language descriptions. A request as simple as *Put the wrench away!* says nothing about where the wrench is to go or where on the target surface it is to be placed. A command such as *Place the wrench parallel with the front edge of the shelf, 4 inches from the left end and 2 inches from the front edge!* will leave little doubt about where to place the wrench. But such detailed and specific descriptions are tedious and, in fact, unnecessarily detailed. The speaker will typically have in mind not some one point but an area of the shelf — the left side,.somewhere in the middle, towards the far right, for instance — and will sometimes be quite neutral as to the object's orientation. Some way must be found for the specific intention of the speaker to be communicated naturally and for the robot to respond appropriately. We aim to show how verbal instruction accompanied by autonomous learning provides a way.

In the instruction dialogues we have in mind the operator uses high-level commands to request some action from the robot — *Pick up the receiver!* or *Pick up the fork in the middle!*, for instance. When the robot responds to such a request, the operator makes free use of qualitative commands to correct or confirm the action taken — *Not so far up!*, *That's fine!*, or *Be more careful!*, for instance. These qualitative corrections are expected not only to alter the robot's current behavior, but also to influence its behavior in the future whenever the original request is repeated. In other words, the robot is expected to learn from its interaction with the operator.

It is useful to ask in what circumstances robot instruction is most appropriate. Whenever the robot's basic repertoire of skills must be expanded over time to meet the demands of a changing task load or a changing environment, instruction has an important role to play. If, in addition, it is the operator who must adjust the robot's functioning and the operator is not a robotic specialist — in space applications, for instance, the operator will typically be a specialist in his or her own field — instruction has an essential role to play. It is imperative in these circumstances that the operator be able to request action from the robot and adjust the robot's subsequent behavior in as natural a way as possible.

Skills or tasks to be taught are categorized by the number of response variables involved. For many robot tasks, each response variable will have the dimension of space or the dimension of time and we therefore talk of tasks being 1-dimensional or 2-dimensional, and so on. A typical 1-dimensional task is that of learning to select an interval on a line. This learning problem would arise, for instance, with a request to put one object on another much larger object — a box on a table, for instance — if the robot did not know the desired position for the box along the length of the table. The task becomes 2-dimensional if the setback of the box from the front of the table were also to be learned. A 2-dimensional learning task arises whenever the robot is directed to go somewhere (or move its arm someplace) in order to perform a specific action. For instance, if the robot is to go to a refrigerator unit or a storage cabinet to fetch something, it must stop in front of the refrigerator or cabinet at a point where it will not impede the door's opening. The set of points that are near enough to the refrigerator or cabinet but not in the way constitutes the region the robot must learn. Another 2-dimensional task is the seemingly straightforward one of setting the table for dinner. This activity entails a large number of learned skills: what the orientation of each knife, fork, and spoon should be relative to the plate, how far right of the plate the dinner knife should go, how far from the edge of the table the dinner plate and side plate should be placed, and so on. A wide range of 3-dimensional task skills are required in any assembly or disassembly process — placing one part in, under, or next to another part, for example. When the dimension of time is introduced, motions and sequences of motions can also be learned. In this paper we concentrate on 1-dimensional tasks.

## 2. Instruction dialogues

Each instruction dialogue is seen as a sequence of trials or steps. On each trial, the robot responds to a natural-language command from the operator by taking the action described in the command. This response is followed by feedback from the operator indicating whether the response was acceptable or not. The feedback is itself a natural-language command, either a congratulatory command such as *That's fine!* or *OK!* which indicates that the response was acceptable, or a corrective command such as *Further to the left!* or *That's way out!* which indicates that the response was unacceptable. When the response is acceptable we refer to it as a "hit," when unacceptable a "miss." After a hit, the operator typically repeats the original command to check that the robot has learned to respond appropriately to it.

The corrective feedback is nondeterminate in that it does not let the robot know exactly what its response should have been. It merely indicates what the robot can do to improve its response on subsequent trials. Typically, there is no one correct response on a trial anyway but a range of acceptable responses within the target interval or region, or a range of motion paths. In addition, the operator will often not be able to provide determinate feedback. He or she will have a target interval, region, or motion in mind but will be unable to specify the endpoints of the interval or the exact coordinates of the region or the precise trajectory of the motion path. Instead the operator will use his or her judgement to determine whether the observed response appears acceptable or not. Although an essential guide to the robot's learning, that judgement is not infallible.

There are three categories of feedback: congratulatory feedback given after a hit (*Good!*, *That will do!*, *OK!*, *Fine!*, etc.), positional feedback given after a miss (*To the right!*, *Much further to the right!*, *A little bit to the right!*, *Too far right!*, *Much too far right!*, *Not that far!*, *More!*, *Again!*, *Further still!*, *A bit more!*, etc.), and accuracy feedback given after a miss (*Be more careful!*, *No need to be so cautious!*, *Slower!*, *Faster!*, *Slower next time!*, etc.). Figure 1 gives an example of an instruction dialogue. Note that there is no immediate motor response to congratulatory feedback or accuracy feedback. In either case, the robot waits for the original command to be repeated or for positional feedback.

| | | Robot's response |
|---|---|---|
| Original command: | *Put the wrench on the shelf!* | $x_0$ |
| Positional feedback: | *Not that far left!* | $x_1$ |
| Congratulatory feedback: | *That's fine!* | — |
| | | |
| Original command: | *Put the wrench on the shelf!* | $x_2$ |
| Positional feedback: | *A little further to the right!* | $x_3$ |
| Positional feedback: | *More!* | $x_4$ |
| Accuracy feedback: | *Be more careful!* | — |
| Positional feedback: | *A little to the left now!* | $x_5$ |
| Congratulatory feedback: | *Good!* | — |

Figure 1: Example of an instruction dialogue

It is assumed that on each trial the state of the robot with respect to the skill being taught is represented by a probability distribution. This distribution enters into the interpretation of the original command (the one that describes the skill being learned) and it comes into play in the interpretation of all feedback from the operator. In addition, the distribution changes in response to the operator's feedback, which in turn alters the interpretation of all subsequent commands.

The distribution plays another important role. It represents the target interval, region or motion associated with the skill being learned. For 1-dimensional tasks, therefore, we can use a single distribution of one variable. We use the notation $k_{m,v}(x)$ where $m$ is the mean of the distribution, $v$ the variance, and $x$ a value of the response variable. If on trial $n$, $m = m_n$ and $v = v_n$ and the robot responds to the original command, then the probability that the response on this trial will lie between $a$ and $b$ is given by $\int_a^b k_{m_n,v_n}(x)dx$. After the robot has made response $x_n$ on trial $n$ in response to a natural-language command, i.e., after it has selected a point on the response continuum as evidenced by its moving to that point in response to the command, one of the two kinds of events described earlier occurs: the robot is either told that it was unsuccessful, i.e., it missed the target interval, or that it was successful, i.e., it landed within or hit the target interval. Known as a *smearing* or *smoothing* function, $k_{m,v}(x)$ has the effect of spreading the effect of feedback at a point $m$ around $m$ on the continuum of responses.

The question we face in designing the robot so that it learns from its interaction with the operator is as follows: What effect should a hit or miss have on the function $k_{m,v}(x)$? Before we can answer that question, we must describe the choice of distribution for 1-dimensional tasks and the interpretation of commands.

## 3. The beta distribution

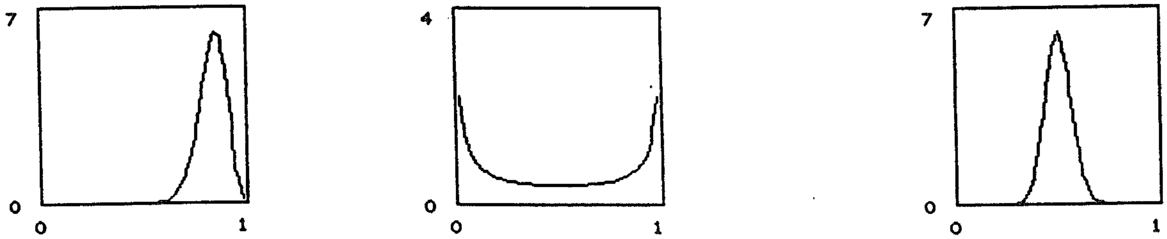
The probability distribution we use for 1-dimensional tasks is the beta distribution. It has two parameters $\alpha$ and $\beta$ and is defined as follows for $0 < x < 1$ ($\Gamma$ designates the gamma distribution):

$$\beta(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}x^{\alpha-1}(1-x)^{\beta-1}, \qquad \alpha > 0, \ \beta > 0.$$

The mean, m, and variance, v, of the distribution are calculated as follows:

$$m = \frac{\alpha}{\alpha+\beta}, \qquad v = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}.$$

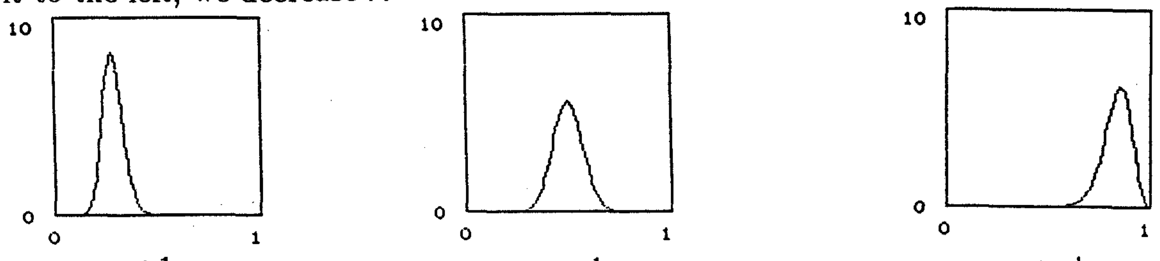
The beta distribution has several properties that make it suitable for our purposes. First, its usefulness in models of learning has already been demonstrated in studies of human learning (see [8] – [11]). In addition, with appropriate values of $\alpha$ and $\beta$ the distribution quite effectively represents target intervals we wish the robot to learn. For instance, with $\alpha = \beta = 1$ the distribution is the uniform distribution on (0, 1) and represents the intervals described by phrases such as *anywhere on the shelf* or *anywhere in front of the desk.* (The (0,1) interval must, of course, be mapped onto the actual interval of interest for the task — that corresponding to the length of the shelf or the width of the desk, for instance.) The uniform distribution is also generally used to represent the state of the robot before all instruction starts. Figure 2 shows various curves for different choices of $\alpha$ and $\beta$ along with the natural-language expressions describing the intervals associated with the curves. The response variable is plotted along the x-axis, the probability distribution along the y-axis.



*the righthand side* *anywhere but preferably at either end* *anywhere around the middle*

Figure 2: The representation and description of target intervals

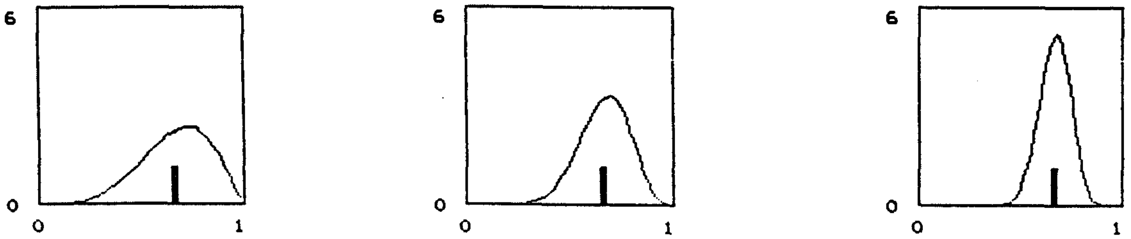
A computationally efficient way of altering the initial distribution and all subsequent distributions is to change the values of $\alpha$ and $\beta$ directly. As long as the ratio $r = \alpha/\beta$ remains constant, the mean of the distribution does not change. If $r = 1$ (i.e., $\alpha = \beta$) then the distribution is symmetrical around the midpoint of the $(0, 1)$ interval. If $r > 1$ (i.e., $\alpha > \beta$), the distribution is shifted to the right. If $r < 1$ (i.e., $\alpha < \beta$), the distribution is shifted to the left. See Figure 3. To shift a distribution to the right, therefore, we increase $r$. To shift it to the left, we decrease $r$.



$r < 1$ $r = 1$ $r > 1$

Figure 3: Adjusting the distribution to the left and right

If the ratio of $\alpha$ to $\beta$ is kept fixed, increasing $\alpha$ (and $\beta$ correspondingly) reduces the variance without altering the mean. Reducing the variance has the effect of increasing the accuracy of the robot's subsequent responses to positional feedback. It further has the effect (for $\alpha, \beta > 1$) of reducing the target interval. See Figure 4 for examples in which $r = 2$. The mean is indicated by a vertical bar.



$\alpha = 5,\ v = .026,\ m = .667$ $\alpha = 10,\ v = .014,\ m = .667$ $\alpha = 25,\ v = .006,\ m = .667$

Figure 4: Adjusting the variance of the distribution

## 4. Interpretation of commands: the robot's motor response

As the preceding discussion has shown, the beta distribution can be altered through direct manipulation of the parameters $\alpha$ and $\beta$. Before we can describe in detail how these changes are brought about, however, we must discuss the link between the probability distribution and the interpretation of the natural-language commands. We will not discuss in detail the process by which commands are interpreted by the robot nor the control mechanisms that enable the robot to move in response to a command. Our earlier publications describe this work in some detail as implemented for the robotic aid, a device being developed for people with severe physical disabilities by the Rehabilitation and Research Center of the Veterans Administration in Palo Alto ([2], [7]). More general discussions of our work on natural-language understanding for robots can also be found in other publications (see [1], [3] - [6]).

Given limited space, our primary concern here is to show how a probability distribution enters into the interpretation of a command, specifically, the role it plays in generating the robot's motor response. For simplicity, we present highly schematic interpretations of sample commands. We use as our example of a command whose meaning must be learned by the robot through instruction (the original command in an instruction dialogue) *Put the wrench on the shelf!*

**Original command:** Suppose *Put the wrench on the shelf!* is translated into the following robot plan: SEQ( Grasp-Free(wrench), Move-Gripper(xval,yval,zval), Place(wrench) ). SEQ indicates that the robot actions Grasp-Free, Move-Gripper, and Place are executed in sequence. The point in 3-dimensional space that the robot must move its gripper to is given by the triple <xval, yval, zval >. Values for yval and zval are obtained from the robot's knowledge base (information that is provided either by the sensors or through earlier instruction). The value of xval, however, the point along the length of the shelf where the wrench must be placed, is obtained by sampling the $(0,1)$ interval using the probability distribution that is currently associated with the command. Initially, this distribution is the uniform distribution and any point along the length of the shelf is as likely as any other to be selected. In general, if $k_{m_n,v_n}(x)$ is the current probability distribution (step $n$ of the instruction dialogue has just been completed), a response to the original command at step $n+1$ is generated by the following sampling procedure: Take the cumulative probability distribution $K_n(x) = \int_0^x k_{m_n,v_n}(x)dx$, generate a random number $y$ between 0 and 1, and find $x$ such that $K_n(x) = y$. This $x$ gives the robot's response at step $n+1$. We write $x_{n+1} = sample(k_{m_n,v_n})$.

**Positional feedback:** Suppose the interpretation of the command *Much further left!* is DO( Pilot(Translate, Left) UNTIL Distance-Covered(Translate, Left, <distance>) ). This interpretation ensures that the gripper is moved left until the designated distance (indicated by <distance>) has been covered. DO ... UNTIL is one of the control structures, like SEQ described above, that control the temporal and logical order in which basic robot actions such as Pilot and Distance-Covered are executed. The Pilot action moves the gripper in the direction indicated by its second argument; the motion is either translational or rotational as indicated by its first argument. The actual distance covered depends on the current probability distribution in the following way. If the variance, v, of the distribution is small, a move leftward or rightward should be correspondingly small. If the variance is large, the move should be correspondingly large.

In addition, a correction such as *Much further left!* must generate a larger move than a correction such as *Left just a little!* Clearly, the adverbial, and sometimes the adjectival, component of the language must be allowed to make its contribution. Currently, all positional corrections fall into three equivalence classes: those that generate a relatively small move, those that generate a relatively large move, and those that generate a move of intermediate extent. Three range constants are therefore needed, $c_L$, $c_M$, and $c_S$ for "large," "medium," and "small" moves respectively. These constants are set for the robot and the task (in our current instruction model, $c_L = 2$, $c_M = 1$, and $c_S = .5$). The total displacement for a move is given by the product of the appropriate range constant and the square root of the variance. A request for a "large" move therefore generates a displacement of $c_L\sqrt{v}$. The robot must never move outside the $(0,1)$ interval; extreme leftmost and rightmost points (*minx* and *maxx* respectively) are therefore designated. They are set for the robot and the task. It is important to note that although the response to positional feedback is not generated by a sampling procedure on the underlying probability distribution, the actual extent of the move depends on that distribution.

**Accuracy feedback and congratulatory feedback:** The interpretation of a command giving accuracy feedback during instruction does not generate any motor response from the robot. Nor does the interpretation of a command giving congratulatory feedback. Both forms of feedback change the probability distribution, however.

## 5. Effect of feedback on the probability distribution

**Positional feedback:** For positional feedback, we want to shift the distribution to the left or the right. As indicated earlier in our discussion of the beta distribution, we can accomplish this shift by adjusting the ratio $r$, increasing it for a shift to the right and decreasing it for a shift to the left. As before, the adjustments fall into three equivalence classes, large, medium, and small. The same three range constants ($c_L$, $c_M$, and $c_S$) are used to determine the amount of increase or decrease in $r$. Keeping $\alpha$ fixed, we decrease (or increase) $\beta$ by the amount necessary to increase (or decrease) $r$ by the appropriate range constant.

**Accuracy feedback:** For feedback such as *Be more careful!* which asks for greater accuracy in the robot's responses, we decrease the variance of the current distribution. For feedback such as *No need to be so cautious!* which encourages the robot to make larger adjustments, we increase the variance. As indicated by our earlier discussion of the beta distribution, the variance can be increased or decreased by increasing or decreasing $\alpha$. The amount of increase or decrease should be in proportion to the value of $\alpha$ itself. We determine the appropriate amount of increase or decrease from the slope of the tangent to the variance, $f_\beta(\alpha) = (\alpha\beta)/((\alpha+\beta)^2(\alpha+\beta+1))$, at $\alpha$. Specifically, if we are at step $n+1$ of the instruction dialogue ($\alpha = \alpha_n$, $\beta = \beta_n$), we increase or decrease $\alpha$ by the square root of the absolute value of the derivative of the variance (with respect to $\alpha$) evaluated at $\alpha_n$, i.e., the square root of the absolute value of $f'_{\beta_n}(\alpha_n)$. The ratio of $\alpha$ to $\beta$ is kept constant to preserve the mean. We therefore alter $\beta$ correspondingly by setting $\beta_{n+1}$ to $\alpha_{n+1}/r_n$.

**Congratulatory feedback:** For congratulatory feedback, the mean of the distribution shifts to the robot's response $x$ on that trial and the variance is halved. A new probability distribution is produced by solving for $\alpha$ and $\beta$ given this new mean and new variance.

Each kind of physical robot will have its own performance constraints — both accuracy limits and, in the case of a manipulator, limits of reach. We have already introduced explicit range constants to determine the extent of a move to the left or right. Other performance constraints are reflected in maximum and minimum values for $\alpha$ and $\beta$. In our current model we have set $max\alpha\beta = 30$ and $min\alpha\beta = .25$. As with the range constants, these constants are set for the robot and the task to get a proper balance between sensitivity to feedback and realistic performance demands. The computation that increases or decreases $r$ for positional feedback at step $n+1$ takes $max\alpha\beta$ and $min\alpha\beta$ into account with the constraint that $\alpha_n/max\alpha\beta < r_{n+1} < \alpha_n/min\alpha\beta$. The computation that alters $\alpha$ and $\beta$ for accuracy feedback also takes $max\alpha\beta$ and $min\alpha\beta$ into account. To ensure that $\alpha_{n+1}$ and $\beta_{n+1}$ are both greater than $min\alpha\beta$ and smaller than $max\alpha\beta$, if $\alpha$ is being increased and $r_n > 1$, it obeys the constraint that $r_n min\alpha\beta < \alpha_{n+1} < max\alpha\beta$. Similar constraints hold for $r_n < 1$, and when $\alpha$ is being decreased. In response to congratulatory feedback, $\alpha$ and $\beta$ are permitted to exceed $max\alpha\beta$ up to a current maximum of 100 (once again, a constant set for the robot and the task). At that time $max\alpha\beta$ is reset to the new $\alpha$ or $\beta$, whichever is greater.

We note here that we are assuming congratulatory feedback to be thoroughly effective in that the mean and variance of the distribution always change in response to the feedback. It is common practice to introduce a learning parameter $\theta$ into the model and to assume that with probability $\theta$ feedback is effective on any trial (the distribution changes), and with probability $(1-\theta)$ it is not effective (the distribution stays the same). We have in effect set $\theta$ to 1 in our instruction model. We made this choice in part because of the explicitness of verbal instructions — congratulatory feedback, in particular, is hard to misconstrue — but also because we are assuming that the robot's total cognitive resources are dedicated to learning from instruction. This assumption would change if the robot were simultaneously attending to some other task. For example, in a space application the robot might be monitoring the status of a process for an emergency condition to which it had to respond.

Table 1 summarizes the core computations in our current instruction model using representative commands. It assumes that $k_{m_n,v_n}(x)$ is the current probability distribution, with parameters $\alpha_n$ and $\beta_n$. The mean of the distribution is $m_n$, the variance is $v_n$, $r_n = \alpha_n/\beta_n$, $x_n$ is the robot's most recent motor response, $f'_{\beta_n}(\alpha_n)$ is the derivative of the variance with respect to $\alpha$ at $\alpha_n$, and $c_S$, $c_L$, $max\alpha\beta$, $min\alpha\beta$, $maxx$, and $minx$ are all performance constants as described earlier.

| Response and feedback | Distribution changes and robot's motor response |
|---|---|
| **A HIT:** | |
| *Good!* | $m_{n+1} \leftarrow x_n$<br>$v_{n+1} \leftarrow v_n/2$ |
| **A MISS:** | |
| **Positional Feedback:** | |
| *A little to the left!* | $r_{n+1} \leftarrow \begin{cases} r_n - c_S & \text{if } r_n - c_S > \alpha_n/max\alpha\beta \\ \alpha_n/max\alpha\beta & \text{otherwise} \end{cases}$<br>$x_{n+1} \leftarrow \begin{cases} x_n - c_S\sqrt{v_n} & \text{if } x_n - c_S\sqrt{v_n} > minx \\ minx & \text{otherwise} \end{cases}$ |

*Much further right!*

$$r_{n+1} \leftarrow \begin{cases} r_n + c_L & \text{if } r_n + c_L < \alpha_n / min\alpha\beta \\ \alpha_n / min\alpha\beta & \text{otherwise} \end{cases}$$

$$x_{n+1} \leftarrow \begin{cases} x_n + c_L\sqrt{v_n} & \text{if } x_n + c_L\sqrt{v_n} < maxx \\ maxx & \text{otherwise} \end{cases}$$

**Accuracy Feedback:**

*Be more careful!*

$$\alpha' \leftarrow \alpha_n + \sqrt{\left|f'_{\beta_n}(\alpha_n)\right|}$$

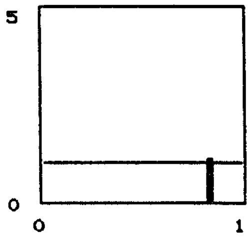
$$\alpha_{n+1} \leftarrow \begin{cases} r_n min\alpha\beta & \text{if } \alpha' < r_n min\alpha\beta \\ r_n max\alpha\beta & \text{if } \alpha' > r_n max\alpha\beta \text{ and } r_n < 1 \\ max\alpha\beta & \text{if } \alpha' > max\alpha\beta \text{ and } r_n > 1 \\ \alpha' & \text{otherwise} \end{cases}$$

$$\beta_{n+1} \leftarrow \alpha_{n+1} / r_n$$

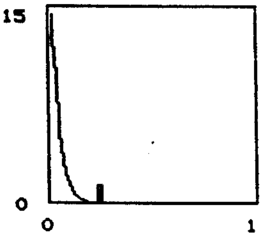
Table 1: Core computations in the 1-dimensional model

## 6. Sample instruction sessions

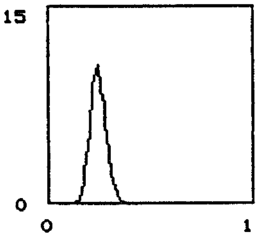
We now show two sample instruction sessions. The 1-dimensional skill being taught to the robot is where to stop in front of double swing doors so that the robot can enter through the righthand door when it opens. The doors swing out; the robot is on the outside. Taking the combined width of the two doors as the interval of interest, we want to teach the robot to select any point somewhat to the left of the midpoint. The robot begins in each case with the beta distribution initialized to $\alpha = \beta = 1$. We show at each step what was said, what the robot's motor response was (if there was indeed a response), and what the resulting distribution looks like. The robot's response is indicated by a vertical bar. The response variable is plotted along the x-axis, the probability distribution along the y-axis. The first instruction dialogue was quickly successful (only three steps), although in subsequent instruction the operator may wish to shift the responses nearer to the middle. The second dialogue lasted longer (nine steps). The y-axis on each graph goes from 0 to 5 or the least integer greater than the maximum distribution value plotted.



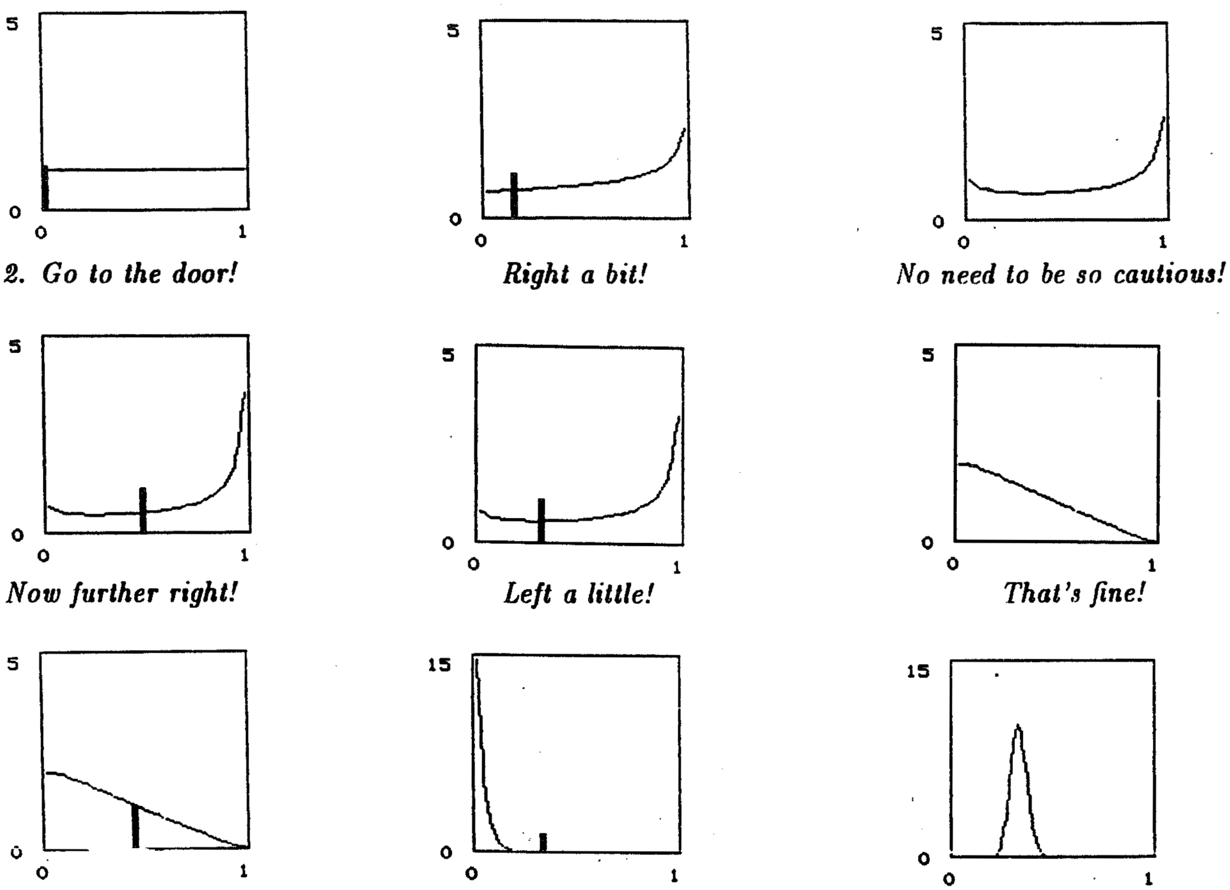
*1. Go to the door!*



*That's much too far right !*



*That's fine!*



*2. Go to the door!* *Right a bit!* *No need to be so cautious!*

*Now further right!* *Left a little!* *That's fine!*

*Go to the door again!* *Left a little!* *Good!*

Figure 5: Two sample instruction sessions

Note the somewhat surprising shape of the distribution after the first trial in the second dialogue. This is due to the particular form of learning, i.e., the rule for changing the beta distribution, we have adopted. We are in the process of exploring and testing other learning rules for comparison.

## 7. Concluding remarks

The learning procedure we have described is congenial to a Bayesian viewpoint, but embodies a number of specific learning assumptions that go beyond a purely Bayesian framework. For example, it is not part of Bayesian theory to postulate how specific words of feedback should change the underlying response probability distribution used by the robot. Such interpretation is more in the spirit of mathematical learning theory (see [8] – [11]).

Directions for future work are clear. First, we must make extensions to the model for 2-dimensional and 3-dimensional tasks and for the dimension of time. We plan to move to 2-dimensional tasks immediately. The computation problems are more severe for probability distributions whose domains are arbitrary surfaces. Our initial effort will use some strong simplifying assumptions. First, we use as the 2-dimensional distribution the product of two beta distributions, which assumes we can assume for working purposes independence in probability along the two coordinate axes. Second, we fit the shape of the surface of interest by conditionalizing the 2-dimensional distribution to the given surface as domain. The rules for interpreting English instructions will inevitably be more numerous and complex. The instructions themselves will certainly be more complex, for example, *To the left and up!*, *A good deal further back but not so far left!* Our ultimate objective is to have a substantial grammar of English that generates the rules of learning for these higher-dimensional tasks.

We also plan to implement the learning models on an actual robot system. We expect this implementation to greatly expand the range of feedback expressions we consider, and to provide a realistic test for the performance constants discussed earlier. An important overall change we plan for the models is to allow feedback to be given, and to take effect, while the robot is still responding to the previous command. So, for instance, if the robot is moving forward in response to a command from the operator, the operator should be able to say *That's far enough!* and have the robot stop where it is. Not only is this form of feedback entirely natural, it makes for safer robot operation and it should provide faster learning.

# References

[1] Crangle, C. (in press). On saying *Stop* to a robot. *Language and Communication, 9, No. 1.*

[2] Crangle, C., Liang, L., Suppes, P, & Barlow, M. (1988). Using English to instruct a robotic aid: an experiment in an office-like setting. In *Proceedings of the International Conference for the Advancement of Rehabilitation Technology, 25-30 June, 1988, Montreal.* (pp. 466-7).

[3] Crangle, C., & Suppes, P. (in press). Geometrical semantics for spatial prepositions. In P. French, T. Uehling, & H. Wettstein (Eds.) *Midwest Studies in Philosophy Vol. 13, Contemporary Perspectives in the Philosophy of Language II.*

[4] Crangle, C., & Suppes, P. (1987). Context-fixing semantics for an instructable robot. *International Journal of Man-Machine Studies, 27*, 371-400.

[5] Crangle, C., Suppes, P., & Michalowski, S. (1987). Types of verbal interaction with instructable robots. In G. Rodriguez. (Ed.) *Proceedings of the Workshop on Space Telerobotics*, (JPL Publication 87-13, Vol. II). (pp. 393-402). Pasadena, California: NASA Jet Propulsion Laboratory.

[6] Maas, R.E., & Suppes, P. (1985). Natural-language interface for an instructable robot. *International Journal of Man-Machine Studies, 22*, 215-240.

[7] Michalowski, S., Crangle, C., & Liang, L. (1987). A natural-language interface to a mobile robot. In G. Rodriguez. (Ed.) *Proceedings of the Workshop on Space Telerobotics*, (JPL Publication 87-13, Vol. II). (pp. 381-392). Pasadena, California: NASA Jet Propulsion Laboratory.

[8] Suppes, P. (1959). A linear model for a continuum of responses. Reprinted in R.R. Bush and W.K. Estes (Eds.) *Studies in Mathematical Learning Theory.* Stanford: Stanford University Press. Pp. 400-414.

[9] Suppes, P. (1964). Some current developments in models of learning for a continuum of responses. (Discrete Adaptive Processes Symposium, American Institute of Electrical Engineers, June 1962.) *The Institute of Electrical and Electronics Engineers Transactions on Applications and Industry, 83*, 297-305.

[10] Suppes, P. & Zinnes, J.L. (1966). A continuous-response task with nondeterminate, contingent reinforcement. *Journal of Mathematical Psychology, Vol. 3, No. 1*, 197-216.

[11] Suppes, P. (in press). Current directions in mathematical learning theory. In E. Degreef and E. Roskam (Eds.) *Progress in Mathematical Psychology Vol. II.*

N90-29885

# Interset: A Natural Language Interface for Teleoperated Robotic Assembly of the EASE Space Structure

Daniel K. Boorsma

Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

## Abstract

A teleoperated robot was used to assemble the EASE (Experimental Assembly of Structures in Extra-vehicular activity) space structure under neutral buoyancy conditions, simulating a telerobot performing structural assembly in the zero gravity of space. This previous work carried out at NASA and MIT used a manually controlled teleoperator as a test bed for system performance evaluations. From these results several Artificial Intelligence options were proposed. One of these was further developed into a real-time assembly planner written in Prolog.The interface for this system is effective in assembling EASE structures using windowed graphics and a set of networked menus. As the problem space becomes more complex and hence the set of control options increases, a natural language interface may prove to be beneficial to supplement the menu based control strategy. This strategy can be beneficial in situations such as: describing the local environment, maintaining a data base of task event histories, modifying a plan or a heuristic dynamically, summarizing a task in English, or operating in a novel situation.

Interset is conceptually a natural language extension to an assembly planner approach, it attempts to map a sentence into a "speech act network", the representation emphasizing spatial and topological relations about work pieces and key processes.There are four general system goals:1) provide a natural language process capable of verbal dialog about the telerobot and EASE structures, 2) maintain a database of control events with respect to assembly conditions over the lifetime of the construction project such as, the list of completed subtasks, 3) update the systems contents to reflect a changing world model, and 4) modify or test system parameters more rapidly such as avoid using a sensor and observe the resulting degraded system. Currently, the prototype is written in Scheme (lisp) and implemented using a VAXstation 2000, IBM RT, and MacIntosh II workstations. Interset consists of the following structures: a 'slightly' grammatical parser, a deductive assertion maintenance system, rule application control, event and time referencing via a filter, perspective stereo graphics, several dozen small knowledge bases including planning representations and rules.The program demonstrates an AI application of natural language and a semi-autonomous telerobot caring out the assembly of the EASE space structure.

## Introduction

Interset's general theme is an isomorphism between single sentences about concrete objects mapped to generalized concrete object models. Sentences are viewed as speech acts in an action based "commitment" framework [14]. Interset is the name of a system that is written in Scheme [13] and runs on either a MacIntosh or a MicroVax. The sentence understanding capability uses a schema based approach. Sentences are parsed and analyzed for "shape" before schema activations. As a result, space station truss assembly may prove be an applicable domain because of the need to carry out a variety of assembly information processing tasks about concrete objects. Specifically, the MIT EASE space structure (a tetrahedral modular element for construction of the space station truss network). In addition, provide an effective interface with the astronaut and the teleoperated robot.

Natural Language (NL) fits into a larger context of how to get the most effective Man Machine Interface (MMI) for a task. The computer is really a type of media for disseminating structural, parametric and control information. "Media Computations" are two concepts put together, MMI plus an AI system that understands the target processes and can display that understanding effectively [7]. Media computations can use: windows, keyboard, mouse, data glove, joy stick, EVA cuff controller, voice, heads-up display, etc. What's important is the interface understands enough to get the operator a "good" understanding of the key processes. Increases effectiveness, and carries out recurring operations. In an environment such as space operations there will be many complex systems requiring effective interface technologies.

In the space vehicle systems context: control, dynamics, solid mechanics, thermodynamics, fluid dynamics, and propulsion characteristics will continually undergo change. Even when the systems are functioning properly control and monitoring will be required expending resources to carry out these functions. Simplifying communications during telerobotic operations could provide a useful tool for managing this complexity. English input and output can be helpful to: reduce mistakes, improve operator attention, increase productivity and probably understand our own communications needs better and what is important when discussing space construction concepts. Essentially, the focus is to allow sentence and/or sentence fragments to be used for the construction operation queries. Also relieve the operator from performing repetitive cognitive operations that would normally require ongoing attention. Such as looking through lists or having to remember context dependent logistics. This approach will unfortunately have its detracting points.

First of all the task environment is still evolving this makes knowledge acquisition a moving target in that the representations in the system will have to change based on engineering changes elsewhere in the system. In operation ambiguous interpretation could cause serious accidents. While the system may or may not have influence on the process control functions of teleoperation, faulty logistical information could be just as hazardous. An argument stating that, "Operational telerobotics does not really need logistical support and that it's really quite straight forward", could be made. Another argument is natural language is not advanced enough yet to be used as a reliable technology. Better would be to use menus or specialized command languages. All of these have a certain amount of validity.

## The Environment

There are a variety of resources in low earth orbit they are: shuttle and crew, pallet of spares and connectors, beam assembly teleöperator (BAT), previous construction, astronaut that can be EVA or IVA, assembly teleoperation work station (computer, console, etc.), telerobotic operator (mission specialist), subassembly construction area, final construction work area and perhaps an EVA work station [1]. The EASE structure that is assembled at neutral buoyancy is analogous in structure to space station design criteria. EASE Design and Development is part of a joint effort between Marshall Space

Flight Center and MIT Space Systems Laboratory the Principal Investigator is Dr. David Akin. The goals of the collaboration is to compare construction techniques between the Neutral Buoyancy Simulator (NBS) and Low Earth Orbit (LEO). Build a rigid stable structure that is simple and does not clutter the work environment and demonstrate a realistic subassembly for the final space station truss assembly.

The assembly consists of struts and interlocking connectors which are called clusters. Interlocking 6 struts and 4 clusters to form a tetrahedron. EASE structures can be attached together to form truss assembly. The truss assembly can be decomposed in a variety of ways (for example use the octahedron as the motif rather than the tetrahedron). Interset uses a combination of connection topology and a tetragonal space group (4-3-fold which is not an example of a space station truss assembly, that group is actually isometric where a = b = c and the planner angles are at 90 degrees) [4]. Construction is carried out at two primary sites the shuttle subassembly staging area and the truss assembly area. A subassembly is connected at the first staging area and is then transported to final assembly area. Where it is attached to the truss end. Subassemblies are constructed with the aid of a planner written in prolog. The teleoperator using a joy stick and commands the BAT to construct a tetrahedron. The BAT then moves the new assembly to the truss end where is attached. During this process English commands could be given to operate BAT or to request logistical information to improve operations in case of error, malfunction or anomaly.

## A "Slightly" Grammatical Parser

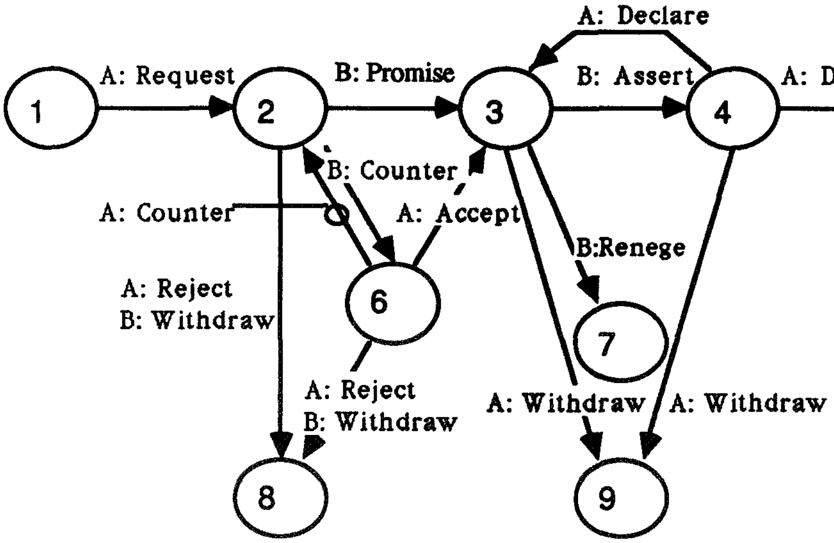
Given that a sentence in natural language corresponds to an internal world model. It is possible to create a formal representation system. Then, For any relevant fact about the world there can be a corresponding structure in the representation. Systematic formal operations in representation structures can be devised to carry out valid reasoning. However considerable effort to make the schemas and knowledge representations is usually required. The reference to the adjective slightly in the name of the parser procedure addresses two limitations. One, having a perfect syntactic parse is probably not needed if the domain is fairly simple and there is potentially a large set of knowledge that can correctly interpret the intent of the sentence in the discourse context. Two, using and creating ungrammatical sentence fragments is actually the rule of most communication. In addition, periods of duress and quick decision have a tendency to increase the noise in the sentences (such as: extraneous cursing, miscuing of material, etc.) So an approach of this sort is really of ergonomic practicality.

Natural language activity in Interset consists of a variety of functional areas: actors model, a task state space, Interset status information, assembly and construction model, a Filter for the present event space, speech act instantiation, execution control if an unambiguous sentence is "understood", schemas for semantic interpretation, deductive query section, lexicon with endings and meanings. Also, speech act networks, sentence shape, word shapes, A rule based parser and illocutionary act (input sentence).

Walking through the system operation first that occurs the illocutionary act. Words are found in the Lexicon. The parser uses syntactic rules using a head first scan. Syntactic analysis often times can be used to disambiguate word meaning [15]. Predicates are added at this time to label the sense of the word meaning based on the parse. These predicates are still in flux and will not be developed in this paper. The resulting parse tree represents the syntactic categories (noun, adjective, verb, etc.), word root with suffix, infix, or prefix, and the word meaning senses. Word meaning senses is used to characterize the sentence in a general sense (the "shape" of the sentence).

Sentence shape can be used to make decisions about whether an appropriate schema exists and where in the action network the conversation probably is. A "Schema" of "Frame" contains expectation values, defaults programs, connections to other objects or schemas, and identifies itself as being of a general "type" of thing. We can use all of these types of information potentially to help resolve the what the sentence or sentence fragment might mean. Meaning corresponds to an appropriate representation or process in the data base or as often times referred to as the knowledge base. Sentence shape can be viewed as a vector pointing to the "space" of schemas [12]. If a schema is not found then a "neighbor" could be selected, if this is not appropriate then the space is marked to indicate that a sentence was indexed to this point but nothing was there. Use of learning algorithm could provide knowledge acquisition at this point if one was available to create additional meaning structure.

Dialogues about limited domains often follow similar patterns or chains of discourse (ie. the action network below). The method to generate these is to listen to conversations made by astronauts during the construction process to see what kinds of references are made, the recurrent chains, and the range of sentence types. Then building on the structure as needed when new situations appear they will be properly encoded as schematic information resulting in a robust language understanding capability.



**Schema extended to conversation for action (nodes represent states in the conversation)**

Winograd and Flores - Understanding Computers and Cognition.

The main property of the shape metric is categorical and/or based on number of nodes away rather than on a continuous means end analysis. The categories are based on the shape predicates. "Nearness" can help disambiguate fragment interms of context. The temporary activation of a schema acts to preserve relevant data to be used for interpreting the next sentence or in generating a response. The "History" of activation can be used to generate explanations and activity logs.

## Knowledge Base and Representations

Process entities, things that can operate upon the domain (such as the BAT). Are important for determining relative position and orientation w.r.t perspective (ie. up, down, left, right, etc.). Process entities also contain status information about their functional parameters (ie. control parameters, orientations and local diagnostics, sensor information, if available).

Task state space. Building the truss assembly is probably best thought of as a state space. Tasks will generally be completed based on a depth first search.

Interset status information. Interset like all systems will have many states that will be relevant to completing the set of construction tasks this information will be accessible to the teleoperator and the ground station as needed. Updating interaction preferences inregards to media computations will be controlled here also.

Assembly and Construction Model. Contains knowledge and rules for assembly. Contains the memory model for space groups, topologly, and sets.

## Human Factors

- Interset can be used in a relative operation cycle between 5 seconds and hours. 5 Seconds is the lowest bounds because of typing and minimal computations to carry out the operations.

- Support logistics and information requests are the primary goals at present.

- If interacting with process control systems the BAT, control should be controlled at a fairly high level (ie. "Put that strut together with the cluster." or "Transport that subassembly to the truss work end." ) to avoid unnecessary detail.

- Provide task cues.

- Allow for imprecise labels or forgotten terms.

- Viewing human operator w.r.t Single Channel Hypothesis [8] problems exist with modalities of control and coordination of tasks (ie. what level should tasks be abstracted to?)

- Optimally, acoustic input of words.

## Conclusion

Interset maps English sentences to representations used to reason geometrically about assembling structures that are similar in nature to space station components. The speech act network is an important segment of schemas which are used to facilitate mapping sentences to the appropriate internal representations. This approach is a familiar one albeit requires considerable knowledge acquisition effort. The overall interface operates in media computation paradigm. Human factors are of great importance in making the system useful. Ease of use and avoiding errors probably will prove to be important capabilities.

**References:**

[1] Aerospace Environmental Systems Conference #16,The Development of an EVA Universal Work Station, Miles Moffatt and Fred Abeles (Grumman Corp.), Society of Automotive Engineers, Inc., July 1986.

[2] Artificial Intelligence Applications in Teleoperated Robotic Assemble of the EASE Space Structure, Herbert E. M. Viggh, February 1988, MIT SSL#6-88.

[3] Chomsky's Universal Grammar an Introduction, V. J. Cook, Basil Blackwell Ltd. 1988.

[4] Contemporary Crystallography, Martin Buerger, McGraw-Hill Inc.1970, pp. 1-44.

[5] Counterexamples in Topology, Lynn Arthur Steen and J. Arthur Seebach, Jr., Springer-Verlag New York Inc.1978, pp. 1-30.

[6] The English Imperative, Eirlys Davies, Croom Helm Ltd.1986.

[7] The Media Lab, Stewart Brand, Penguin Books, 1987.

[8] Motor Control and Learning a Behavioral Emphasis, R. Schmidt, Human Kinetics Publishers Inc., 1982.

[9] Optimization Issues in the Design and Control of Large Space Structures, Edited by Manohar P. Kamat, American Society of Civil Engineers, 1985.

[10] Scheme Reference Manual (MIT), Scheme Release 7 Draft, October 18, 1988.

[11] Space Groups for Solid State Scientists, G. Burns, Academic Press 1978.

[12] The Society of the Mind, Marvin Minsky, Touchstone Book, 1985.

[13] The Structure and Interpretation of Computer Programs, Hal Abelson, Gerald Sussman and Julie Sussman, 1985.

[14] Understanding Computers and Cognition, Terry Winograd and Fernando Flores, Addison-Wesley Publishing Inc. 1987.

[15] UNITRAN A Principle-Based Approach to Machine Translation, Bonnie Jean Dorr, MIT AI Lab, December 1987.

# TELEROBOTIC SPACE OPERATIONS

# ESTABLISHING VIABLE TASK DOMAINS FOR TELEROBOT DEMONSTRATIONS

Wayne Zimmerman
Telerobot Testbed System Engineer
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

## Abstract

A suite of telerobotic tasks has been compiled and assessed for the purpose of selecting viable tasks for near and far term laboratory demonstrations. The primary intent of developing the task suite is to provide some technical guidelines, with supporting data, for focusing NASA laboratory demonstrations toward application domains that address a wide array of potential telerobot tasks and required technologies. This wide application would then result in a rich technology development environment to meet the broad task requirements of a system such as the Flight Telerobot Servicer (FTS).

This paper describes the methodology and results of the telerobot task assessment, including a ranking of the final select suite of major tasks. The study approach, database, and results of the task ranking computation are presented along with guidelines for both interpreting the task ranking results and setting programmatic objectives based on these results. The report also provides detailed data about the task candidates and their respective levels of complexity, task primitive actions, and the actual relative measures of task worth as associated with key tradeoff variables such as cost, available research resources, technology availability, and importance to the user community.

## Introduction

The primary purpose of this task study was to compile a list of tasks that represented viable candidates for laboratory demonstrations, and satisfied two major constraints:

1. The tasks must clearly demonstrate application to a real-world user problem in the space environment, such as Space Station assembly or servicing.
2. Selection of the suite of tasks must reflect existing resource constraints within NASA telerobot research community.

In the process of structuring the task assessment and developing the suite of demonstration tasks, it became clear that the assessment contained additional information that could be useful to the telerobot research community:

1. The assessment, through its structure, provides a means of rationalizing the task selection process.
2. The assessment provides a traceable means for reasoning why one task tends to represent a better demonstration target than another.
3. The assessment, through its methodology and supporting task-related data (e.g., cost, technology contribution, resource availability, and user importance), provides a blueprint for mapping out near-term and long-range technology development and demonstration objectives as a function of varying task-complexity levels.

## Approach

In order to develop the prioritized suite of tasks, we first wrote a strawman report that included a preliminary list of tasks extracted from NASA documents, description of the multi-attribute decision-analysis method for ranking these tasks, tentative data (attribute weights and utility values) for computing the ranking, and preliminary task-ranking results. This report was distributed to several telerobot NASA Centers and contractors for review. We then visited the following NASA Centers and contractors:

NASA Goddard Space Flight Center
NASA Langley Research Center
General Electric RCA, Advanced Technology Laboratories
Massachusetts Institute of Technology (MIT), Aerospace Engineering Dept.
NASA Marshall Space Flight Center
NASA Johnson Space Center
Jet Propulsion Laboratory
NASA Ames Research Center

At each Center the straw-man report was presented, the findings were discussed and feedback was received from the Center's professional staff. The feedback from all of the Centers was used to adjust the results to reflect their respective resident experience.

One of the first steps in performing the task assessment was to compile a straw-man list of potential tasks. It was understood at the outset of this study that the deadline for completion of the assessment constrained the scope of the study. It was therefore decided to first develop broad potential categories of tasks, and then list tasks within each category. The major task categories, derived from existing NASA documents, were as follows:

1. Assembly - The process of physically connecting mechanical or electrical components.
2. Servicing - Various processes involving the removal and replacement, adjustment, refurbishment, or reconfiguration of spacecraft components.
3. Inspection - The process of using telerobotic sensing to determine the integrity of a spacecraft component.
4. Material Handling - The process of actively transporting and replacing supplies for performing assembly or servicing functions (e.g., pick-and-place supplies from the shuttle to an assembly site).
5. Manufacturing - The process of converting raw materials into finished products in the Space Station.

Tasks were selected drawing largely from documents describing the Robotic Assessment Test Sets (RATS)[1] and the Polar Platform Payload Servicing requirements [2]. We also examined Space-Station tasks and manifesting studies performed by the various work-package subcontractors [3,4,5]. One question that arose was the degree to which the suite of straw-man tasks comprehensively represented the total array of near-term tasks related to the Space Station. We examined two larger, more comprehensive task studies, the MIT ARAMIS and McDonnell Douglas THURIS studies [6,7], and compared them against our categories and the straw-man list. We found that, although they are more comprehensive in defining different tasks elements, both the ARAMIS and THURIS task listings could be represented as components of the straw-man suite of tasks we selected as demonstration candidates. Task comprehensiveness was also affected by inputs from the different NASA Centers we visited. During each session in which we discussed the straw-man task-ranking analysis with the respective Center's staff, we collected additional potential tasks. These tasks were then examined to determine whether they were different from the list already compiled, or whether they could be considered a subset of another task already on the list.

Table 1 lists 23 tasks considered unique from the standpoint of representing different size and scale of objects, different types of manipulation, and different kinds of technologies. Among the 23 tasks in Table 1, only the first 18 tasks were ranked because the details of the remaining ones were insufficient. Having selected this set of candidate tasks, the next step was to rank them in the order of their utility ("worth" or suitability) as domains for demonstration of telerobotic techniques. These techniques may be applicable to future NASA missions, in particular to those associated with the Space Station. A variety of factors were listed, called attributes, that have a direct effect on the task ranking. We then used the Multi-Attribute Decision Analysis (MADA)[8] method to rank the candidate tasks on the basis of the effect of each attribute on the suitability of each task as a domain for telerobotics R&D. MADA provides a decision structure with which to rank several task options based on an aggregate value of the net worth measured across several attributes. The following discussion defines the MADA technique and explains how it was applied to the task ranking.

## Capabilities of Multi-Attribute Decision Analysis (MADA)

Task ranking based on a set of different attributes may pose some problems. The original guidelines under which the task assessment was initiated indicated that we had to compile a suite of tasks that was appealing to the user community and that could be developed and demonstrated in the laboratory within limited resources. These guidelines may lead to conflicting results, because tasks that possess attributes important to NASA's user community may require more laboratory and professional resources than the NASA Centers have. Further, some attributes, such as "Importance to User" or "Technological Contributions," are more easily measured qualitatively than quantitatively.

Multi-attribute decision analysis facilitates decision-making in the presence of conflicting attributes, measured either qualitatively or quantitatively. MADA is based on the premise that individuals knowledgeable in the area of interest (in this case, task demonstration designers) can express their preferences among different options if provided with an appropriate decision structure that enables them to make comparisons and quantify their responses.

The quantification of their responses is in the form of a so-called utility value - a metric that measures the net worth, or effectiveness, of a given option to the group of knowledgeable individuals. The quantification process draws on well-developed and tested methods of decision analysis [8,9,10].

Selection of Attributes

To be able to apply MADA to the task ranking, the following seven attributes were selected:

- Cost - The approximate design, development, test, and evaluation (DDT&E) cost of the hardware and software required to demonstrate a task.
- Technology Demonstration/Development Schedule - The time required to acquire and/or develop the technologies, hardware, and software required for a task, design and integrate the system, and demonstrate the task to a specified level of robustness.
- Importance to User - The degree to which a task can be applied to real world problems, meets the requirements of the user community, utilizes a domain the user community feels is important, and, if done successfully, enables completion of other similar tasks.
- Productivity/Safety Impact - The potential for a telerobotic task to increase the productivity of an astronaut and reduce his/her adverse risk. Aspects of productivity include reduction in EVA time and the amount of time an astronaut or ground crew is freed to do other tasks. Safety is primarily related to reduction in hazard exposure, whose two main factors are the hazard severity and exposure time.
- Center Resources - The degree to which the hardware and software and the technical personnel required by a task are available in any of the various NASA Centers.
- Technological Contributions - Contributions to advancing the state of the art of the technology elements required to perform a task.
- Possible Near-Term Demonstration Success - The estimated confidence that a laboratory demonstration of a given task will succeed.

TABLE 1: List of Candidate Telerobotic Tasks

ASSEMBLY TASKS

1. Truss Assembly
2. Utility Tray Deployment and Pop-Up Connector Utility Line Installation
3. Station Interface Adapter (SIA) to Truss Connection
4. Payload Interface Adapter (PIA) to Station Interface Adapter (SIA) Connection
5. Solar Dynamic Array (SDA) Radiator Assembly and Deployment

SERVICING TASKS

6. Solar Power Converter Orbital Replacement Unit (ORU) Changeout
7. High Resolution Solar Observatory (HRSO) Film Canister Changeout
8. Hubble Space Telescope (HST) Axial Instrument Changeout
9. Hubble Space Telescope (HST) Reaction Wheel Assembly (RWA) Changeout
10. Gamma Ray Observatory (GRO) Refueling
11. Earth Observatory System (EOS) Instrument/Orbital Replacement Unit (ORU) Changeout
12. Solar Maximum Mission (SMM) Main Electronics Box (MEB) Replacement
13. Earth Observatory System (EOS) Instrument Reconfiguration
14. Earth Observatory System (EOS) Instrument Recalibration/Adjustment
15. Extra Vehicular Activity (EVA) Retriever
16. Telerobotic Docking

INSPECTION TASKS

17. Electrical Connector Removal/Inspection
18. Clean/Inspect Surface (Solar Cell Cleaning)

TASKS CONSIDERED UNIQUE BUT NOT RANKED BECAUSE OF INSUFFICIENT INFORMATION

19. Position/Push RCS Thruster
20. Alpha Joint Position/Installation
21. Antenna Position/Installation
22. Repair Manipulator Arm on Platform
23. Specific Failure Recovery Schemes

The importance of each of the above seven attributes relative to each other depends on the main drivers behind the development of telerobotic techniques and task demonstrations. We represent the relative importance of each attribute by an attribute weighting factor, called attribute weight, whose value varies between 0 and 1. During our visits to the various Centers we were urged to consider two different sets of attributes, user attributes and research attributes. As a result, we established the two attribute lists shown in Table 2.

TABLE 2: Attributes Important to User and Research Communities

| User Attributes | Research Attributes |
|---|---|
| a. Cost | a. Cost |
| b. Technology/Demo Development Schedule | b. Technology Demo Development Schedule |
| c. Importance to User | c. Importance to User |
| d. Productivity/Safety Impact | d. Center Resources |
| | e. Technological Contribution |
| | f. Possible Near-Term Demo Success |

Our visits made clear the need for a new attribute called "Demo Fidelity" or "Demo Realism." Changes in dynamics due to scale, inertial characteristics, lighting, or weightlessness definitely alter the usefulness of the results. This attribute was incorporated into the attribute "Importance to User." Therefore, the more closely a laboratory demonstration approaches the real on-orbit task, the higher its utility to the user.

The agreed-upon approach for handling the two sets of attributes was realized by applying different weights to the attributes, depending on whether the task ranking was being done for the user community or the research community. Those tasks that ranked equally high for both communities would then make up the desired task suite. The other major distinguishing factor for segregating the desired task suite from the rest of the tasks was whether the tasks that had high ranking in the research community had application to the high-ranking (but different) tasks in the user community. This would mean that, indeed, the final suite of tasks had importance to the user community as a whole.

## Utility Values

Using MADA, once the attributes and their relative importance weights are established, the next step is to develop measures by which to determine the utility (the net worth) of a given candidate task in relation to each attribute. The measure of worth of a given task candidate is the utility value of that particular task. For example, the cost attribute, measured in 1988 dollars, reflects the rough cost of constructing the required hardware/software testbed in the laboratory to perform a given task. Tasks that could be done within the budget constraint have high utility values for that attribute; conversely, tasks that exceeded the budget constraint have low utility values. Similarly, different utility values correspond to each task for each of the other attributes. The measures used to estimate each of these utility values are described in Reference [11]. It is important to recognize that derivation of actual utility measures, using empirical data, provides the most accurate ranking outcome. The data presented in Reference [11] reflects an attempt to use as much empirical data as possible.

Classically, MADA requires that utility values range between 0 and 1. To facilitate the assignment of utility values, we classified the utility of each attribute relative to any task into three levels: Low utility (0.0 to 0.3), medium utility (0.3 to 0.6), and high utility (0.6 to 1.0). These ranges are shown in Table 3.

TABLE 3: Utility Value Ranges

| Attributes | Low Utility (0.0-0.3) | Medium Utility (0.3-0.6) | High Utility (0.6-1.0) |
|---|---|---|---|
| Cost | High Cost | Medium Cost | Low Cost |
| Technology/Demo Development Sch. | Far Term | Medium Term | Near Term |
| Importance to User | Low | Medium | High |
| Productivity/Safety Impact | Low | Medium | High |
| Center Resources | Absent | In Process | Existing |
| Technological Contributions | Low | Medium | High |
| Possible Near-Term Demo Success | Low | Medium | High |

Detailed utility values were calculated by using the measures defined above. Applying the ranges indicated in Table 3 allowed Center participants to understand what was meant by a task having low, medium, or high utility value. During the Center visits, we received feedback regarding the attribute weights, the utility values, and the ranking results.

Utility Functions

Following the assignment of the attribute weights and the utility values, the final step in the MADA process uses these numbers to calculate the value of an overall task utility of each task candidate. Once the calculations are completed, the candidate tasks can be ranked in the order of highest to lowest overall task utility.

The overall task utility is computed by means of a utility function - a function of the attribute weights and the utility values. There are two forms of the MADA utility function: the additive form and the multiplicative form.

Although the additive form, or weighted sum, is more intuitive in its design, it is restricted to being applied only when the various attributes (as measured by the utility values) are independent of one another, i.e., the utility value obtained for one attribute should not change if the utility values of other attributes are adjusted. This condition can be difficult to meet because attribute utility independence is rare in the real world. However, if the attribute weights are not normalized and the above condition is not met, then use of the additive form can result in an incorrect ranking.

The attribute utility independence condition discussed above is the major complication in using the additive form. To resolve the potential problem of ranking errors, the multiplicative form of MADA was developed. The multiplicative form, although more complex than the additive form, is more rigorous because it does not require utility independence (i.e., changes in the utility values for one attribute can be traded off pairwise against the utility value of another attribute). Reference [8] provides a detailed derivation of the multiplicative form of the utility function, which is as follows:

$$U(x) = \frac{1}{K} \{\Pi_{i=1}^{n} [1 + Kk_i u_i(x)] - 1\} \ , \qquad (1)$$

where $U(x)$, $k_i$, $n$, and $u_i(x)$ are defined above and K is a master scaling constant, which is inserted into the function to ensure that $U(x)$ falls between 0 and 1, as required by the definition of a utility value. The value of K is derived by setting $U(x) = u(x) = 1$ in the above equation and numerically solving the following nth order polynomial for K:

$$1 + K = \Pi_{i=1}^{n}(1 + Kk_i) \ . \qquad (2)$$

Among the different real values of K, the single value satisfying $-1<K<0$ should be chosen [8]. Once K is calculated, U(x) can be determined discretely for each task option through the multiplicative combination of all the attribute utility values for a given task.

Task Complexity

The scope of this study precluded the generation of detailed specifications for each task. A wide disparity was found in the levels of detail to which the tasks were described in the literature. In addition, different levels of telerobotic technology advancement were required by different tasks. For these reasons, we introduced the notion of task-complexity levels.

We analyzed each task from the point of view of demonstrating, in the laboratory, the technologies necessary to perform that task. Rather than selecting a single task demonstration scenario, we postulated several scenarios, at increasing levels of complexity, for each task: Level A was the most complex, Level B the second most complex, Level C the third most complex, and Level D the least complex. For each task, we specified scenarios for these three or four complexity levels. We set the levels so that there would be a high confidence of success if task scenarios of low complexity level (Level C or D) were to be demonstrated today, while those of the highest complexity level (Level A) would have a low confidence of success (the Level A demonstration represents the task as it would be performed in the real application environment). See Reference [11] for specific examples of different levels of task complexity.

We considered only task complexity and made no assumptions about how a given task-complexity level should be implemented (i.e., by teleoperation or automatically). Additionally, we made no attempt to correlate the complexity levels across different tasks; for example, Level C of one task could be more complex than Level B of another.

Task breakdowns and task rankings performed in this study were done assuming Level-A complexity for all tasks. The task complexity levels are useful because they provide (1) a progression of increasingly complex demonstrations as a means for measuring R&D progress toward the ultimate (Level A) objective, and (2) a fallback implementation; if a given complexity level cannot be achieved within budget and schedule constraints, a lower level may be attainable. Each set of task levels, therefore, provides only a progressive set of objectives.

Task Breakdowns

Some of the attributes used in the task ranking depend on the technology required to perform a given task [12,13]. Progressive, hierarchical task breakdowns to the task-primitive level are intended to provide the means for identifying the underlying task technologies. The low-level breakdowns can be viewed as generating pseudo-code subroutines for performing tasks. It is at the primitive level of the task breakdowns that the required technologies become apparent. Several studies have drawn upon task-breakdown analysis for deriving the required technology elements [12,13,14,15,16].

Task breakdowns are used only to define the required task technologies; they are not intended to specify the approach for implementing a task demonstration, although they may serve as a good starting point. The breakdowns have been done from a telerobotic perspective because the demonstrations are intended to be performed by a telerobotic system. Care has been taken to ensure that the telerobotic actions are generic enough to be accomplished autonomously, by teleoperation, or by a mixture of both. For instance, a common function is to determine the location of a part. This function could be performed automatically by a vision system; it could also be performed in a mixed mode by having the system display a processed image of the scene, which helps the teleoperator determine the part's location.

Ideally, a task breakdown would be performed for each of the tasks considered (see Table 1). However, to make the most of the limited time of this study, we decided to break down only three tasks, one from each of the three task categories (assembly, servicing, and inspection). We assume that tasks within a given category will require similar technologies so that it should be possible to find a representative task. The following tasks were selected for breakdown:

|  |  |
|---|---|
| •Assembly: | Truss Assembly |
| •Servicing: | Solar Maximum Mission MEB Changeout |
| •Inspection: | Solar-Cell Cleaning/Inspection |

A typical breakdown of one of these tasks is given in Reference [11].

Results

The multiplicative-form ranking method described earlier was applied to rank the first 18 tasks listed in Table 1 on the basis of the initial (straw-man) attribute weights and utility values. These initial weights and values were later modified based on inputs from NASA Center participants, and the ranking was subsequently recalculated.

Attribute Weights and Utility Values

In our discussions with the NASA Centers it became clear that the research community views the relative importance of the attributes differently than the user community (see Table 2). For this reason we performed the task ranking for two sets of attribute weights, one representing the research community, and the other representing the user community. Table 4 presents the attribute weights for the two cases. Feedback from the Centers also indicated that some attributes are not significant in ranking the tasks for one community or the other; in this case, these attributes are not used in the ranking and do not have a weight (see Table 4).

TABLE 4: Attribute Weights ($k_i$)

| Attributes | User Community | Research Community |
|---|---|---|
| Cost | 0.5 | 0.9 |
| Technology/Demo Development Schedule | 0.75 | 0.7 |
| Importance to User | 0.95 | 0.6 |
| Productivity/Safety Impact | 0.95 | - |
| Center Resources | - | 0.7 |
| Technological Contributions | - | 0.95 |
| Possible Near-Term Demo Success | - | 0.7 |

The utility values for each task, assuming Level-A complexity, are listed in Table 5.

TABLE 5: Task Utility Values $u_i(x)$

| | Attributes | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Cost | | Tech./Demo Development Schedule | Importance to User | Productivity/ Safety Impact | Center Resources | Technological Contribution | Possible Near-Term Demo Success | |
| Tasks | Tel. | Aut. | | | | | | Tel. | Aut. |
| Truss Assembly | 0.9 | 0.5 | 0.5 | 0.8 | 1.0 | 0.8 | 0.8 | 0.7 | 0.4 |
| Utility Tray Deployment ... | 1.0 | 0.9 | 0.7 | 0.7 | 0.4 | 0.6 | 0.4 | 0.7 | 0.7 |
| SIA to Truss Connection | 0.8 | 0.0 | 0.4 | 0.8 | 0.3 | 0.2 | 0.6 | 0.4 | 0.1 |
| PIA to SIA Connection | 1.0 | 1.0 | 0.9 | 0.4 | 0.2 | 0.9 | 0.3 | 0.9 | 0.8 |
| SDA Radiator Assembly & Deployment | 0.8 | 0.0 | 0.1 | 0.7 | 0.4 | 0.2 | 0.8 | 0.3 | 0.1 |
| Solar Power Converter ORU Changeout | 1.0 | 1.0 | 0.9 | 0.4 | 0.2 | 1.0 | 0.2 | 1.0 | 0.9 |
| HRSO Film Canister Changeout | 1.0 | 0.8 | 0.4 | 0.7 | 0.4 | 0.5 | 0.6 | 0.5 | 0.3 |
| HST Axial Instrument Changeout | 0.9 | 0.6 | 0.4 | 0.8 | 0.5 | 0.3 | 0.8 | 0.5 | 0.1 |
| HST RWA Changeout | 1.0 | 0.8 | 0.4 | 0.6 | 0.2 | 0.5 | 0.5 | 0.5 | 0.3 |
| GRO Refueling | 0.9 | 0.6 | 0.4 | 0.7 | 0.9 | 0.6 | 0.7 | 0.7 | 0.2 |
| EOS Instrument/ORU Changeout | 0.9 | 0.9 | 0.8 | 0.7 | 0.6 | 0.8 | 0.5 | 0.9 | 0.7 |
| SMM MEB Replacement | 1.0 | 0.4 | 0.6 | 0.9 | 0.4 | 0.7 | 0.9 | 0.7 | 0.2 |
| EOS Instrument Reconfiguration | 0.8 | 0.3 | 0.3 | 0.7 | 0.6 | 0.4 | 0.8 | 0.3 | 0.1 |
| EOS Instrument Recalibration/Adjustment | 1.0 | 0.8 | 0.5 | 0.6 | 0.6 | 0.7 | 0.4 | 0.8 | 0.7 |
| EVA Retriever | 0.8 | 0.0 | 0.6 | 0.8 | 0.9 | 0.7 | 1.0 | 0.7 | 0.0 |
| Telerobotic Docking | 0.6 | 0.0 | 0.3 | 0.7 | 0.7 | 0.6 | 0.7 | 0.7 | 0.0 |
| Electrical Connector Removal/Inspection | 1.0 | 0.6 | 0.2 | 0.7 | 0.5 | 0.5 | 0.8 | 0.4 | 0.1 |
| Clean/Inspect Surface (Solar-Cell Cleaning) | 0.9 | 0.4 | 0.1 | 0.7 | 0.6 | 0.3 | 0.8 | 0.6 | 0.1 |

Top Ranked Task Candidates

One of our primary objectives was to compile a suite of tasks that both researchers and users would find useful. We noted that five tasks appeared among the top eight tasks in the rankings, regardless of whether the ranking was performed for the user or research community, and regardless of whether the task was to be mostly automated or mostly teleoperated. All of these tasks were basically equal and are discussed briefly as follows:

- The EVA Retriever task has an extremely high technical contribution rating because it requires the combination of challenging levels of most technologies, including manipulation, mobility, sensing and perception, reasoning, and communication. This task also has a high rating for the attribute "Productivity and Safety."
- The SMM MEB Replacement task also requires multiple technologies, and has a high "Importance to User" rating because the actions required to perform the task are highly generalizable to other tasks.
- The Truss Assembly task currently would require the highest amount of astronaut EVA time, so it has the highest "Productivity and Safety" rating; in addition, it requires significant perception, reasoning, and manipulation technologies, yielding a high "Technical Contribution" score; and since some centers are already working on this task, the "Center Resources" rating is high.
- The EOS Instrument/ORU Changeout task is relatively inexpensive, requiring only a medium-size mock-up and a single arm; it has a high "Importance to User" rating because there are about 50 different kinds of instruments on the EOS; in addition, it receives a high "Center Resources" score.
- The HST Axial Instrument Changeout task is included because of its important technological contributions to the handling of large and massive objects and the use of flexible arms.

The subsequent technologies required to implement the tasks are shown in Reference [11]. Note that two utility values associated with the attributes "Cost" and "Possible Near-Term Demo Success" are given for each task, one (marked by "Tel") assuming that the task will be controlled mostly by teleoperation and the other (marked by "Aut") assuming that the task will be mostly automated. The rationale for the selection of the utility values in Table 5 is also given in Reference [11].

## Decision Framework for Determining Task Objectives

The results of the task ranking can help develop a program plan for both the near-term and far-term research and demonstration objectives. While carrying the task breakdown analysis down to the task-primitive action level, we became aware that the selection of tasks is highly dependent on technology availability. It became clear that the task demonstration selected by a research Center should support that Center's technology research goals. Taken one step further, selecting technology research goals within an application environment implies meeting cost, resource, and schedule constraints. Therefore, the decisions about what to pursue in terms of a task demonstration objective require an iterative process of selecting a task, comparing it with given technology objectives, and verifying that the schedule and resources limits are not exceeded. The decision framework shown in Figure 1 illustrates the process of using the task-ranking results, task complexity, and supporting task-utility data to formulate program plan objectives. The JPL $3M limit was used as the budget ceiling for example purposes. The 1-2 year demonstration cycle is the present preferred time-frame for exhibiting technologies because it is essential to show "progress" as a means of substantiating the associated yearly funding support.

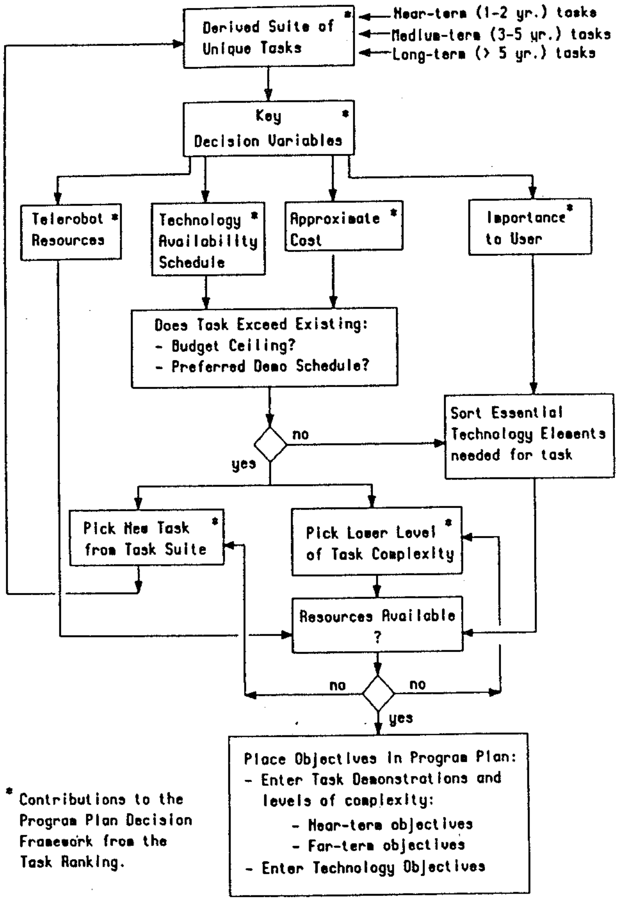

Derived Suite of Unique Tasks *
Near-term (1-2 yr.) tasks
Medium-term (3-5 yr.) tasks
Long-term (> 5 yr.) tasks
Key Decision Variables *
Telerobot Resources *
Technology Availability Schedule *
Approximate Cost *
Importance to User *
Does Task Exceed Existing:
- Budget Ceiling?
- Preferred Demo Schedule?
no
yes
Sort Essential Technology Elements needed for task
Pick New Task from Task Suite *
Pick Lower Level of Task Complexity *
Resources Available ?
no
no
yes
Place Objectives in Program Plan:
- Enter Task Demonstrations and levels of complexity:
- Near-term objectives
- Far-term objectives
- Enter Technology Objectives
* Contributions to the Program Plan Decision Framework from the Task Ranking.

Figure 1: Decision Framework for Establishing Objectives

The decision framework starts with the suite of tasks composed of the highest ranked tasks. The key tradeoff variables are available resources, technology availability and schedule, cost, and importance to user. A task objective is selected from the suite of tasks and evaluated initially as a function of the cost ceiling and as to whether the technology and task domain can be successfully demonstrated in the near term. The next step is to pick the key technology elements essential to the user community and determine whether the existing testbed and workforce resources can complete and demonstrate the technology and the task. If the cost ceiling and schedule constraints are exceeded, then either a new task that is lower on the ranking list or a lower level complexity of the same task is examined. The process is repeated until task objectives that reasonably meet the programmatic constraints have been established. This process can be applied to setting both near-term and far-term task objectives.

## Conclusions

Based on the preceding analysis, it would seem feasible to give priority to the development and demonstration of the five tasks listed in Table 6, each at the complexity level outlined in that table.

TABLE 6: Recommended Tasks and Their Complexity Levels

| Task | Complexity Level |
|---|---|
| EVA Retriever | Level C |
| SMM MEB Replacement | Level C-B |
| Truss Assembly | Level C-B |
| EOS Instrument/ORU Changeout | Level A |
| HST Axial Instrument Changeout | Level C |

- EVA Retriever: Obstacles, moving targets, natural lighting, and testing the system in a 3-D environment require a significant amount of software and hardware development, which is probably an unrealistic goal for a two-year time frame. Hence, we recommend a demonstration task at Level C complexity.
- SMM MEB: Portions of Levels B and C have been demonstrated for this task. However, manipulation of flexible materials, such as thermal blankets and cables, will require significant development time. For this reason a Level C-B demonstration is a reasonable goal in the near term.
- Truss Assembly: This task has been demonstrated at Level D in the laboratory, both under teleoperation and automatically. A similar truss assembly task has been demonstrated at Level B purely teleoperation control. We therefore recommend a Level C-B demonstration in the laboratory with the emphasis on automatic operation.
- EOS Instrument/ORU Changeout: This task has been demonstrated in the laboratory at a Level B complexity. Several Centers have mock-ups of EOS or EOS-type ORUs. Thus, there is little impact on cost and schedule from having to develop mock-up equipment. For these reasons we recommend a Level A demonstration in the near term.
- HST Axial Instrument Changeout: A Level C complexity is recommended for the following reasons: Significant time and money are required to develop the mock-ups needed for a full-scale demonstration. In addition, we anticipate that a significant amount of R&D is needed to handle the large payloads entailed in this task in constrained areas by means of large flexible arms.

## References

[1] Goddard Space Flight Center, "Robotic Assessment Test Sets: Levels 1,2,& 3," Technical Report SS-GSFC-0029, Goddard Space Flight Center, Greenbelt, Maryland.

[2] JPL, "Polar Platform Payload Servicing Requirements," Technical Report JPL D-3177, Rev. A (internal document), Jet Propulsion Laboratory, Pasadena, California, December 1986.

[3] McDonnell Douglas, "Space Station EVA Time Requirements," Briefing to Critical Evaluation Task Force (CETF), September 4, 1986.

[4] Grumman, "Space Station Assembly Study," Internal Study, March 1986.

[5] Rockwell International, "Space Station Assembly Sequence and Telerobotics," Internal Study, March 1986.

[6] R. Miller, M. Minsky, D. Smith, and D. Akin, "Space Applications of Automation, Robotics, and Machine Intelligence (ARAMIS)," Internal Study, August 1982.

[7] McDonnell Douglas, "The Human Role in Space (THURIS)," Internal Study, 1984.

[8] R.L. Keeney and H. Raiffa, Decisions with Multiple Objectives: Preferences and Value Trade-Offs, John Wiley, New York, 1976.

[9] Feinberg, A., et al., "Advanced Vehicle Preference Analyses," Technical Report, JPL D-941 (internal document), Pasadena, California, September 1984.

[10] W.F. Zimmerman, J. Bard, and A. Feinberg, "Space Station Man-Machine Automation Trade-Off Analysis," JPL Publication 85-13, Pasadena, California, February 15, 1985.

[11] W.F. Zimmerman, J. Meyers, and D. Ruth, "Task Ranking for the Telerobot Demonstration," SRI/JPL Project No. 3520, July 1988.

[12] SRI International, "NASA Space Station Automation: AI-Based Technology Review," Technical Report NASA TBD, SRI Int'l., Menlo Park, California 1985.

[13] NASA, "Advancing Automation and Robotics Technology for the Space Station and for the U.S. Economy," Technical Report NASA 87566, ATAC, Houston, Texas, March 1985.

[14] W. Zimmerman and N. Marzwell, "Space Station Level B Automation Technology Forecasting/Planning Structure," JPL D-2862 (internal document) April 30, 1985.

[15] D. Tesar, "Next Generation of Technology for Robotics," February 1985. University of Texas, Austin, Texas.

[16] D. Tesar, "An Assessment of the Development and Application Potential for Robots to Support Space Station Operations," September 1985, University of Texas, Austin, Texas.

[17] B. Boehm, Software Engineering Economics, Prentice-Hall, New Jersey, 1981.

# The Telerobot Workstation Testbed for the Shuttle Aft Flight Deck: A Project Plan for Integrating Human Factors into System Design

T. Sauerwein

NASA Goddard Space Flight Center

Abstract

This paper begins by describing the human factors design process in developing a shuttle orbiter aft flight deck workstation testbed. The design methodology is presented along with the results of the design and the problems and solutions regarding human factors design principles.

## 1.0 Introduction

### 1.1 Purpose of the Paper

The purpose of this paper is to describe how to direct a design effort focused on the human operator. In developing an operator workstation to control various laboratory telerobots, strong elements of human factors engineering and ergonomics are integrated into the design process. The integration of human factors is performed by incorporating user feedback at key stages in the project life-cycle. An operator centered design approach helps insure the system users are working with the system designer in the design and operation of the system. Through practical experiences at the Goddard Space Flight Center(GSFC) Robotics Laboratory, a project plan is being implemented which will aid in achieving an operator centered approach.

The described project plan represents an approach to incorporating human factors into the design process. Other designers of operator centered projects can follow the model described in this paper. Some elements in the project plan, which are directed toward telerobot workstations and the GSFC Robotics Lab, may not be applicable to other systems.

### 1.2 Background on the Project

One of the purposes of the Robotics Laboratory at the NASA Goddard Space Flight Center is to develop Flight Telerobotic Servicer(FTS) task scenarios. The task scenarios are laboratory versions of the tasks to be performed by FTS on Space Station Freedom(SSF), the Demonstration Test Flight(DTF-2) and to a lesser extent, the Development Test Flight(DTF-1). These scenarios are integrated, end-to-end, from the operator interface to robot worksite hardware. The scenario integration process is driven by various disciplines including human factors.

The FTS robot is controlled from both the National Space Transportation System(NSTS) and the SSF workstations. The Robotics Laboratory at Goddard provides FTS and DTF emulation using a pair of force-reflective robot arms. A gantry robot is used to emulate a transport system for the FTS and DTF-2. Functional mock-ups are constructed for both of the NSTS and SSF workstations. The NSTS mock-up workstation reflects the Aft Flight Deck(AFD) area.

### 1.3 Reasons for Integrating Human Factors

The human factors discipline needs to be incorporated into the development of the AFD workstation because of the significant human operator role in system performance. The human operator has capabilities and limitation. Furthermore there are constraints and limitations due to the NSTS and FTS systems. These constraints are due to the NSTS AFD and the FTS requirements and tasks. The AFD puts constraints on the design because of spatial limitations(2 Sq. Ft. control and display area), visual limitations(little or no direct viewing) and operational limitations(FTS operator must not interfere with other operators). The FTS mandates one person operation and tasks requiring dexterity from the operator. These constraints affect operator interface design significantly. To ensure that the system can be operated effectively and safely, an approach that addresses operator requirements as well as system requirements is necessary.

## 2.0 Project Plan to Integrate Human Factors

A project plan for integrating human factors into telerobot workstation design was initiated in the fall of 1988. The plan is now in the preliminary design phase.

### 2.1 The Design Goals

The AFD mock-up workstation at Goddard serves several purposes. First, the operator workstation interfaces to a number of different robot systems. The Goddard Robotics Laboratory's control system development approach is to implement a NASREM-compliant control system independent of the underlying hardware. The operator interface is designed to be a modular subsystem of the overall NASREM control architecture. The control system interfaces are isolated within the NASREM architecture. The workstation serves as the operator control point for a variety of different robotic systems in the laboratory and provides modularity and flexibility for future evolution.

Another purpose of the workstation testbed is to utilize existing operator interface technology and apply human factors principles and guidelines in a testbed facility. The AFD workstation development project offers an excellent opportunity to apply many of the human factors engineering results already developed under previous NSTS and SSF research and development efforts. To make the workstation testbed as similar to flight as possible, requirements and constraints from both FTS and NSTS Aft Flight Deck are applied.

Finally, the workstation testbed project is developing a human factors design methodology that ensures the requirements of the user are effectively met. The laboratory mock-up workstation is being developed at the same time and within the overall framework of the NASREM-compliant control system. Furthermore, the mock-up workstation requirements are driven by the constraints of the NSTS AFD workstation for the FTS. By emphasizing human factors, end user requirements are addressed throughout the development process. It is also recognized that user requirements and operator interface capabilities change and this change involves trade-offs based on hands-on evaluation by the users. Therefore, the development philosophy is to provide testability, adaptability and future evolution in the workstation testbed.

### 2.2 The Project Plan

A project plan is devised to effectively integrate human factors into the design process. The initial step of this plan is to organize a formal design team composed of engineers, computer systems developers, robot system operators and human factors specialists. Due to the extensive nature of human factors, at least one team member must be trained in human factors research.

Also, feedback from the user community must be incorporated for good human centered design. In order to coordinate feedback, contacts are identified and fostered with other teams working on related human factors projects or research.

After forming the design team, objectives and deliverables are developed. The objectives for the team reflect typical design procedures:

1. Task Analysis

2. Requirements Definition

3. Preliminary Design

4. Detailed Design

5. Fabrication

6. Test and Evaluation.

The workstation testbed project began with personnel at Goddard working without robot operators and human factors specialists. The team members were software designers and programmers with an interest in human factors. The goal was to develop a well-engineered telerobot workstation testbed that conforms to the basic principles of human factors design. It was soon determined that the task was too large and too complex for the design team. A specialist in human factors and ergonomics joined the team, and contacts were established with the design engineers, human factors experts and astronaut crew representatives engaged in FTS workstation development at the Johnson Space Center(JSC).

Throughout the entire design process, reviews were conducted. These reviews were conducted with the intention of receiving operational feedback as well as technical feedback on the design. The reviews could have been based on formal sign-off lists or consensus reached through discussion. Because the design team was small(10 people) it was not necessary to incorporate formal sign-off authority within Goddard. Instead a consensus of the entire design team was reached on each issue. In order to keep track of input from reviews external to GSFC, a more formal review requiring sign-off authority has been implemented.

The project plan deviated slightly from the typical procedure described above:

1. Task Analysis

2. Requirements Definition

3. Conceptual Design

4. Preliminary Design

5. Detailed Design

6. Fabrication

7. Test and Evaluation(Verification)

## 2.3 Task Analysis

The next step in the project plan is to analyze the tasks to be performed. These tasks are derived from project goals and objectives. The task analysis uses a standard breakdown and terminology to categorize the actions performed by the system. The task analysis must be understood before any attempt is made to determine the operator interface.

Before the workstation testbed project began, it was assumed that an overall task analysis of FTS functions had already been completed. In fact, a task analysis had not been completed for FTS. Thus, it was necessary for the team to analyze the proposed FTS tasks and currently defined laboratory robot tasks. This process involved the analysis of FTS, NSTS, SSF, NASREM and robotics laboratory documents. Even though experienced personnel were compiling the task analysis, completing this task analysis increased the requirements definition phase of the project considerably. An example of the format used for the present task analysis effort is shown in Table 1.

## 2.4 Requirements Definition

Once the task analysis is complete, the requirements for the system are assembled. For an operator centered system, these requirements address the needs of the operator in performing the tasks as well as system requirements. One approach to finding the operator's needs is interviewing operators of similar systems and incorporating their feedback.

The primary sources for the requirements definition of the workstation testbed were:

1. The System/Function/Task Analysis of FTS functions(see above)

2. The Phase C/D Source Requirements for the procurement of the FTS system

3. The constraints imposed by the NSTS

4. Interviews and documentation pertaining to the operations and plans of the Robotics Laboratory.

Robotics laboratory robot operators at GSFC were also interviewed to gain insight into operators' requirements. The interviews with Robotics Laboratory personnel proved to be a valuable part of the requirements definition process. Since many of the interviewees were users of laboratory robots, their ideas had a significant impact on driving the requirements toward operator centered design. The final Top-Level Requirements document identified over 100 mock-up workstation testbed requirements. An example of the format used for presenting these requirements is shown in Table 2.

## 2.5 Conceptual Design

The operator interface should be addressed in a conceptual design document. This document includes the physical layout of the operator displays and controls that are used to accomplish a task. The display and control capabilities are described, as well as the functionality of the operator interface components. The conceptual design prompts early feedback from the operator concerning the layout of the displays and controls. This is important because the physical layout of the interface affects design issues. By reviewing the operator interface first, a human centered design approach is achieved. The Conceptual Design document describes: 1) the basic operator interface design philosophy, 2) the overall system configuration, 3) the approach proposed for displays and controls, 4) the principles suggested to integrate the displays and controls and 5) the necessary internal and external interfaces. It is generally easier to obtain feedback when a concrete design can be referenced.

This Conceptual Design document was the mechanism used to obtain early feedback from robot operators, human-factors researchers, astronaut crew representatives and other groups involved in workstation design. In fact, the feedback received from the Conceptual Design document was instrumental in forming a more flexible design approach for the AFD workstation testbed. A sample of two possible workstation layout proposals can be found in Figure 1.

## TABLE 1

## CONDENSED SYSTEM/FUNCTION/TASK ANALYSIS OF FTS LABORATORY SIMULATIONS

| | NASREM LEVEL | NASREM DESCRIPTOR | SYSTEM/FUNCTION/TASK DESCRIPTOR | |
|---|---|---|---|---|
| I. | | (Purpose) | 8.0 | Goals: |
| | | | 8.1 | Multi-purpose robot system |
| II. | | (Implementation) | 7.0 | Flights Manifest |
| | | | 7.1 | A DTF-1 |
| III. | Level 6 | Mission | 6.0 | Functions: |
| | | | 6.3 | Service (Maintain) |
| IV. | Level 5 | Planning/Scheduling | 5.0 | Scenarios: |
| | | | 5.7 | Replace a module |
| V. | Level 4 | Object/Task | 4.0 | Tasks: |
| | | | 4.5 | Sub-ORU changeout |
| VI. | Level 3 | E-Move | 3.0 | Subtasks |
| | | | 3.26 | Seat |
| VII. | Level 2 | Primitive | 2.0 | Actions |
| | | | 2.1 | Teleoperate |
| VIII. | Level 1 | Servo | 1.0 | Elements |
| | | | 1.2 | Joint positions |

## TABLE 2
## FORMAT FOR AFD WORKSTATION REQUIREMENTS DEFINITION

1.0 Workspace Requirements

- 1.1 Location and Enclosure
- 1.2 Consoles
- 1.3 Anthropometry and Ergonomics
- 1.4 Environment

2.0 Display Requirements

- 2.1 General Display Information
- 2.2 Visual Television Displays (CCTV)
- 2.3 Visual Computer Displays (Monitors)
- 2.4 Auditory Displays
- 2.5 Tactile Displays

3.0 Control Requirements

- 3.1 General Control Capabilities
- 3.2 Hand Controllers
- 3.3 Camera Controls
- 3.4 Lighting Controls
- 3.5 Modes of Control

4.0 Display/Control Integration (Architecture) Requirements

5.0 Communications Requirements

- 5.1 Operator Communications
- 5.2 Equipment Communications

6.0 Labelling Requirements

7.0 Restraint System Requirements

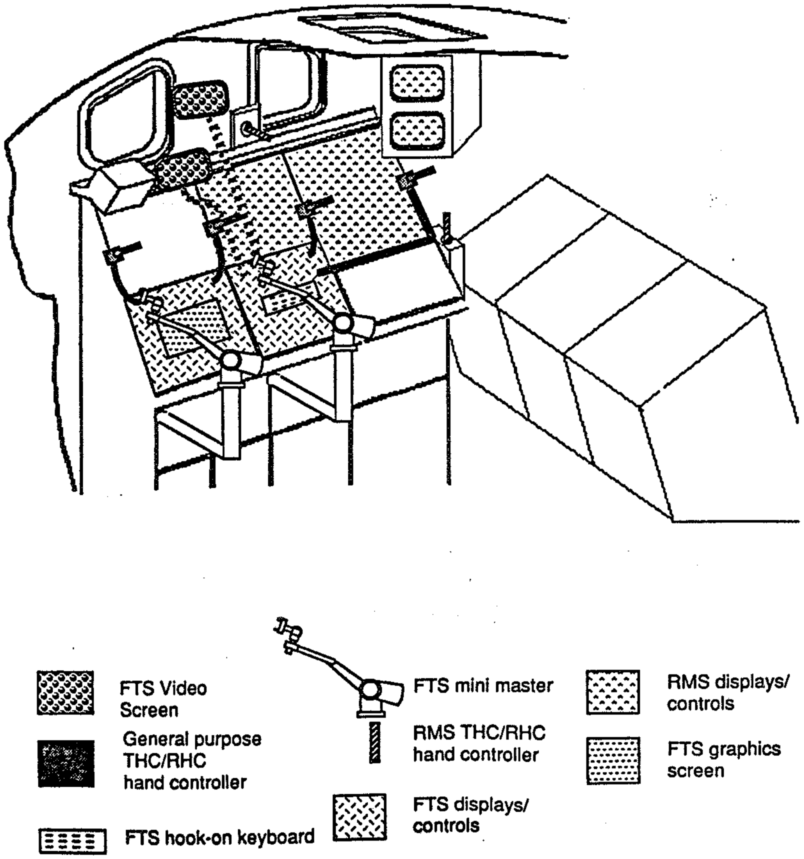
8.0 Safety Requirements

9.0 Maintenance Requirements

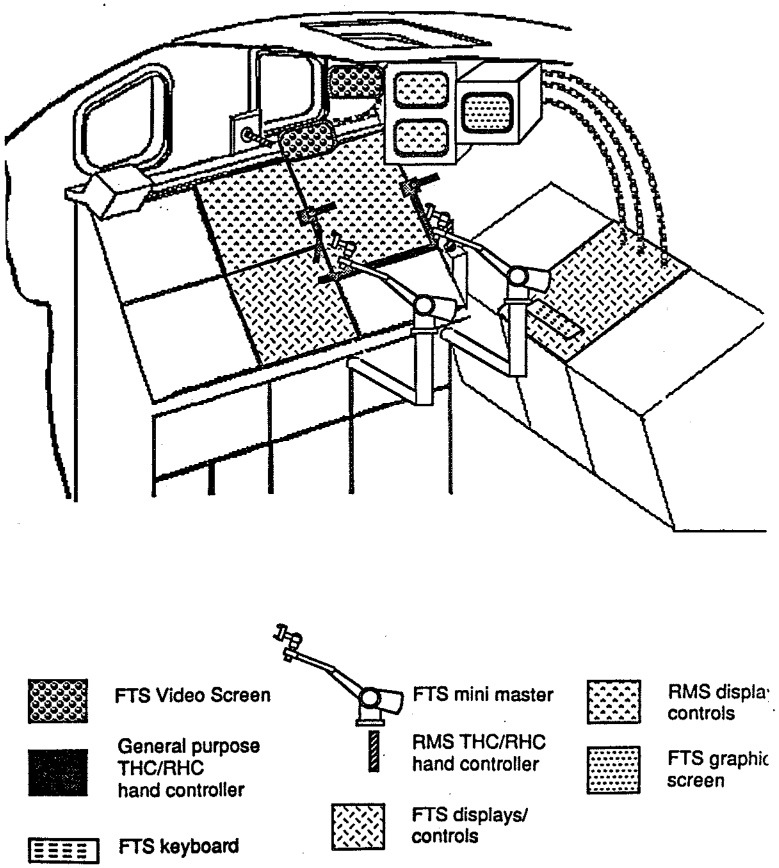
10.0 Electrical Requirements

11.0 Thermal Requirements

## DAP



## Combined Version 1



## 2.6 Prototyping

Conceptual design also involves constructing a mock-up of the operator interface. This is a full-scale non-operational form-and-fit mock-up with all applicable console spaces represented in approximately correct sizes and locations. The mock-up is useful in evaluating different configurations and high-level operator interface issues(such as display location) by testing participants using a physical model. This mock-up is modifiable to incorporate necessary design changes.

The AFD workstation testbed prototype has the capability to evaluate different numbers of operators sharing different functions at different workstation panels. Workstation project personnel, senior engineers, managers and novice research participants are tested to obtain preliminary data on ease of operator functioning.

## 2.7 Preliminary Design

The Preliminary Design for the workstation testbed is a complete design at the system and subsystem level. Preliminary console layout drawings showing approximate dimension of equipment are prepared and feedback concerning technological and operational feasibility is encouraged. Operator feedback at this stage of design is used to critique the preliminary operator execution sequences. These sequences or scripts are "walked through" by users of similar systems, and checks are made for conflicts concerning operation. The preliminary system configuration must satisfy the equipment, personnel, software and procedure specifications laid out or implied in the Top-Level Requirements document. Also, the system design must incorporate the feedback from the Conceptual Design document.

## 2.8 Detailed Design

The next stage in the project plan is the detailed design. The detailed design deals with the specific equipment rather than systems and subsystems. One of the most important human factors functions that is performed in this phase is the checking of display and control devices. Once displays, controls and configuration have been specified, detailed layouts are evaluated for compliance with human factors criteria(i.e. NASA-STD-3000, Man-Systems Integration Standards, etc.). At this stage, the important characteristics to evaluate include size, color, number of controls and displays, and control-display arrangement. These evaluations are performed on detailed drawings of the operator interface.

From these evaluation, a list of preliminary Human Engineering Deficiencies(HED) is generated. The HED's document the instances where human engineering design deficiencies exist. They also document the possible implications of each deficiency in terms of operator error, delay or dissatisfaction. Through operational and technical reviews, a decision is made to correct, ignore or compensate for the deficiency.

## 2.9 Design Verification

The next stage of the development process is operational testing and evaluation. From the human factors perspective, these tests determine if the assembled system meets human engineering criteria and is compatible with overall system requirements. Such testing and evaluation provide initial quantitative measurements of operator as well as system performance.

The Test and Evaluation portion of the project plan is composed of two types of activities:

1. Final checking of Human Engineering Deficiencies(HED)

## 2.10 Tests and demonstrations of operator/machine performance.

The final checking is made on operating components. Checking the workstation to determine final conformity is far more thorough and precise. All HED's and their predicted consequences are documented. A decision is made to correct the deficiency or to leave it and note any implications for training and operations.

An AFD workstation test plan is prepared in which test objectives will be identified and the proposed test methods will be described. Different populations are sampled to study various aspects of the user interface. The results of these experiments, tests and demonstrations are described in an operational test report. This report covers the test background, objectives, methods, controls, participants, prior training, apparatus, data collection, data reduction, data analysis and conclusions. The operational test report addresses quantitative results concerning how well operators performed the tasks as well as qualitative feedback from the operators about the workstation(i.e. ease of operation). This is the final stage were human factors feedback can be incorporated.

## 2.11 Documentation and Training

The final phase of the project plan insures effective workstation performance after final testing and acceptance. Procedures are developed to use the workstation. Proper operator's manuals are provided, and training implications are identified for task scenario evaluation. The task analysis, supplemented by test and evaluation results, serves as the basis for procedure development and training definition. Human engineering principles are applied to insure that the human functions and tasks are organized and sequenced for efficiency, safety and reliability of operation. Adequate operational, training and technical publications exist to properly support the workstation.

## 3.0 Summary of Results

The experiences of developing and applying this project plan have revealed important results. Some of these results are stated as general principles to guide thoughts and activities toward producing a more user-oriented design. Others are expressed as concrete steps and self-checks that can be applied to insure that the design effort stays oriented toward the user.

## 3.1 General Principles

The following is a list of some of the general principles that were the result of this experience:

1. Basic human factors guidelines should first be compiled
The research literature and application examples on basic human factors is extensive. The workstation testbed project needed a broad set of guidelines or principles compiled from the literature to serve as a practical base for the design.

2. Human factors should as much as possible drive the design process
User issues were addressed as much as possible early in the system design process. A set of requirements was derived based on human factors design principles as applied to the NASA robotics environment. These requirements then formed the basis for later design activities.

3. The user should be involved throughout the development process
Conducting interviews with potential users was very helpful, especially for setting the proper direction early in the project. It was also important to include the actual and potential users in the review processes.

4. The operator interface should be designed for flexibility

The workstation testbed would be reconfigured as basic ideas and basic system capabilities were added. Therefore, it became important to design in this flexibility and to build a modular, reconfigurable system.

## 3.2 Concrete Steps

1. Acquire a formal task analysis before beginning

2. Organize a team that has at least one human factors expert

3. Hold reviews that include users

4. Develop a conceptual design of the operator interface early in the design process

5. Build a physical prototype to resolve operator interface issues

6. Evaluate operator interface and performance soon after final integration.

## 4.0 Conclusion

The purpose of this paper is to describe how to successfully direct a system design effort that is focused on the human operator. A project plan designed to insure the proper integration of human factors into the design process is described. The project plan makes use of operator feedback as the mechanism for human factors integration. The user feedback received thus far in the development of the workstation testbed indicates that this feedback is helping to create a good design. The final test will come when an astronaut uses the AFD mock-up workstation to operate the FTS robot emulator.

# MULTI-LEVEL MANUAL AND AUTONOMOUS CONTROL SUPERPOSITION FOR INTELLIGENT TELEROBOT

S. Hirai and T. Sato

Autonomous System Section, Intelligent Systems Division
Electrotechnical Laboratory, MITI
1-1-4 Umezono, Tsukuba-shi, Ibaraki 305 Japan

## Abstract

Space telerobots are recognized to require cooperation with human operators in various ways. Considering the issue, this paper describes multi-level manual and autonomous control superposition in telerobot task execution. To realize the concept, we propose the object model, the structured master-slave manipulation system and the motion understanding system. The object model offers interfaces for task level and object level human intervention. The structured master-slave manipulation system offers interfaces for motion level human intervention. The motion understanding system maintains the consistency of the knowledge through all the levels which supports the robot autonomy while accepting the human intervention. The superposing execution of the teleoperational task at multi-levels realizes intuitive and robust task execution for wide variety of objects and in changeful environment. The performance of several examples of operating chemical apparatuses is shown.

## 1. Introduction

A telerobot is a highly integrated system. It should be able to execute task autonomously and at the same time should be able to cooperate with human operators in multi-level. The telerobot with such requirements should cover wide spectrum of technology from AI to manipulator control. Not only the development of various component for the telerobot but also the integration scheme of them should be well considered.

The first stage of teleoperator development was providing remote manipulators with autonomous functions. The accumulation of task repertories and related task data was the basis of such autonomous functions [1-4]. Recent telerobots [5-8] and telerobotic systems [9-11] incorporate knowledge base or world model. While the effects of accumulated autonomous functions have been recognized in a general sense, the framework for the knowledge base and the world model has been paid less attention from robotics researchers. The point is that the framework should take into account not only simple data and procedures to handle objects but also infrastructural functions for cooperative execution of tasks. The **cooperative execution framework** considering the nature of telerobot tasks is required.

Cooperative execution of telerobot tasks is necessary for several reasons [7-8,12-13]. Robots are not enough intelligent that they cannot make complete plan of tasks. So, human operators should help by expanding tasks into sequence of elementary operations. This is the task level intervention. The object level intervention, lower than the task level one, is also necessary because robots cannot recognize environment and objects as human operators can do. So, human operators should help robots by teaching.

Robots needs further intervention at the lowest level, i.e. at the manipulator level. It is the case robot cannot generate skillful motion to do the task and is very different from intelligent intervention usually incorporated in the conventional supervisory controls [12]. We need further consideration on this issue.

Autonomous functions provides repetitive execution of task. On the other hand, fixed set of task repertories lacks adaptability to even small changes of task conditions. It is also difficult to apply fixed task repertories to a new environment because most data needed for task programs are not available. Direct maneuvering of a telerobot is suitable to execute task immediately, while repetitive task execution is tedious and it is difficult to control a fine and compliant motion from the remote site because of the degraded communication channel. Since space telerobots feature a wide spectrum in both task environment and communication capability, there should be a new cooperative way of teleoperational task execution between a robot and a human operator, i.e. **superposition of autonomous function and direct maneuvering.**

In addition to this, utilization of both the autonomous functions and the direct maneuvering generates another problem specific to telerobots. It is the matter of consistency maintenance of the robot knowledge while accepting the multi-level human intervention. On of the most important and recent topics related to this issue is the world model management [6-8]. As far as tasks are executed through the autonomous functions of telerobots, automatic keeping of the world model with the real environment data is possible by introducing world model maintenance instructions as discussed in this paper. However, if the tasks are executed through the direct maneuvering of a human operator, those maintenance instructions will not work. The **automatic management of the world model** while accepting the human intervention is required.

The Model Enhanced Intelligent and Skillful TEleRobot (MEISTER) [15], being developed at the Electrotechnical Laboratory, is an integrated test bed to study the multi-level cooperation between man and a robot. The MEISTER features the object model which works as the framework to accumulate task repertories with integrity, the structured master-slave manipulation system which can superpose both autonomous task execution of robots and direct maneuvering of a human operator, and the motion understanding system to realize the world model management even for the direct maneuvering. In the followings, we present prototype of the MEISTER and discuss those features suitable to space telerobots.

## 2. Object Model Suited for Telerobot [15]

Compared to the industrial robots on production lines, the telerobots are required more flexibility with respect to the task conditions. Objects are moved and their relationships in environment changes dynamically. Objects are to be handled even if some data are indeterminate yet in emergency. Procedures to handle an object should be arranged according to the changeful environment.

### 2.1. Outline of Teleoperation Using Object Models

Figure 1 outlines a telerobot system equipped with object models. Each object model contains knowledge to execute tasks. To start a task to handle an object, the operator gives a command of the task to the corresponding object model. Then the model supports the task execution.

The model of an alcohol lamp, for example, contains knowledge such as procedure for lighting and data of the wick point. Procedures and data for pick and place tasks are also available as general knowledge through hierarchical modeling system. The operator may command the model of lamp to light it. When all the information necessary for the task or the environment satisfies conditions to start the task, the lighting procedure is executed by the robot automatically. If the model lacks any data necessary for the task, it invokes special procedure in which the model acquires the data while the operator executes the task. The operator can also command a model to execute a part of the task or from the middle of it. The purpose of the teleoperation with object models is to enable the utilization of autonomous functions with immediacy and flexibility in this way.

### 2.2. Policies for Construction of Object Models

In such teleoperation style the object models offer the following features.

a) Knowledge for a task to handle the object are described, stored and accessed easily.

b) Task flow is changed according to the environment state.

c) Maintenance of consistency between model data and real environment is done automatically.

d) Knowledge for primal tasks in teleoperation such as pick and place is prepared.

e) Ta k can be started quickly even when model data is incomplete.

In the construction of the MEISTER system, we have introduced three policies for the object model to be suited for teleoperation.

**Making a module of handling knowledge object by object:** In contrast with collecting handling knowledge based on procedure types, the knowledge is collected based on the types of objects to be handled. It brings about the following merits:

a) Since a variety of task knowledge using an object are packed in the same framework, they can be retrieved intuitively by specifying the name of the object.

b) Changing task flow according to environment becomes easy. Since information representing the state of the object in the environment is stored object by object, not distributed over the whole programs, the state of the object can be checked easily.

c) Management for model data changing along with task execution becomes easy. Information and procedure to handle an object is stored in the same framework. Management process can also be stored in the same framework. These two features of the object model enables to generalize and concentrate the management process.

**Define "general handling model":** Pick and place tasks are fundamental and frequent in teleoperation. They are essential operations in object handling tasks. Either of them is composed of the same motion patterns for wide variety of objects. Therefore we define a general model of handling objects called 'general handling model' to describe the properties and procedures common to all the objects as a target of pick and place tasks. On the other hand, a model for the class of specific object like an alcohol lamp is called a 'specific object model.' Figure 2 illustrates the hierarchical relationships among the general handling model and specific models. Typical properties and procedures of the models are listed in the figure.

By definition, properties and procedures of general handling model are shared by all the object models. It corresponds to the inheritance mechanism of the object oriented system by which we have implemented the models. Introduction of this model hierarchy brings the following merits:

a) Defining a specific object model, such as an alcohol lamp, to be a specialization of the general handling model, it becomes automatically the target of pick and place tasks. Consequently, fundamental object handling procedures stored in the general handling model becomes available in specific object models without any additional definition. Sharing general knowledge among the general object and specified objects saves the space for memorization.

b) It gives unity in management of environment state changing with task execution. Environment state here means object pose and affixment relationship among objects. Basically, pick and place tasks change the environment state. In other words, moving an object is mainly commanded by fundamental motion commands which constructs pick and place operation. Basic model data management is concentrated if the management procedures are built in the general handling model with pick and place related procedures.

**Introduce teaching-executing method:** As pointed out before, objects are to be handled even if some data are indeterminate yet in emergency. To cope with the situations, we introduced a teaching-executing method [4]. In this method, the model checks the availability of data. If it is not available, the model requests the operator to control the robot for the execution and the model acquires the data from the executed task. The method brings about the following merits:

a) It enables to start a task quickly by direct manual control, and also to execute the task automatically from the next time using the model data taught through the manually executed task.

b) There are cases where task executing motions themselves are not appropriate for model data teaching. Special teaching procedures for such cases can be possessed by the model. Using these procedures, the teaching-executing method realize reliable acquisition of the environmental data.

The object models of the MEISTER allow object level intervention by the teaching-executing method. They also allow task planning level intervention by macro expansion capability of tasks and confirming execution of the expanded procedures.

## 3. Structured Master-slave Manipulation System

To allow manipulator level intervention in the MEISTER system, we provide the structured master-slave manipulation system. As is going to be made clear in the followings, 'structured' means that our system offers formalized motion patterns which can be superposed on the conventional master-slave control without changing control mode.

Figure 3 shows the functional block diagram of the structured master-slave manipulation system. The available commanding devices are the master manipulator, a teach pendant, a keyboard and the time dial. Control schemes to be superposed on the conventional master-slave control are as follows.

**Resolved motion rate control scheme** [16]: The slave manipulator is moved in the selected axis of the named coordinate system at a specified velocity while the button of the teach pendant is pressed. The scheme is suited to realize precise linear motion either in joint angle space or in Cartesian coordinate space.

**Incremental control scheme:** The x, y and/or z position of the slave manipulator can be incremented in a named coordinate system at a specified amount when the operator strikes the corresponding key. Increments of rotation in each axis can be commanded in the same way. It has been reported that the scheme offers higher precision than the master-slave control scheme if signal transmission delay exists between the master and the slave [17].

**Indexing scheme:** The coordinate transformation between the master and the slave manipulator is calculated every time when the master-slave control scheme is initiated. It enables to set the slave manipulator at a pose suitable for the task execution and the master at a pose comfortable to the operator [18]. This function also virtually expands the movable range of the master manipulator.

**Software jigs** [4]: A software jig describes specific motion constraint superposed on the slave manipulator motion. It has the effect of a hardware jig, but is specified by a special software package. Consider the task of carrying a glass filled with water. The operator must focus his attention on maintaining the orientation of the glass vertical in order not spill the water. This constrained motion increases the working load of the operator and the difficulty of the task. The software jig supports the motion constraint while the jig is active and enables the operator to concentrate on only its positional movements.

**Skill superposition scheme** [19]: In software jigs, the robot supports fixed constraint motions in a task motion. However, some tasks requires more complicated control like compliant motions. Typical one is a peg in the hole, where the orientation of peg should be compliant to the hole. Even a simple placement of an object on a table requires to guard not to hit the table hardly and not to incline the object too much. The skill superposition scheme of the MEISTER supports control for these secondary motions while leaving the operator the control of whole task progression.

Figure 4 explains the effect of this scheme in a task to pull a peg out of a hole. Employed skill here is that the robot keeps the peg compliant in orientational motion. The operator controls movement parallel to the axis. The effect of the scheme is clear at the early trials of the experiment. The difference of achievement with and without the scheme decreases along with the trials. It is because the operator has acquired the skill. This shows that the skill superposing scheme helps a novice operator to achieve skillful operations.

**Programmed control scheme** [14]: When the task is controlled by a program, the slave manipulator moves along the given trajectory. If the operator moves the master manipulator, the motion of the slave manipulator is modified correspondingly.

There are several possible realization of this scheme. The following is one example. If the master manipulator is moved such that the slave leaves a virtual tube specified by absolute value of radius as threshold concentric to the calculated trajectory, the slave manipulator goes to the master position/orientation instead of following the trajectory. When the master manipulator is not moved beyond the threshold, the slave tries to return to the nearest point on the trajectory. After the slave returns to the trajectory, it continues the programmed motion.

The time dial can be used to change the speed and direction of the program progression. Its potential applications are, teaching, monitoring and error recovery of robot tasks. For example, in monitoring of the task execution, slowing down the speed of the execution makes guarded motion easy. Reversing the direction of time is useful for error recovery. By specifying appropriate time point on the simulated execution on graphics, the operator can restart the task from the appropriate step or he can skip needless sub-tasks easily.

## 4. Motion Understanding for telerobot

### 4.1. Management of World Model

So far as the robot is commanded a task with higher level instructions which include necessary instructions to update the change of the world model, it can maintain the consistency of it. On the other hand, it becomes impossible for the robot to maintain the consistency if the real world is changed by commands or events other than such instructions. Typical example is when the operator intervenes at the motion level. However motion level intervention is inevitable in telerobot tasks as discussed previously. Therefore we have developed the motion understanding system which recognizes the meaning of the operated motion [15]. Recognized results are used for the world model maintenance. Thus the motion understanding system maintains the consistency of the robot knowledge of all the level while accepting the human intervention.

What kind of task motion should the system understand? There exists wide variety of tasks to do for telerobot such as machine assembly, tool operations and so on. Among these task motions, the followings concern about the model management stated above; grasp motion, move motion, attach motion and set motion. These are so called pick and place task related motions. Since pick and place is the most basic robot task which is closely coupled with the model management, we focus on the motions for a pick and place task.

### 4.2. System Architecture

The block diagram of the motion understanding system is shown in Fig. 5. Symbolizers continuously monitor the signals such as pose and finger force of the remote manipulator, extract specific task events such as closing of the finger, and convert them into predefined symbols. Detected events are sent to the motion understanding interpreter. Cell manager realizes efficient processing by dynamically limiting the target objects to only those close to the manipulator hand. The interpreter tries to recognize the meaning of the remote manipulator motion by matching the events from symbolizers with the pre-state and post-state of the task models expressed in the form of rules. The Rule Base contains the rules.

While symbolizers work at the ratio of servo cycle of master-slave control, the interpreter works when the symbolizers report the events. The separation of processing level in this manner makes the system hierarchical consisting of the symbolizer and interpreter layers.It contributes the efficiency of overall processing because the interpreter, which is more complicated and time consuming procedure than that of symbolizers, runs only when events are reported. It also enables the notation of rules to concentrate on only the processing level of the task model description and contributes the readability of the motion understanding rules.

### 4.3. Experiment

The conducted experimental task is to transfer an alcohol lamp on a table from one place to another using the master-slave manipulator. Figure 5 shows the task environment and the loci of the robot hand: The alcohol lamp at position A is transferred to the point C over the match box B.

The system continuously monitors the distance between the coordinate center of the robot hand and the target object. At position 1, the system recognizes the approach motion by detecting that the hand comes close to the lamp. At position 2, the system recognizes that the robot hand is out of the approach region of the lamp and is in the grasping region. At the same position, the grasp motion is recognized when the system detects the closing of the slave fingers and the increment of grasping force. After recognizing the grasping of the lamp, the system generates an affixment relationship between the slave hand and the object. Hereafter the system can keep the current value of the lamp pose by monitoring the movement of the slave hand. At position 3, the system detects that the lamp is pushed to the table and then detects that the robot hand released the lamp.

This experiment shows that the system can recognize a pick and place motion of the human intervention using a master-slave manipulator. It also shows that the consistency between world model and real world can be maintained by the results of motion understanding system.

## 5. Demonstration of MEISTER system

As the benchmark of the whole MEISTER system, handling chemical apparatuses is selected. Figure 7 shows the set up of the working environment. Using these apparatuses, we have been testing the object models, the structured master slave manipulator and the motion understanding system. Typical operations and functions of the MEISTER employed are as follows.

In picking up the spoon, stored pick and place procedures and teaching-executing method of the general handling model are employed.

In scooping up sample material with the spoon, special motion for scooping up defined in the model of a spoon is employed.

In carrying the sample on a paper on the balance and carrying the paper containing the sample to the bowl, a software jig to keep the posture of the spoon and the paper is employed.

In grinding the sample, programmed motion to move the pestle on the circle trajectory is used. In the midst of the grinding, program superposition scheme is employed to crack a specific particle by the intervention through the master manipulator.

In rearranging the alcohol lamp position, the human operator uses the master manipulator. This operation is recognized by the motion understanding system and the model data of the lamp is updated automatically.

In lighting the alcohol lamp, special task procedure defined in the lamp model and the updated data is used. In striking the match, special procedure defined in the match model is called.

Returning operation of the spoon are basically executed by the robot automatically using the data taught in the teaching-executing method in pick up operation of them.

The rest procedures for a flame reaction experiment is executed in the same manner. These operations and employed schemes of the MEISTER show the variety of telerobot tasks and the usefulness of multi-level cooperation of man and the robot.

## 6. Conclusion

This paper presented the Model Enhanced Intelligent and Skillful TEleRobot (MEISTER). It consists of knowledge base on the object models, the structured master-slave manipulation system and the motion understanding system. These three components form a trinity to realize multi-level human intervention and task cooperation in telerobot tasks.

It is quite natural for the space telerobot, which will achieve variety of missions in space station, shuttle and so on, to have handling knowledge about objects and execute tasks autonomously basedupon it. Flexible modification capability of robot task execution at multi-level will contribute wide application of telerobots to material, biological, chemical experiments etc., and also in changeful task environment conditions in space.

The MEISTER is an integrated test bed to study the intelligence for telerobots.

## 7. Acknowledgment

The authors would like to express their great appreciation to Dr. T. Yuba, manager of the intelligent systems division of Electrotechnical Laboratory, and Dr. M. Kakikura, chief of the autonomous system section for the promotion of this research. They are indebted to Dr. K. Takase and Mr. T. Suehiro of Electrotechnical Laboratory for providing the ETA-II Direct-Drive manipulator as well as their invaluable advice on this research. They are also indebted to Dr. T. Ogasawara and Mr. T. Matsui for providing ETALisp and LEO, lisp and object oriented system developed for robotics research, with which the MEISTER system has been implemented.

## 8. References

[1] D. E. Whitney, "State Space Models of Remote Manipulation Tasks," Proc. 1st IJCAI, pp.495-507, 1969.

[2] W. R. Ferrell and T. B. Sheridan, "Supervisory control of remote manipulation," IEEE SpectrumOctober, pp.81-88, 1967.

[3] R. Fournier, P. Gravez and C. Mangeot, "High-Level Hierarchical Control in Computer Aided Teleoperation (CAT)," Proc. 87 ICAR, pp.411-420, 1987.

[4] T. Sato and S. Hirai, "Language-Aided Robotic Teleoperation System (LARTS) for Advanced Teleoperation," IEEE J. Robotics and Automation, Vol. RA-3, No. 5, pp.476-480, 1987.

[5] T. Sato and S. Hirai, "Design of Teleoperator System utilizing the World Model (in Japanese)," J. of the Rôbotics Society of Japan, Vol.4, No.4, pp353-362, 1986.

[6] J. S. Albus, R. Lumia and H.McCain, "Hierarchical Control of Intelligent Machines Applied to Space Station Telerobots," Proc. Space Telerobotics Workshop, Pasadena, CA, 1987, Vol. 1, pp.155-165.

[7] N. O. Sliwa and R. W. Will, "A Flexible Telerobotic System for Space Operations," ibid., Vol.2, pp.285-292.

[8] L. E. Parker and F. G. Pin, "Architecture for Task Allocation in a Man-Robot Symbiotic System," presented in 1987 SPIE Cambridge Symposium: "Advances in Intelligent Robot Systems,"

[9] T. Matsui and M. Tsukamoto, "An Integrated Teleoperation Method for Robots using Multi-Media-Display (in Japanese)," J. of the Robotics Society of Japan, Vol.6, No.4, pp.33-42, 1988.

[10] T. Hasegawa, T. Suehiro and T. Ogasawara, "Model-Based Integration of Environment Description and Task Execution, " NATO Advanced Research Workshop "CAD Based Programming for Sensor Based Robots" IL CIOCCO, Italy, 1988.

[11] Ph.Even and L.Marce, "PYRAMIDE:AN INTERACTIVE TOOL FOR MODELLING OF TELEOPERATION ENVIRONMENTS," Proc. IROS '88, Tokyo, 1988, pp. 725-730.

[12] L. Conway, R. Volz and M. Walker,TELE-AUTONOMOUS SYSTEMS:METHODS AND ARCHITECTURES FOR INTERMINGLING AUTONOMOUS AND TELEROBOTIC TECHNOLOGY, Proc. 1987 IEEE Int. Conf. on Robotics and Automation, pp. 1121-1130.

[13] V. Hayward, "AUTONOMOUS CONTROL ISSUES IN A TELEROBOT," presented at IEEE Systems Man and Cybernetics Conf., Pekin, China, Aug. 1988.

[14] T. Sato and S. Hirai, "MEISTER:A Model Enhanced and Skillful Teleoperational Robot System," in R.Bolles and B.Roth eds. Proc. 4th ISRR, pp. 155-162, the MIT Press, 1987.

[15] S. Hirai and T. Sato, "OBJECT MODEL FOR TELEROBOT," Proc. IROS '88, Tokyo, 1988, pp. 725-730. pp. 701-706.

[16] D. E. Whitney, "Resolved-motion rate control of manipulators and human prostheses," IEEE Trans., Man-Machine Syst., Vol. MMS-10, No. 2, pp. 47-53, 1989.

[17] G. P. Starr, "A Comparison of Control Modes for Time-Delayed Remote Manipulation," IEEE Trans. SMC, Vol. SMC-9, No. 4, pp. 241-246, 1976.

[18] S. M. Killough, H. L. Martin, H.L. and W. R. Hamel, " Conversion of a Servomanipulator from Analog to Digital Control," Proc. of IEEE Conf. on Robotics and Automation, pp. 734-739, 1986.

[19] S. Hirai, T. Sato and H. Urabe, "Teleoperation using Skill Superposition (in Japanese)," preprints for the Fifth Annual Conference of Robotics Society of Japan, pp. 99-100, 1987.

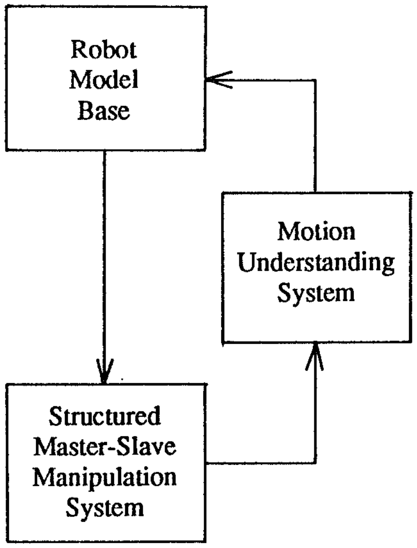

Robot Model Base
Motion Understanding System
Structured Master-Slave Manipulation System

Fig. 1 MEISTER system architecture.

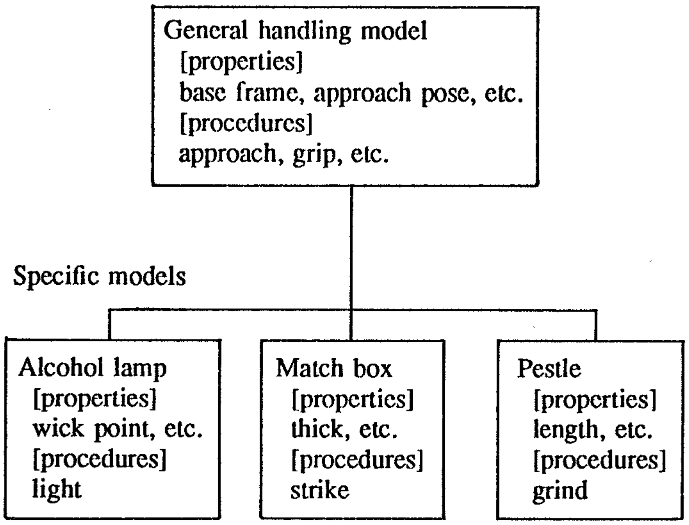
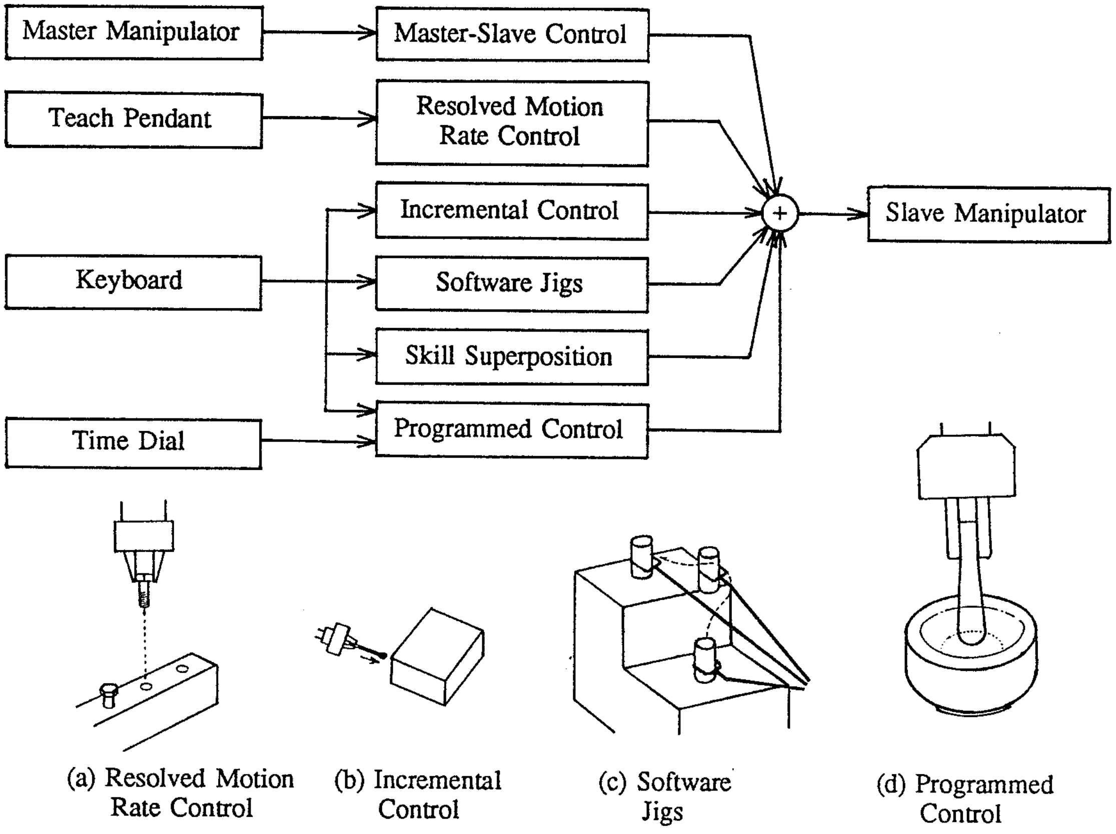


Fig. 2 Hierarchy of object models.



(a) Resolved Motion Rate Control

(b) Incremental Control

(c) Software Jigs

(d) Programmed Control

Fig. 3 Block diagram of Structured Master-Slave Manipulation System.

Table 1
Results from first to sixth trials.

| Trials | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| without Skill | s | f | f | f | s | s |
| with Skill | s | s | s | s | s | s |

s: success f: fail

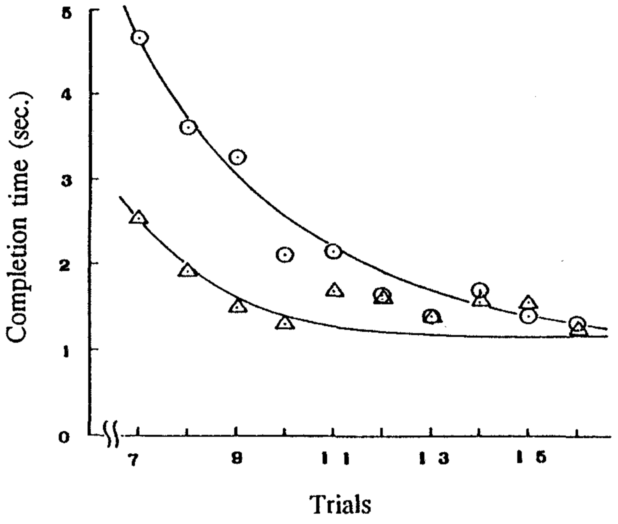


Fig. 4 Effects of skill superposition. "⊙" denotes result without skill, "△" denotes result with skill.

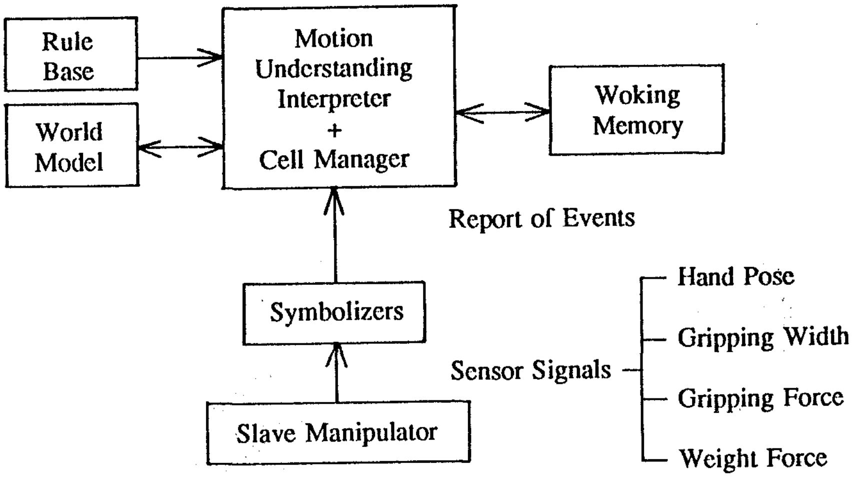


Fig. 5 Block diagram of motion understanding system.

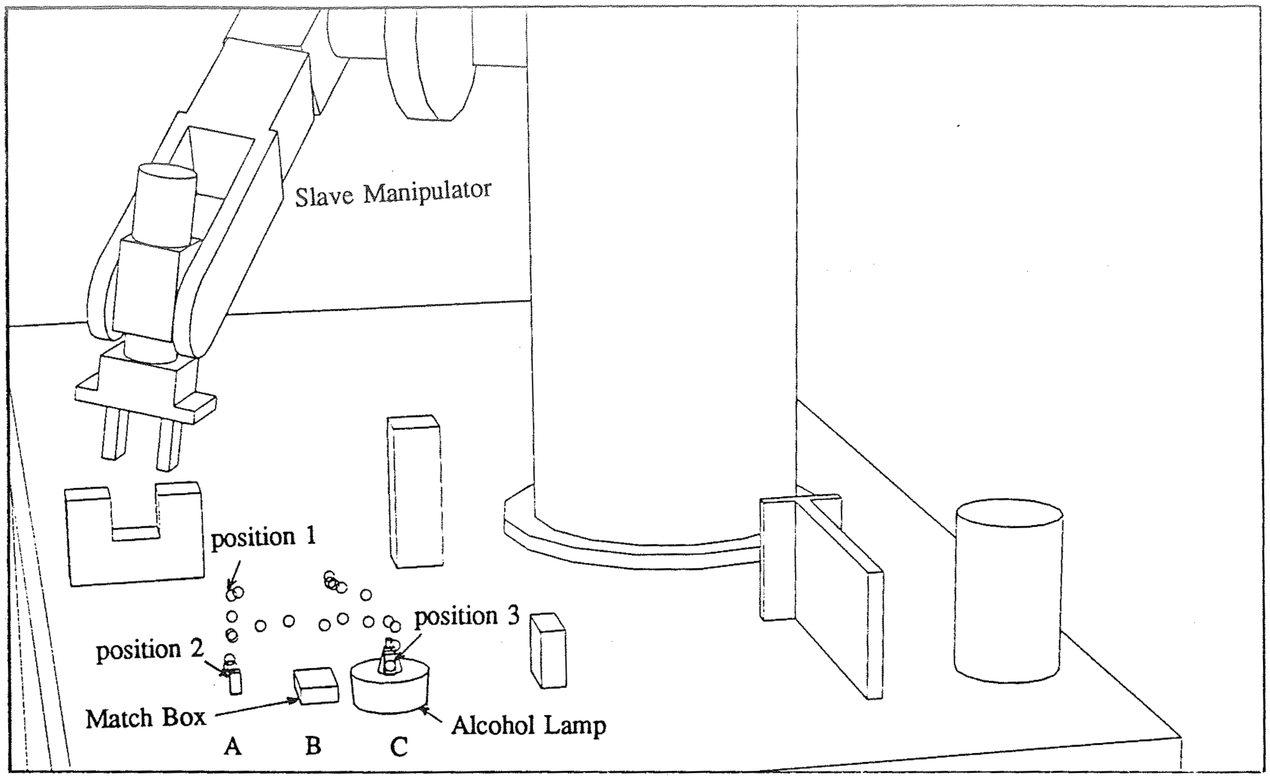


Fig. 6 Motion understanding experiment of carrying alcohol lamp.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

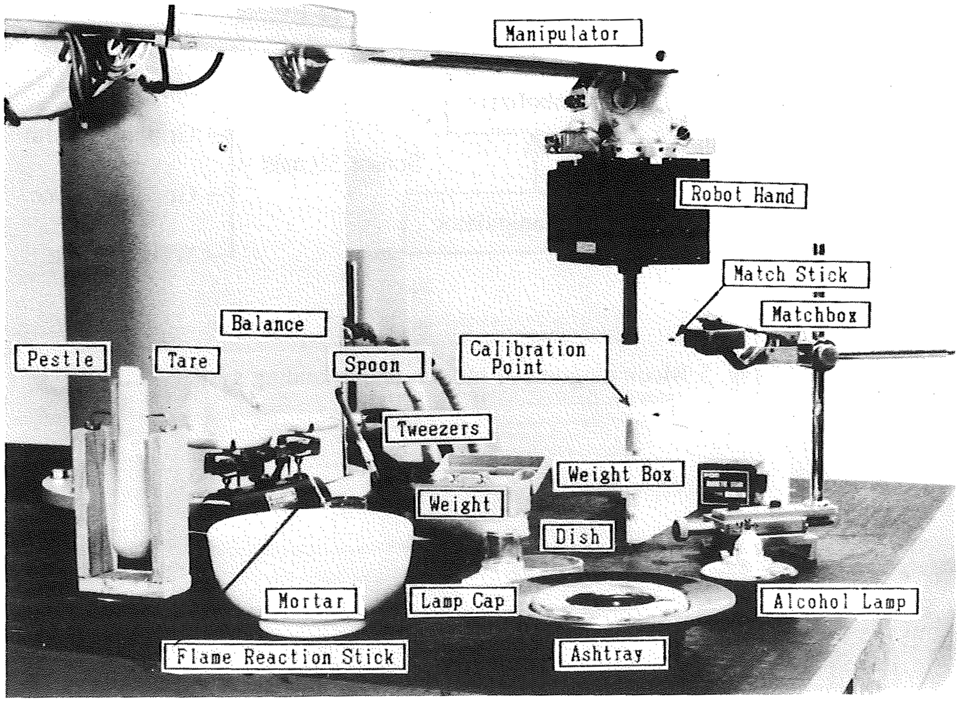


Fig. 7 Experimental working environment.

# An Alternative Control Structure for Telerobotics

P.T. Boissiere and R.W. Harrigan
Sandia National Laboratories *
Albuquerque, N.M. 87185

## Abstract

This paper discusses a new telerobotic control concept which couples human supervisory commands with computer reasoning. The control system is responsive and accomplishes an operator's commands while providing obstacle avoidance and stable controlled interactions with the environment in the presence of communication time delays. This provides a system which not only assists the operator in accomplishing tasks but modifies inappropriate operator commands which can result in safety hazards and/or equipment damage. Research and development of this concept is being carried out in the Telerobotics Research Laboratory at Sandia National Laboratories.

## 1 Introduction

This paper describes the KRITIC (Knowledge based Review and Intervention To Impose Constraints) controller, a multiprocessor layered control architecture for telerobotic systems. In this concept, a computer local to the robot serves as an intelligent agent to monitor operator commands and to perturb those commands, if needed for safe operation, based upon environmental constraints. This provides automatic obstacle avoidance and controlled interactions with the environment even in the case where significant time delays are present in the communication link between the operator and robot. Such time delays exist, for example, in earth-based control of robots in space. Responsive, yet safe operation is achieved by adjusting the robot's speed to the computational capabilities of the controller. Thus, the robot moves slowly in cluttered environments where extensive obstacle avoidance computations may be necessary and quickly in obstacle-free environments. While operating in an obstacle-free environment, the KRITIC controller monitors inputs from force and proximity sensors in order to respond to obstacles not represented in the world model. Implementation of this control concept in a system containing a PUMA 560 robot is discussed.

This work is motivated by the difficulty most operators experience while controlling a robot using either a teach pendant or joystick. The difficulty arises from several sources. Most robot manipulators can be operated in any of several coordinate frames. The *best* coordinate frame depends strongly on the task to be executed and skill of the operator. Coordinate frames can, in fact, change during execution of a task. In addition, most commercial robot manipulator systems do not provide sensory feedback to the robot operator. Tasks which involve, for example robot contact with the environment can prove to be especially difficult in the absence of force feedback to the operator. Even operations involving visual feedback (especially in the form of a conventional television monitor as in a teleoperated system) can be difficult to execute even for highly trained operators [7].

*Work Supported by the U.S. Department of Energy at Sandia National Laboratories under Contract DE-AC04-76DP00789

Mohr [8] discusses recently initiated research programs to develop the concept of *telepresence* in which sensory feedback to a human operator would be of such quality that the operator would feel as if he or she were co-located with the robot experiencing the same interactions with the environment. Research has been undertaken on the influence of robot environmental and status displays on operator performance [10]. Bejczy and Handlykken [2] discuss the incorporation of force reflection to the operator in a master/slave manipulator system. Of particular concern in space-based teleoperated robotic systems is the impact of communication delays on system control stability when operators controlling the robots are on earth [1]. Much of this work to compensate for time delays has centered around the development of predictive displays which show the operator what the results of a robot command will be before the robot actually executes the command [5]. An additional problem, in the case of a robot in space, is that the communication bandwidth necessary to transmit the sensory feedback to the operator may not be feasible.

The KRITIC control concept integrates computer with operator control. Other approaches to such integrated control of robot manipulators have been referred to as *human supervisory control* by Sheridan [11] and *telerobotics* by others [9]. The term telerobotics will be used here. In the telerobotic control architecture discussed in this paper the sensory feedback to the operator can be limited to visual information. Other feedback from the sensory subsystems (*i.e.*, force and torque, robot position, and proximity) is used locally by the KRITIC controller to stabilize the robot system while interacting with the environment, accomplishing object avoidance, and avoiding robot and workspace limits.

Advantages of the KRITIC controller include reduction in the amount of information which must be transmitted to the operator and system stability in the presence of operator/robot communication delays. These attributes are very desirable in environments such as space and underwater exploration where communication bandwidths are limited. The integration of human and computer control described in this work not only aids the operator in the presence of poor sensory feedback and communication delays, but provides a redundancy check on commanded robot motions. This significantly increases robot safety independent of the distance between robot and operator.

## 2 An Architecture for Telerobotic Control

The basic function of the KRITIC controller is to monitor operator commands, evaluate the impact of those commands with respect to the robot's operating environment, and modify the commands if necessary to achieve the *intention* of the operator in a safe manner. Determining the *intention* of the operator is, of course, a difficult task in general but, with respect to KRITIC, the basic assumption is that the operator is serving in a supervisory position. However, the operator may not know all the constraints which may prevent the completion of a commanded operation. Thus, the operator commands are assumed to be basically valid but may require perturbations to account for environmental and robot constraints.

The KRITIC controller must possess several characteristics. It must provide for smooth transitions from one constraint region to another without direct operator intervention, it must provide for both model-driven control and real-time servo control, and it must achieve the intent of the operator in a natural manner that produces no surprises. Otherwise the operator will not trust the system. The operator's input to the system is evaluated with respect to an approximate world model and sensory information. The KRITIC controller determines what modifications, if any, to the commanded motions are appropriate. These perturbed motions are then communicated to the robot controller. The world model is constructed by combining *a priori* knowledge (usually a mathematical description of the robot and its work environment) with sensory information (*e.g.*, vision) and then associating this knowledge with constraints defined within the world model. Environmental regions are modeled as *exclusionary* (the robot must not enter a particular region of the environment under any circumstances) or *cautionary* (the robot may enter a particular region of the environment but satisfy predetermined conditions when doing so). Environmental constraints associated with a region typically depend on the task. A cautionary region in one task may become an exclusionary region in another. Nonenvironmental constraints such as robot joint limits are independent of task.

Exclusionary regions include obstacles within the environment which the robot must not contact. The robot must either avoid or stop before reaching an exclusionary region. There should be no way for the robot to enter the region defined as exclusionary. Cautionary regions include objects with which the robot may interact (*e.g.* , touch) but must do so in a controlled manner. For example, a table-top may represent a cautionary region in that the robot may place something down on the table; however, the robot must place the object down on the table *gently*. Thus, associated with cautionary regions are constraints which define how the robot should interact with the environment. As illustrated in the example of placing an object flat on a table top, the constraints are the limits to the forces and torques of interaction. These constraints are implemented by adjusting the forward path gains and setpoints for the control law which allows the robot to comply with the table top as it makes contact. In this case, the robot would be constrained to servo about zero torques and a predefined normal force. Therefore, the constraints on robot motion are zero setpoints for the three torques and, for example, a one pound setpoint for the robots world $Z$-axis force and the forward path gains which allow the robot to approach and contact the table top in a stable manner.

Constraint free regions contain no constraints except for those governing the stable control of the robot. Characteristics of constraint free regions of the environment include direct pass through of operator commands to the robot, and a willingness to accept relatively high contact forces should the robot inadvertently collide with an object in the environment. Basically, a constraint free region is one where there is high certainty that an arbitrary operator command will not result in damage.

A layered parallel control structure was developed to achieve the above characteristics [3]. Figure 1 shows the basic structure of the KRITIC controller and its interactions. The basic concept behind the KRITIC controller architecture is that complex system behaviors can be developed through linear combinations of simpler responses to sensory or mapped information. The relative contribution of each control layer to the overall system behavior is determined by adjusting the weighting coefficients ( $k_i^{\prime}s$ in Figure 1) associated with each layer. Thus, as the task varies, the relative weighting of the $k_i^{\prime}s$ can be adjusted to develop the desired system response.

Perhaps the most important feature of the KRITIC architecture is that the model of the environment and the sensory information pervade the control system at all levels. Since the KRITIC controller is layered there is no single computational bottleneck through which each command must pass. Each layer runs all the time with all control layers individually contributing perturbations to the original operator command. The magnitude of the perturbation varies with time depending upon the robot's perceived location within the environment or on sensory feedback. Each layer of the KRITIC controller can be rather complex and, in fact, layered as well.

This architecture differs from the subsumption architecture of Brooks [4] in which a given layer of control suppresses, or subsumes, the behavior of other control layers. Only one control layer at a time actively influences the system's overall behavior. The KRITIC controller's architecture allows all control layers to influence the overall system behavior at all times. This allows a range of system behaviors to be derived from a limited number of basic response actions.

When an operator command is received by the KRITIC controller it is directed to all control levels simultaneously. The basic idea of the controller is that as soon as an operator's command is received, the *Pass Through* (Figure 1) layer communicates this command to the robot controller. This continues as long as the operator directs the robot to move in a given direction. At the same time, however, the

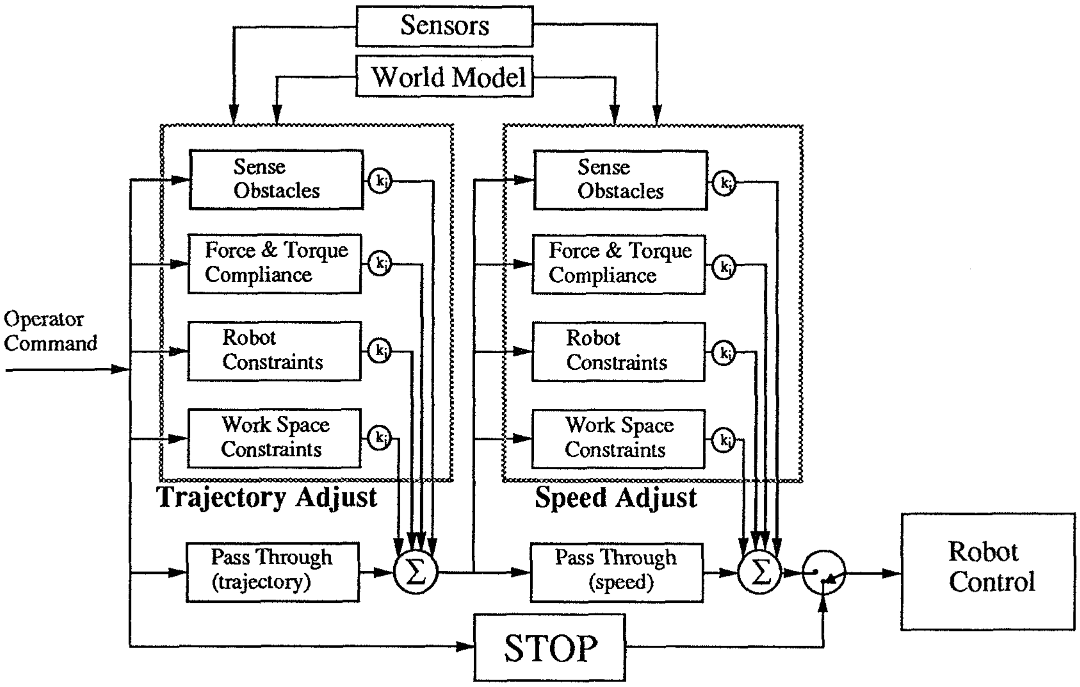


Figure 1: KRITIC Control Architecture

other control layers receive the same command and develop perturbations to the original command based upon inputs from the world model or real-time sensors. These trajectory perturbations are applied to the operator's command (exiting from the *Pass Through* layer) to generate a new robot trajectory. Since the complexity of the control algorithms within the different layers of the KRITIC controller vary from one layer to the next, the outputs from the different layers of the constraint analyzer are not synchronized. Thus, all perturbation computations must be done quickly relative to the speed of the robot. When the robot enters a region where the world model indicates obstacles and thus significant perturbations to the commanded motion are likely, the *Speed Adjust* portion of the constraint analyzer reduces the robot speed. The amount of speed reduction must be balanced to the computational requirements of the constraint analyzer to provide a responsive yet stable control system.

Thus, within the control layers of Figure 1, for example, the *Workspace Constraints* layer checks for the location of known obstacles and limits to the work space within the environment with respect to the robot's location and produces perturbations to the robot's motion to avoid the obstacle or limit. Similarly, the *Robot Constraints* layer monitors the robot's approach to conditions such as joint limits and singularities and provides perturbations to avoid these situations. Sensing of the environment is also provided. Force and torque compliance layers are shown explicitly since this control layer allows interactions with the environment. Other sensing modalities, such as ultrasonic proximity sensing, are included as well, as indicated by the *Sense Obstacles* layer.

As the robot approaches the vicinity of either a sensed or known obstacle (*i.e.*, exclusionary region), the speed of the robot is gradually reduced. The intent is to reduce the robot's speed to allow time for the robot's controller to respond without applying excessive force in the event the robot strikes something. Eventually, if the robot approaches an obstacle closely enough and cannot go around the obstacle, its speed is reduced to zero.

Thus, the robot's direction is first computed followed by computation of an appropriate speed depending on the presence of obstacles. An additional control layer of the constraint analyzer is the *STOP* layer which, rather than perturb the operator command, overrides it with an emergency stop. The robot should immediately cease all motion when the operator stops commanding the robot to move. In the architecture shown in Figure 1, this is accomplished by activating the *Stop* layer whenever there is no operator input. The *Stop* control layer is unperturbed and communicates directly with the robot controller. As such, *Stop* serves the role of a software emergency stop. Thus, any delayed perturbations passing out of KRITIC which might result in residual robot motion after cessation of an operator command, are intercepted and not passed on to the robot controller.

## 3 Implementation and Experimentation

As currently configured, the KRITIC controller models the joint limits and singularities of a PUMA-560 robot manipulator (*Robot Constraints*), its work environment (*Workspace Constraints*), which consist of a table top and objects (located by a vision system) on the table top. In addition, force and torque information is provided by a six axis force sensor and proximity sensing is available at the robot gripper in the tool $Z$-axis direction. An adaptation of the artificial potential field approach discussed by Khatib [6] was implemented to represent physical objects within the environment. In this representation, the robot manipulator is assumed to move in a two dimensional field of repulsive (obstacles) and attractive (goals) forces. A detailed description of this representation and it's implementation is given by Boissiere and Harrigan [3]. Figure 2 shows the movement of the robot in the presence of obstacles which results from the KRITIC controller. All deviations from straight line motion were generated by KRITIC as perturbations to the operators command.

Robot joint limits are not modeled as repulsive fields. Instead, the current joint positions of the robot are computed using the inverse kinematics and compared to the stored limits for each joint. If the robot approaches a joint limit, the robot is slowed and stopped before reaching the limit. If the robot approaches a singularity, small joint angle increments are automatically added to the affected joints allowing the robot to pass around the singularity and continue along the path smoothly.

The detection of unexpected objects within a constraint free region results in placement of an exclusionary region within the world model. Once the nature of the new obstacle is determined, the operator may reclassify the region and constraints associated with the object if desired to allow the robot to interact with the object. This may be necessary to identify or map the extent of the object.

During contact with an object input from three different control layers might be used to allow the robot to comply with the environment. The first and most important layer is the *Force & Torque* control layer. For example, if a block in the robot gripper is placed on a table, and the contact surface of the block is not parallel with the surface of the table top, torques will be generated as the block and table make contact. The control algorithm used for force and torque compliance computes the perturbations required and the robot actively complies to establish zero torque. The result is that the block is automatically placed flat on the table with the required contact force. The second control layer to have an effect during this task is the *Workspace Constraints* layer. Here the distance between the robot and the contact surface is computed from the current robot position and information contained within the world model. This distance is used to compute a perturbation which slows the robot as it approaches an obstacle in a cautionary region. The third control layer which might add a perturbation is the *Sensed Obstacle* layer. This layer complements the *Workspace Constraints* layer by providing accurate distance measurements from a proximity sensor. The sum of the perturbations from these three control layers allows the robot to contact objects in a controlled

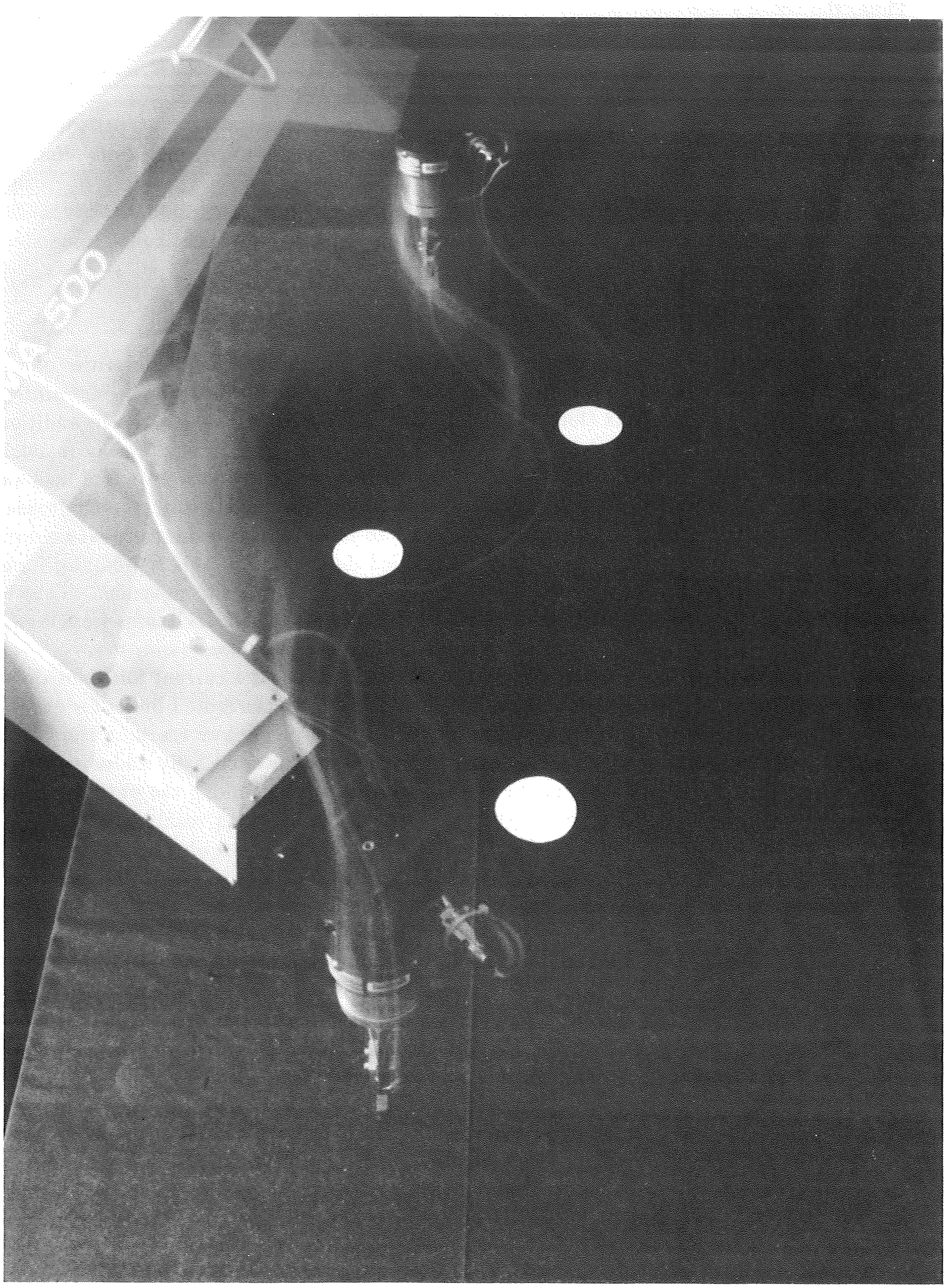


Figure 2: Object Avoidance in Telerobotic Testbed

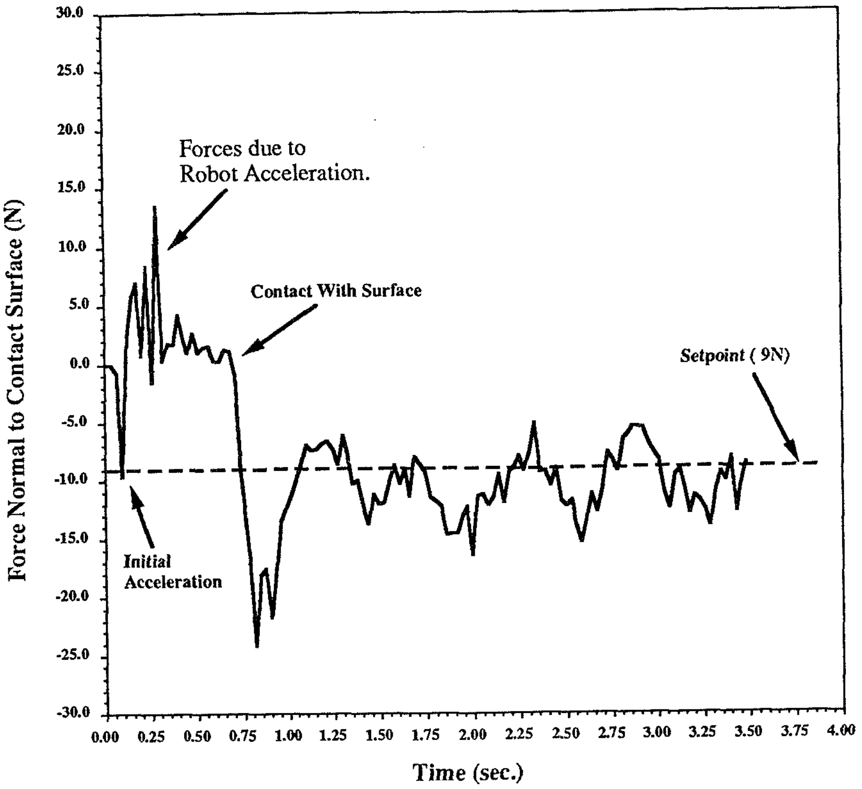


Figure 3: Force History During Contact With Table Top.

and stable manner.

Figure 3 shows the forces generated when a rectangular object held in the robot's gripper is placed on a table. The robot starts 18 inches above the table, places the object flat on the table, and then slides the object along the the surface with approximately $9N$ of force. The existence of the table top and the constraints associated with placing an object on the table top are obtained from the world model. This information is used to set the controller gains so that the robot can approach and contact the table top. In Figure 3 there is an initial overshoot of $15N$ during contact. This is generated from the interplay between all the control layers. Each control layer is designed to generate critically damped behavior, and when each layer is used individually there is no force overshoot generated. But when all control layers are in operation some underdamped behavior is exhibited. This can be controlled by manipulation of the weighting coefficients in the KRITIC controller during a specific task.

Without the proximity sensing, the robot's approach to the table top must be considerably slower since the information in the world model is typically only approximate and the exact height of the table is not known. The accurate distance information provided by the proximity sensor allows rapid approach to the table top. As illustrated in Figure 3, the time required to move to the table and establish stable contact is approximately 1.0 second. Note that, during the initial acceleration toward the table, forces are generated due to the mass of the gripper mounted on the force sensor. While under normal circumstances these forces would generate perturbations which would affect the robot's trajectory, information from the proximity sensor can be used to determine whether the sensed forces are forces generated by the acceleration of the robot or actual contact forces.

In other telerobotic control architectures delays in communication to and from manually controlled remote robot manipulators can cause instability in control. Anderson and Spong [1] have investigated

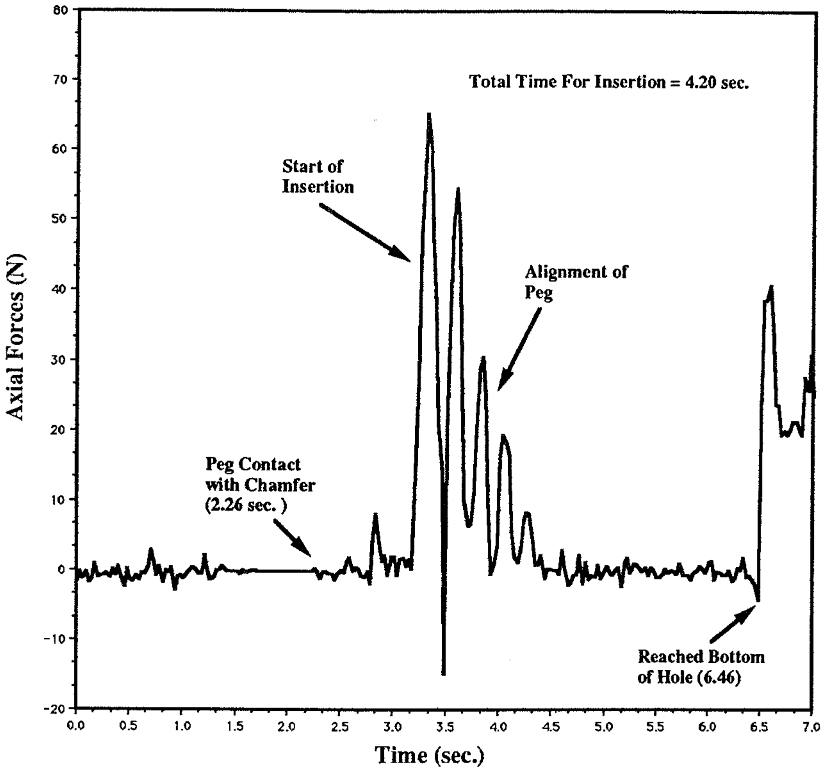


Figure 4: Peg Insertion Five Second Time Delay

the impact of time delays on control stability in manipulators in which forces generated by the robot's interaction with its environment are electronically reflected to the operator's joystick. In the KRITIC controller, communication delays do not affect the stability of the robot system since servo control of the robot is handled locally by KRITIC. In fact, time delays have the effect of placing the operator in a more supervisory role. Once the command from the operator arrives at the KRITIC controller, the control layers generate the appropriate perturbations to insure safe operation. The only effect of communication time delay is the introduction of an equivalent time delay to the execution of the next operator's command. No instabilities have been introduced into the system because stability of the system is handled locally by the KRITIC controller.

Figure 4 shows the force history for the complex telerobotic task of inserting a round peg in a hole with 0.5 mm clearance and a five second communication delay between operator and KRITIC. The operator approximately aligned the peg with the hole and directed the robot to move in the negative world $Z$-axis direction. All forces and torques are computed with respect to the tip of the peg. As the peg is inserted into the hole the forces and torques generated due to contact are used to satisfy the constraints for this task. The constraints associated with a successful peg insertion include minimizing the torques generated about the tip of the peg [12]. Note that, at the point of initial contact the axial force does not sharply increase. However, forces in the plane of the hole are generated as the peg encounters the chamfer at the opening of the hole. The KRITIC controller responds to these forces by moving the peg in the direction of the center of the hole. As the peg is centered with the opening of the hole the operators command to move in the negative $Z$-axis direction drives the peg into the hole. This causes the peg to bind and axial forces start to build as shown. At this pcint KRITIC uses the torques generated to align the center line of the peg with that of the hole. Once this has been accomplished the axial force decreases and the insertion task continues until the bottom of the hole is encountered or the operator stops the command. If the bottom of the hole is encountered while the operator is still executing the command to move in the negative $Z$-axis direction the KRITIC will stop the robot at the bottom of the hole. Using the KRITIC controller an operator can accomplish the peg insertion task in 4–5 seconds after initial contact is made. It was found that the time required to insert a peg from the same starting position and orientation was independent of the amount of communication time delay. As expected, the communication time delay had no affect on the performance of the KRITIC controller.

Notice that rather sophisticated behaviors are generated through the linear combination of relatively simple primitive behaviors. In the case of inserting a round peg into a hole, the *Force Compliance* layer of the constraint analyzer slows movement in the vertical direction to prevent jamming while the *Torque Compliance* layer simultaneously generates robot motions to eliminate any impressed torques on the peg. Finally, the *Pass Through* layer continuously tries to move the peg downward in response to operator inputs. The combined effect of the three primitive behaviors is rapid successful insertion of the peg into the hole under operator control.

## 4 Conclusions and Future Work

The KRITIC control architecture offers a feasible approach to enhancing the flexibility of robot manipulators while providing operator interfaces which reduce the impact of operator error and communication time delays. As a result, teleoperated robot manipulators incorporating a KRITIC controller can accomplish many tasks (especially those requiring interactions with the environment) faster and more reliably than standard master/slave manipulators. The use of a perturbation-based layered control concept provides an environment for real-time model based trajectory and speed control of the robot manipulator. Complex behaviors are generated by combining simple responses of the robot to sensory and model-based information. This concept naturally leads to modular control constructs and allows smooth transitions from one control mode to the next.

The specific implementation of the layered telerobotic control concept discussed in this paper will be expanded in the future. Initially the KRITIC control architecture was implemented on a PDP-11/73 [3]. Currently the KRITIC architecture is being implemented on a VME-based multiprocessor environment. The use of a VME-based computing environment will facilitate experimentation into the distribution of the KRITIC over many individual processors. The increase in computing capabilities will benefit the KRITIC in several ways. For example, as the implementation of this architecture is moved into more complex environments the World Model will also increase in complexity. In order to handle the increase in computational requirements the control layers which use the world model may themselves be distributed over several processors to maintain a responsive system.

Work on prototype telerobotic systems for robots other than the PUMA-560 is also underway. In particular, a telerobotic control structure for a CIMCORP XR6100 gantry robot is being developed for use in a project to remotely inspect and manipulate spent nuclear fuel shipping casks and other nuclear waste containers. Since this is a three dimensional environment, the repulsive potential fields used in this work will be extended and alternative approaches to object modeling will be investigated. The modular nature of the KRITIC architecture allows easy application and even mixing of various concepts. An important application area for the KRITIC control concept is hazardous environments. Telerobotic control which decreases handling times while minimizing the impact of operator error is very important. KRITIC can not only be used to constrain the robot's interaction with the environment but to enforce procedures to ensure proper sequencing of operations as well.

## Acknowledments

The authors would like to thank the following contributors for making the implementation of the KRITIC a reality: William M. Davidson for developing many of the interrupt service routines and overall software architecture, William D. Drotning and Pablo Garcia for hardware and software support for the proximity sensor.

## References

[1] Anderson, Robert J. and Mark W. Spong, "Bilateral Control of Teleoperators with Time Delay", Proceedings of 1988 IEEE Conference on Decision and Control, Austin, TX, 1988.

[2] Bejczy, A. K. and M. Handlykken, "Generalization of Bilateral Force-Reflecting Control of Manipulators", Proceedings of the 4th Ro-Man-Sy, Warsaw, 1981, pp. 242-255.

[3] Boissiere, P.T. and R.W. Harrigan, "Telerobotic Control of Conventional Robot Manipulators", Proceedings of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, PA, 1988.

[4] Brooks R.A., "A Robust Layered Control System for a Mobile Robot", IEEE Journal of Robotics and Automation, vol. RA-2, no. 1, pp. 14-23, 1986.

[5] Conway, Lynn, Richard Valz, and Michel Walker, "Tele-Autonomous Systems: Methods and Architecture for Intermingling Autonomous and Telerobotic Technology", Proceedings of the 1987 IEEE Conference on Robotics and Automation, San Francisco, Ca., 1987

[6] Khatib O., "Real-Time Obstacle Avoidance for Manipulators and and Mobile Robots", International Journal of Robotics Research, vol. 2, no. 1, pp. 90-98, 1986.

[7] McGovern, Douglas E., "Human Interfaces in Remote Driving", SAND88-0562, Sandia National Laboratories, Albuquerque, New Mexico, March 1988.

[8] Mohr G.C., "Robotic Telepresence", 1987 IEEE Proceedings: Annual Reliability and Maintainability Symposium, January 1987, Philadelphia, pp. 25-30.

[9] Pennington J.E., "Space Telerobotics: A Few More Hurdles", Proceedings of the 1986 IEEE International Conference on Robotics and Automation, vol. 2, pp. 813-816, 1986.

[10] Stark L., *et al*, "Telerobotics: Display, Control and Communication Problems", IEEE Journal of Robotics and Automation, vol. 3, no. 1, Febuary 1987, pp. 67-75.

[11] Sheridan T.B., "Human Supervisory Control of Robot Systems", Proceedings of the 1986 IEEE International Conference on Robotics and Automation, vol. 2, pp. 808-812, 1986.

[12] Whitney, D.E., "Quasi-static Assembly of Compliantly Supported Rigid Parts", Journal of Dynamic Systems, Measurement and Control, vol. 104, no. 1, March 1982.

# Integration of a Sensor Based Multiple Robot Environment for Space Applications: The Johnson Space Center Teleoperator Branch Robotics Laboratory

James Hwang, Perry Campbell, Mike Ross
Lockheed Engineering & Science Company
Houston, Texas 77058

Charles R. Price, Don Barron
Johnson Space Center
NASA

## ABSTRACT

The Teleoperator Systems Branch at JSC has developed a robotics laboratory for space robotics technology development. An integrated operating environment was designed to incorporate three general purpose robots, sensors and end-effectors, including Force/Torque Sensors, Tactile Array sensors, Tactile force sensors and Force-sensing grippers. This paper describes the the design and implementation of: (1) the teleoperation of a general purpose PUMA robot, (2) an integrated sensor hardware/software system, (3) the force-sensing gripper control, (4) the host computer system for dual Robotics Research arms, and (5) the Ethernet integration. The space applications for the above projects will be discussed and the future development for the laboratory will be presented.

## INTRODUCTION

The JSC telerobotic program aims at achieving higher levels of autonomy in space operations and to demonstrate the potential of significantly enhancing the agency's current robotic operational capabilities. The near term objective is to provide a more versatile telerobotics system that interacts compliantly with its environment. It is also desirable to have a higher safety system by providing fault-tolerance and obstacle avoidance schemes. Relative to the these goals, the program elements consist of:

- sensor hardware and software technology: vision,tactile, proximity, force/torque.
- fault diagnosis and planning.
- computing architecture and planning.
- telerobotic demonstration.

The Robotics Laboratory for the Teleoperator Systems Branch was founded in April, 1987. The objective for this Lab is to provide a flexible hardware/software testbed environment for various technology demonstrations and to fulfill the goal of supporting the Space Shuttle and the Space Station Freedom Program [1].

* This paper presents the initial results of work carried out at the JSC, LESC, under Contract No. NAS9-17900, JO. 22-110, sponsored by the National Aeronautics and Space Administration.

## LABORATORY CONFIGURATION OVERVIEW

The Robotics Lab of the Teleoperator Systems Branch is located in Building 16, Room 2000 at Johnson Space Center. This Lab currently contains 3 large robots, including a PUMA 762, a Robotics Research (RR) 1607, and an RR 2107. There are also several small educational robots, such as four Microbots, one early model PUMA 250, and a Hero 2000, for internal training purposes. Various sensors, end-effectors and grippers have been installed on these robots. The following section will give a brief description for each of these sub-systems.

### *PUMA 762 ROBOT*

The PUMA 762 manipulator is an industrial robot made by Unimation Corp. It is a six degree of freedom (DOF) robot with about 5' reach and 44 lb. of payload capacity. The controller supplied by the manufacturer is an integrated control and programming unit. The robot can be controlled by a teach pendant or by a program sequence through the VAL II programming language. Since the robot was designed basically for pick-and-place type of applications, only point-to-point motions are supported directly from the VAL II language.

### *Robotics Research 1607 and 2107 Manipulators*

The Robotics Research robots 1607 and 2107 are made by Robotics Research Corp. The 2107 has about 2100 mm reach and the 1607 has 1600 mm. The 2107 can hold only 4 lb of payload while 1607 is able to handle 50 lb. These two robots have 7 DOF and a very unique roll-pitch-roll-pitch-roll-pitch-roll configuration. The extra degree of freedom provides the capability of ORBITING the elbow while keeping the end-effector frame at a fixed position and orientation. The Type II controller with the robot provides basic teach and replay controls of the robot. No robot programming language is available for control sequence generation. Also only the point-to-point motion control is supported at this time.

### *Force Torque Sensors and Tactile Array Sensors*

The PUMA manipulator is equipped with a LORD 125/600 Force/Torque Sensor. It has 125 lb of force capacity and 600 in-lb torque capacity. The other two manipulators are equipped with JR3 UFS-4A-XX Force/Torque Sensors. One JR3 sensor has 100 lb capacity and the other one has 25 lb capacity. The transducers of these force/torque sensors are similar, consisting of a maltese-cross based strain gauge assembly which provides a set of signals which contain the force/torque data. In addition to force/torque sensors, a tactile array sensor made by Lord Corp. was also interfaced to the host computer. The tactile array sensor consists of a 10 by 16 element grid array of closely spaced deflection-measuring sensors. These sensors yield information about the contours of the object they are in contact with. All three force/torque sensors and the tactile array sensor are communicating with the host control computer through RS-232 serial lines.

### *Host Computer*

For the initial build, each of three robots were integrated to a PC/AT 286. These PC's acted as sensor controllers and programming interfaces. As mentioned above, all of those robots were designed for industrial applications, so they are not easily adapted to sensor interfacing. Because a generic programming environment was required to interface to different types of sensors and hand controllers, as well as using this sensor information to control the robot motion, a host computer was needed. For instance, a robotic system may be required to obtain the motion command from the hand controller; to collect data from the force/torque sensor, the video sensor and the tactile sensor; to display sensor data to the screen, pass the motion command to the robot, and to command the motion of the end-effector. All these activities should be controlled by a master program so that all the interactions will be coordinated. The robot programming language usually can only handle the manipulator motion well but not the sensor or end effector control. To coordinate all activities of the robot, sensors, and end-effectors, a PC/AT was used to host all these control programs and the "C" language was used to develop programs.

### *Communications*

There were at least two classes of communication needed in order to implement a telerobotic system in the Lab. The first class was the communication from local sensors to the control computer and the second was the communication among several robot control computers and the workstation. Most sensors in this Lab had RS-232 serial links (some of them even with analog output) and all three robots in the Lab were also equipped with RS-232 ports, therefore the RS-232 was used for the local communication among them. Because a regular PC/AT supported only two serial ports at 19.2 K Baud, a more capable communication device was needed. The Advanced Communication Link (ACL) by Stargate Corp. was selected to build the RS-232 communication sub-system. The ACL is actually a front-end communication processor which has a CPU on board and can handle up to 8 serial ports with Baud rate up to 38.4 K Baud. Since it has its own memory buffer for each channel and the processor on board will service the message transmission and receiving in background, the main CPU has very little overhead. A special ACL software driver was designed for all the sensor interfacing.

One of the major goals for this Lab is to provide a real manipulator operating environment for various Display and Control Workstations other than the simulated graphic scene generator. For providing a communication vehicle among these workstations and controllers, peer-to-peer networking is the most cost-effective approach. The EtherNet and the TCP/IP protocol are good candidates to implement such a network. The physical speed limit of EtherNet is 10 Mbit/sec, although 1 Mbit/sec is a more realistic estimate of the actual data rate with the overhead from many layers of protocol. The EtherNet provides a very convenient way to add a new node without disturbing the existing communication. The communication from any workstation to any manipulator can also be re-configured without any hardware change. The TCP/IP protocol adds further versatility of networking among different kinds of computers with a single programming interface. The Intelligent EtherNet controller, Thick Net EtherNet, and the TCP/IP C socket library by ExceLan Co. were then selected to build this network. Software drivers for message passing were designed and implemented on the PC/AT and several workstations.

### *Hand Controllers*

The hand controller is still an indispensable part of the whole telerobotics system. Two kinds of hand controllers, position and rate, were tested and integrated to manipulator systems. The "Space Mouse" was the first one been tested in the position controller group. It was constructed from a 6 DOF electro-magnetic field sensor made by McDonnell Douglas, named "3-SPACE ISOTRACK". Only a thin cable is connected between the free floating hand-controller and the sensor control box. The RS-232 link was used for sensor-to-control computer communication. Another type of position hand controller in the process of being integrated into the system is the Schilling Minimaster controller. A pair of Schilling 6-DOF hand controllers mounted on a common base were delivered to the Display & Control Lab. Using the Minimaster to drive a graphic simulation has been tested. The next step will be interfacing this hand controller to manipulators through the EtherNet. The other group, rate hand controllers, was also tested. Two MSI hand controllers, one a rotational hand grip, and the other a translational T bar, were interfaced to the PUMA control computer. The phase one design used an A/D interface board to measure the deflection of the MSI controller since only raw voltage output from potentiometers was available. The phase two design is still in progress; it will use a digital serial interface instead of an analog interface, and an embedded single board computer will be used to perform the analog-to-digital conversion and digital data transmission function.

### *End-Effectors*

In order to make the robot interact with the environment and any size of payload, a general purpose end-effector is needed. The gripper, instead of a special tool, was identified as the most generic form of a end-effector. Such a gripper needs to be electrically servo driven with accurate position feedback, and with force sensing capability. When a survey was conducted to locate this kind of industrial gripper, it was a surprise that very few manufacturer made general purpose electrical servo grippers, and the TeleRobotics Inc.(TRI) was the only vendor could provide both the position and the force feedback. Two TRI EP100/30 grippers were then acquired to be used on the PUMA and the RR 1607 manipulators. These two grippers can exert 66 lb. of gripping force and their fingers can open up to 4". One EP90/05P gripper was acquired for the RR 2107 manipulator. It is lighter (weighs about 3 lb without cable) and shorter for very light payload capability of the RR 2107 manipulator. All fingers of the above three grippers have strain gauges to measure the gripping force. The gripper controller offered by the vendor is a stand-alone control box which is controlled from a serial port through Text String commands. There are, however, some major design flaws of this gripper controller. For examples, it does control the force actively but only stops the gripper when the force has exceeded a preset threshold. In other words, it is not capable of gripping a rigid object while maintaining a constant force. Some other flaws exist like not always responding to a position interrogation command on time, and closing the gr$per without recognizing the force limit. Finally, it was decided an in-house gripper controller should be designed and built. More descriptions of this new controller design will be discussed in the following section.

## PROJECTS and SPACE APPLICATIONS

### *Compound Arm Project and Teleoperation Of PUMA*

The first project, named "Compound Arm Project", built in this Lab was to implement the teleoperator mode for the PUMA robot and to control a special end-effector constructed from two Microbots mounted on a common bracket, which was then installed on the tool flange of the PUMA [2].

The task was to control the PUMA by the Space Mouse Hand Controller and to move the PUMA around until the tool point had entered an imaginary washout sphere, whereupon an auto sequence would take control of robots, and execute a changeout procedure of some batteries on a mockup satellite. This simulated the idea of having a smaller dexterous dual arm robot carried by a long boom manipulator to do some servicing work, such as the Flight Telerobot Servicer (FTS) or Special Purpose Dexterous Manipulator (SPDM) on the end of the Space Station Manipulator. Additionally, the PUMA continually acquired arm position data and passed that data back to the display PC which would draw the PUMA's configuration on the screen.

The first problem which had to be solved was that the VAL II controller for the PUMA itself does not support any kind of communication utility even though several serial ports are available. In order to implement a teleoperator mode, it was necessary to establish communication between the hand controller and the robot controller. It was decided to use a host computer (a PC/AT) to control the traffic. Two sets of communication software drivers were developed: one for the VAL II controller, the other for the ACL processor installed on the PC/AT. The driver for the VAL II was written in the LSI-11 assembly language as an interrupt handler, and it was then assembled by a cross-assembler on a VAX11/785 and finally POKEed into memory by a VAL program. A segment of memory was allocated to build four ring buffers for four serial channels. Another VAL program was also developed to pull the message from the ring buffer and to execute motion commands according to the message. On the PC side, a similar program was written to submit messages to the memory ring buffer in the PC, then the message was passed to the serial port by the ACL Communication Processor. After the communication channel was established, the control PC was able to acquire 6 position data from the Space Mouse. That data, being very noisy, was filtered, processed, and appended with error checking code before being passed to the PUMA to direct the arm's position. The display PC would receive the PUMA's joint angles from the VAL controller and draw a view of the PUMA and its environment. This was to aid the operator in remote teleoperation.

### *Sensor and Force Sensing Gripper Integration*

After the Compound Arm Project was accomplished and demonstrated many times, several drawbacks were found during the demonstrations. The most significant problem was that the motion of the PUMA was commanded through a string of "MOVES" VAL commands, which only direct the arm through a sequence of point-to-point motions. No continuous velocity control was available, so that the motion appeared very sluggish and fine motions were very difficult to control. Another problem was that there was no force sensing capability at the end-effector, some battery mock-ups were demolished.

The second project was to integrate all the available sensors to the robot control system and to develop a more accurate control scheme. Software drivers were developed for all of the sensor and the gripper. The force/torque sensor information was displayed in a bar-graph form; the tactile sensor data was displayed as a 16 by 10 array with color representing the displacements of each sensing element; the gripper position was displayed as a picture of moving fingers; then all of these graph were shown on a windowed display screen. A demonstration was developed to show actual application of the force sensing gripper. The robot was guided through and picked up a number of different objects, including a real egg shell, a brick, and a soft drink can. The demonstration displayed the necessity to use a force sensing gripper to hold a fragile egg without breaking it, lifting a heavy brick without losing the grip, and finally picking up a soft drink can, pouring the contents into a cup, then crushing the can and dropping it into a trash can [3].

The above sequence was still controlled through a set of pre-programmed points and the PUMA was still operated under Point-to-Point Move mode. In order to control the PUMA with a smooth, continuous trajectory, a synchronous continuous path control mode was developed. Although the PUMA, or any industrial robot, was not designed to be operated from a Hand Controller, it did provide a special synchronous control mode called "Alter". The major benefit of the synchronous mode over the asynchronous point-to-point mode is that it can control all the intermediate points between the end-points and the time between each intermediate point is fixed. In this mode, the control computer has to be 'synchronous' with the robot, and the robot controller will ask for a set of command position every fixed time period (28 ms for the PUMA), then the control computer must respond to the robot controller by giving the next command position and also receive a current position report. The control computer, however, has to keep up with the speed of the robot and not lose the synchronization, otherwise the motion will come to a stall. The Alter mode for the PUMA was designed for altering the programmed path by applying the sensor data. To adapt this mode for the Hand Controller application, the PUMA was programmed to move to the same position forever and then the Alter mode was used to feed the Hand Controller command. This mode was tested with the Force/Torque Control and some demonstrations were developed for these studies. A separate paper will present results of this study [4] [6].

*Workstations Integration*

As mentioned in the earlier section, this Lab has the reponsibility to support the Display and Control section to establish a reconfigurable manipulator operation environment for their workstations. Two projects were in process of integration of workstations and the manipulator systems. The first one is the Vision Integration Project (VIP), which is to integrate the manipulator system with a Computer Aided Design (CAD) model based workstation. The workstation was supposed to know the 3D model of every object in its view. A 3D vision system, including some cameras, frame grabbers, and pattern recognition software, will identify the object, calculate the Cartesian coordinate of the object and send it to the workstation. The workstation will regenerate the scene of the world out of vision data and display the scene on the screen. The raw camera image will also be available to the operator. The operator can then manipulate the robot from the simulated world scene with much more rich information than the raw video image can contain. For example, the operator can rotate the viewing angle, look inside of the object, or predict the center of mass before any motion. To date, the communication and teleoperation portion of the project have been accomplished. The workstation (a IRIS 4D/70GT) was used to control the teleoperation and vision data interface. The EtherNet and TCP/IP protocol were used to implement the networking between the robot control computer and the IRIS. The IRIS was able to command the motion of the PUMA robot across the EtherNet. The Vision Process is still being developed [5].

Another project is to integrate the Multi-Purpose Applications Console (MPAC), which is being developed by the Display and Control section, to the robot system. Since the host computer ( a microVax ) for the MPAC is also on the same EtherNet and using TCP/IP, the similar protocol as the VIP will be used for this project.

*SRMS Advanced Control Project Support*

The Shuttle Remote Manipulator System (SRMS) Advanced Force/Torque Control Study is a new project in the Teleoperator systems branch. The objective for this project is to demonstrate the Advanced Closed Loop Control by application of OAST/JPL developed Force/Torque sensor and control algorithms to the Shuttle RMS in order to influence definition of future RMS upgrades. The role of the Lab for supporting this project is to provide a laboratory manipulator environment to test and validate the algorithm developed. For the initial build, the Robotics Research Manipulator and JR3 Force/Torque sensor will be used to implement these algorithms. Since the dynamic charateristics are not well defined for RR arms, model identification was the first task for this project. Some tests were done on identifying the servo control parameters, more tests have been scheduled to identify the link inertias and friction.

In order to simulate the RMS with an industrial arm like RR robot as close as possible, some issues have to be resolved. The most unique characteristics of the RMS is that it uses joint rate control with tachometer feedback instead of the common joint position control for most industrial robots. Therefore a plan has been set up to implement a rate control mode by modifying the RR controller.

### *NASREM Implementation*

NASA/NBS Standard Reference Model (NASREM) Architecture is a proposed model to implement the Space Station Telerobot Control System [7]. It was developed by the joint effort of NASA and the National Bureau of Standards. It defines the functional requirements and high level specifications of the control system for the NASA space Station Flight Telerobot Servicer. The recommended architecture is, however, very generic and suitable for any space telerobotic control system. In this Lab, a project has been planned to implement most levels of control architecture of the NASREM model using the existing manipulators and sensor systems. The objective of this project is to verify the NASREM model in a real manipulator environment to demonstrate the advantage of a hierarchical control structure, as well as to investigate any issues arising in the implementation. These studies will assist in establishing the telerobot operation procedures and the specifications of the future space station robots.

In order to implement the NASREM model on existing industrial robots, many problems have to be overcome. The regular RR robot controller does not provide the capabilities of tuning the servo parameters, interrogating the rate and torque feedback, and controlling the manipulator at joint rate or joint torque level. For higher level functions, such as dual arm coordination control and collision avoidance, the processors in the RR controller just can not provide the high number crunching capabilty required. A plan has been drafted to overhaul the RR controller so that it will provide the taps to control and measure the signals of absolute positions, joint rates, and joint torques in digital format. The acquisition of the high level host computer, an Iris 4D/70G workstation, for the RR controller has also been initiated. To match the NASREM architecture, several levels of processors will be used to implement different control levels in the Model.

The lowest level in NASREM is the servo level. To implement this level, the servo level interface upgrade will be acquired from the Robotics Research Corp. The upgrade will provide the digital taps to control the robot at the rate or the torque level. However, the functions of low level servo analog compensators, which keep the arm stable, must to be realized digitally every 2 ms. The original processor can not provide this computation power, so another dedicated processor has to be used for the servo transfer functions. The TMS320C25 digital signal processor (DSP) by Texas Instruments was selected to do the low level servo functions. Initial simulations studies have shown very encouraging results: the DSP is able to complete the calculation of current analog servo functions for 7 joints within 500 micro-second. It means that not only a programmable servo controller can be realized, also more advanced control algorithm might be incorporated as well.

The level a step higher than servo level is the Primitive. At this level, the joint trajectory is decomposed to small motions and sent to the servo controller. Sometimes the inverse kinematics is also solved in this level. The loop time for this level is about

25 ms, which is still synchronous. This level of control is planned to be implemented by a MultiBus embedded processor board. This processor will be equipped with a complete disk operating system (DOS) for the high level language program development. At this level, the Force/Torque Sensor data will be integrated to the control law.

The E-move level and the Object level will be implemented in the IRIS computer. At these two levels, the world model will be established, the Vision data will be incorporated, the collision avoidance will be planned from the model, and the dual coordination task will be executed.

There are three more levels above the Object level, the Task, Service Bay, and the Mission. Some functions of the task level controller can still be done by the IRIS computer, while the other higher control levels will need an AI based machine to coordinate the operations of workstation, the operator side, and the operation of the manipulator, the manipulator side. These functions will be expanded in the future.

### *IN-HOUSE GRIPPER CONTROLLER DESIGN*

The gripper controller which came with the TRI gripper was not sufficient to implement all the functions needed. Therefore, an in-house design of the gripper controller was planned. The phase one design of this project was to build a controller based on PC using off-the-shelf components as much as possible. A PC based motor controller with optical encoder decoder and a PWM power amplifier were the only hardware components needed to build such a controller. The force sensing circuit on the gripper was modified to incorporate the A to D circuits on board. After these modifications, all the sensing signals and control signals were digital and could be controlled by the PC directly. A software driver was written for this controller to incorporate all the functions in the old controller in C functions format. New functions were also added such as the active finger force control, the motion abort capability, the synchronous positioning, and error code returning. The phase two design will be moving all the hardware into a stand alone control box with a processor much like the old controller design. However, all the improvement and added functions will be integrated to the new design. The control computer will communicate to this controller through a serial link or some digital I/O lines for predefined motions.

## CONCLUSION

This paper has given a overview for the Robotics Laboratory of Teleoperator Systems Branch in JSC. Various space telerobotic application research projects have been initiated in this Lab. For the first year and half, most effort has been focused on setting up an integrated sensor based manipulator testbed environment. In order to further support the Space Shuttle and the Space Station Freedom Programs, the spectrum of research has been directed toward the technology development for demands from space programs. In the future, the lab will emphasize on the following research aspects:

- Hierarchical robot control architecture implementation.

- Coordinated multisensory perception (sensor fusion).
- Force/torque and tactile sensing for dexterous manipulator.
- Expert system for automated task planning, and collision avoidance.
- Model-based Telerobotic Control.
- Multiple arms coordination.
- Fault-tolerant manipulator systems.
- 2D, 3D vision and range imaging.

## REFERENCES

[1] C. J. Hwang, "PROPOSAL FOR OBJECTIVES OF ROBOTICS LABORATORY", LEMSCO, Houston,TX, July 1987.

[2] M. Ross, P. Campbell, "ROBOTICS LABORATORY COMMUNICATION SUPPORT FOR COMPOUND ARM PROJECT", LEMSCO-25238, June 1988.

[3] M. Ross, P. Campbell, "ROBOTICS LABORATORY SENSOR AND GRIPPER INTEGRATION PROJECT", LEMSCO-25246, July 1988.

[4] P. Campbell, "ROBOTICS LABORATORY FORCE FEEDBACK PROJECT", LEMSCO-25251, September 1988.

[5] M. Ross, "PROTOTYPE OF REMOTE NETWORK MANIPULATOR CONTROL FOR EF2 VISION INTEGRATION PROJECT", LEMSCO-26071, October 1988.

[6] P. Campbell, "REAL-TIME CARTESIAN FORCE FEEDBACK CONTROL OF A TELEOPERATED ROBOT", NASA Conference on Space Telerobotics, Pasadena, CA, January 1989.

[7] J. S. Albus, H. G. McCain, R. Lumia, "NASA/NBS STANDARD REFERENCE MODEL FOR TELEROBOT CONTROL SYSTEM ARCHITECTURE (NASREM)", National Bureau of Standards Robot Systems Division, December, 1986

# MANIPULATOR CONTROL 2

# Requirements for Implementing Real-Time Control Functional Modules on a Hierarchical Parallel Pipelined System

*Thomas E. Wheatley, John L. Michaloski and Ronald Lumia*
National Institute of Standards and Technology
Bldg. 220 Rm. B124
Gaithersburg, MD 20899
February 1989

**Abstract**

Analysis of a robot control system leads to a broad range of processing requirements. One fundamental requirement of a robot control system is the necessity of a multicomputer system in order to provide sufficient processing capability. The use of multiple processors in a parallel architecture is beneficial for a number of reasons, including better cost performance, modular growth, increased reliability through replication, and flexibility for testing alternate control strategies via different partitioning [GAG86, NAR86]. A parallel architecture mandates three additional requirements not found in a sequential architecture. First, the problem must be partitioned into tasks. Hierarchical structuring will be used to partition a robot controller, with each level in the hierarchy corresponding to a separate task. Each task level is further modeled as a virtual control loop. Second, each task must be scheduled for execution on one or more processors. The problem of scheduling processes to processors will highlight the importance of the programming environment. Third, synchronization of control and data flow (communication) must be performed during execution. A survey of the progression from low-level control synchronizing primitives to higher level communication tools will be presented. The system communication and control mechanisms of existing robot control systems will be compared to the hierarchical control model. The impact of this design methodology on the current robot control systems at NIST will be explored.

## 1.0 Partitioning Requirement

The partitioning of a job into cooperating processes is called multitasking or multithreading. The effectiveness of a parallel implementation depends on the inherent parallelism of the algorithms and the overhead of interprocessor communication. Algorithms that are purely sequential will not run faster on a parallel machine. To achieve the benefits of parallelism, the cost of communications between processors must not exceed the time savings obtained by parallel execution on the different processors. Job partitioning is usually discussed in the context of a fine-grain versus coarse-grain approach [KRU87, LYO87, LYO86]. An entire process which is allocated one processor for the life of its execution could be considered coarse-grain. Solutions to many problems are too complex to be calculated within an appropriate time frame with a simple coarse-grain sequential execution and may require a fine-grained processing approach.

Parallel machines exhibit different system capabilities. General-purpose coarse-grain machines tend to maximize batch throughput. Supercomputers tend to specialize in fine-grained number crunching in parallel for compute-bound operations. Other fine-grain computers specialize in data-intensive operations. The variety in machine capabilities leads to the distinction between systems that maximize the throughput of many jobs, known as *throughput-oriented multiprocessors*, and systems that maximize the execution of one process, known as *speedup-oriented multiprocessors* [DUB88].

This paper will focus on those parallel architectures that best fit the system requirements of an intelligent robot controller in terms of price versus performance. A robot controller is a large system that is composed of layers of fine-to-coarse-grained processes and can be characterized as a speedup-oriented multiprocessing application. Hierarchical structuring offers an easy and systematic parallel approach to partitioning. With a hierarchical control system, levels in the hierarchy can be developed as sequential processes, and then parallel integration can be modeled as communicating sequential processes [HOA78]. Thus, a control system can be developed functionally independent of the implementation and then hierarchically integrated with the communicating sequential processes model. Two design factors are at the

C-3

heart of exploiting the benefits of a hierarchical control system. The notions of "task decomposition" and "response time" are crucial in determining the number of processors in a hierarchical system, the complexity contained within any one level, and how communication should be performed.

*Task decomposition* can be defined as the process of recursively breaking down a task into smaller, more manageable tasks, until some atomic level of activity is reached. At each level in the hierarchical breakdown, an interface exists where the adjoining levels exchange information. The higher level communicates a command to its neighboring lower level. Likewise, the lower level command communicates a status to its neighboring upper level. This communication protocol is analogous in sequential programming to a subroutine invocation as a command and a return value as a status. Defining a hierarchical decomposition for a robot control system starts with a high level task, and through a series of task decompositions, reduces this task to a set of motion primitives. Concurrently, status of the environment filters up the hierarchy to provide feedback to the task decomposition.

Unlike sequential hierarchical decomposition, each lower level in a parallel pipelined architecture is not completely dependent on its neighboring upper level. Although levels may share some data that models the world, each level can be considered to run independently of any other, responding to a command and supplying a status much like a plant in a normal feedback control system. When composed as a system, hierarchical control is built as layers of *virtual control loops*. When executing, each virtual control layer in the hierarchy can be considered as part of a long chain defining the hierarchical state, yet each level's action is based on its own control flow. The virtual control loop software exhibits cyclic feedback behavior that samples inputs including command and status and guarantees some output within a given time.

Performing task decomposition is a function of both the state of the world and the number of operations that must be performed. Determining the amount of work a task must perform is a subjective issue but depends heavily on the amount of time required to make a decision. Responding to an event too late nullifies the control no matter how intelligent the subsequent action. This timing restriction leads to the definition of *response time* as the maximum allowable time duration between an event and an action resulting from that event.

Meeting a response time restriction may limit the amount of intelligent behavior at any one level. By dividing task decomposition into planners and executors, one can maintain real-time control yet concurrently evaluate alternative future actions. The *planner* is responsible for generating a plan consisting of a series of actions. An *executor* is responsible for stepping through a generated plan. The executor matches the current state of the machine against a set of preconditions, which triggers the corresponding action.

The planner and executor together compose the task decomposition module that determines the control behavior. But there are more functions to feedback control than just planning and executing. Supplying and interpreting external sensor information is another important function in feedback control. *Sensor processing* operates concurrently with the control function to interface to the real world. Sensor processing consists of data acquisition, filtering, enhancement, and other functions up to and including object recognition.

The disparity of purpose between control and sensing makes it useful to mediate the information exchanged between the control and sensing functions. The mediating function, defined as the *world model* [ALB81], allows the decision process to become more abstract in its nature of queries about the world, and less dependent on the physical nature of the actual sensors being used. The world model provides the system's best estimate and evaluation of the history, current state, and possible future states of the world. The world model provides accurate and current information for reflexive behavior (i.e. what is), the best estimate of the world in deciding the future plans (i.e. what if), and integrates the information from the sensor processing module. This leads to a partitioning of the system into hierarchical levels of virtual control loops, each of which consists of a control, a sensing, and a world modeling component. Figure 1 illustrates the partitioning of one level within the hierarchy.

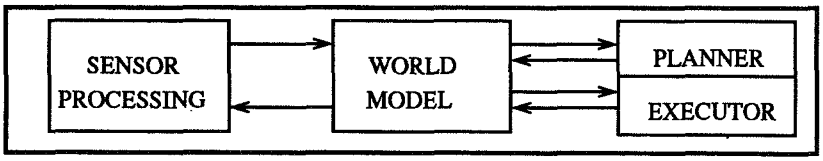


**Figure 1. Control Level Partitioning**

Further refining and partitioning of a control level is possible but is beyond the scope of this paper. Albus more thoroughly covers the decomposition and partitioning of a hierarchical control system [ALB87].

## 2.0 Programming Environment and Scheduling Requirements

A traditional real-time software development environment supports the development and testing of algorithms for logical correctness on a "slower" host machine, with later execution of these logically correct algorithms on a "faster" target system, to allow testing under real conditions. The goal of the host system is to offer as many development tools as possible. Target systems are used to meet the requirements of real-time response which implies a premium on efficiency. Assigning processes to processors is a major concern in multiple processor systems. The process of mapping processes to processors and scheduling tasks within these processes is iterative. Should the granularity chosen not realize the desired speed, additional processors may be required, or revision of the algorithm on the host system for subsequent use on the target system may be required. Because of this iterative nature, understanding the requirements of an efficient and robust programming environment will lead to an increase in software productivity.

## 2.1 Real-Time Target System Software

High data bandwidth, maximized throughput, and fast execution are desirable features of a real-time control system. To achieve this; target systems exhibit a tighter coupling between system support and the control algorithms. Sometimes, the operating systems of target systems are by design as primitive as possible so as to remove any unnecessary overhead [NAR86]. Efficient software performance for real-time systems has been determined to require several features including : 1) fast context switching time between tasks, 2) small interrupt latency, 3) feature priority scheduling, 4) allowing high-priority tasks to instantly pre-empt lower priority tasks, 5) methods for synchronization of tasks, and 6) granting low-priority tasks non-preemptable status.

Although speed is important in a real-time control system, reliability and deterministic behavior are the ultimate system measures that cannot be sacrificed at the expense of optimizing performance. Based on the deterministic criteria, a real-time system has a set of processing requirements that are different from a general purpose operating system. For instance, fairness is not an issue in real-time control. Either the processes are permanently dedicated to the hardware or are guaranteed resident during critical sections (i.e. non-preemptable). Another area affecting the deterministic behavior of the real-time model is processor scheduling. Time-sliced and round-robin scheduling algorithms result in nondeterministic performance and imply difficulty in verification. Exact time quantum round-robin scheduling can overcome the non-determinism, but the overhead of continual context switching overrides the use of the round-robin method except in very low speed applications. Instead, a priority-based system where processes execute in known sequences and surrender the processor willingly results in predictable, hence deterministic, behavior.

## 2.2 Host System Support Software

General purpose computer operating systems provide a robust set of programming tools, such as a wide variety of compilers, editors, source code control tools, and many tools ideally suited for software development. General purpose, multi-user computers are not useful as target systems because of the unpredictability of the underlying operating system. With so many responsibilities to handle concurrently, large general purpose operating systems are not generally designed to include features required for running in real-time. Therefore, these are separated and an efficient communication link (such as a local area network) from the real-time target system to the host development system is not strictly required but greatly improves system flexibility.

## 2.3 Analysis of System Software Requirements

The programming environment for a hierarchical real-time control system is comprised of both host and target system needs. The necessity to divide the software system into two components is based on the need to meet timing requirements. The distinction between the two components becomes hazy as one moves up the hierarchy until eventually the distinction disappears.

At the very lowest levels, the functional components must be efficient and highly streamlined since parts of the control system may have to run without the luxury of any operating system assistance. A low-level servo controller cannot afford the time to allow multi-tasking or other system overhead. A kernel which handles spurious interrupts, limited background I/O service, and a basic monitor for off-line board level troubleshooting is sufficient for streamlined system support. However, stand-alone levels still require interprocessor communication to other levels in the hierarchy.

Higher levels are allowed longer processing intervals so that context switching using multi-tasking among different processes can be performed. Further, higher levels may require a high-speed file system for data logging and performance analysis that can be used as an evaluation tool or as a postmortem data recorder box (or "black box") after an unforeseen crash of the system.

This review of system requirements leads to the observation that a computing system with a rich set of software tools is necessary to handle the broad variety of processing needs ranging from the real-time hierarchy to supporting the user interface. A prototype architecture for such a system would exhibit differing degrees of concurrency that then can be implemented as a blend of multiple processors and interleaved execution on a single processor. Because of the disparity of response times required at various levels in the hierarchy, different levels of granularity are required. At the lowest levels, planning may be impossible, and even execution may require multiple processors to achieve a solution. Moving higher in the hierarchy, timing constraints prevent the planner and executor from residing on the same processor. In this case, the two processors would run in parallel, and asynchronously the planner would update plans. At higher levels, completion time for plans is less critical so that multitasking on a single processor can supply enough processing power for both the planner and executor. If the timing constraints are not stringent, multiple levels can be combined onto one processor, or the use of a LAN can be used to connect other computers as a part of the controller. Each of these configurations is based on the response time requirement of the level. Figure 2 provides a guideline for the type of performance required of differing levels in the hierarchy. This is based on the premise of allocating a nominally fixed percentage of response time devoted to communications versus computation for each level of the hierarchy. The actual times chosen are relative, and will differ according to one's choice of hardware and operating system support. This percentage of time concept and the problem of timing analysis for a hierarchical control system is more thoroughly treated in [MIC88].

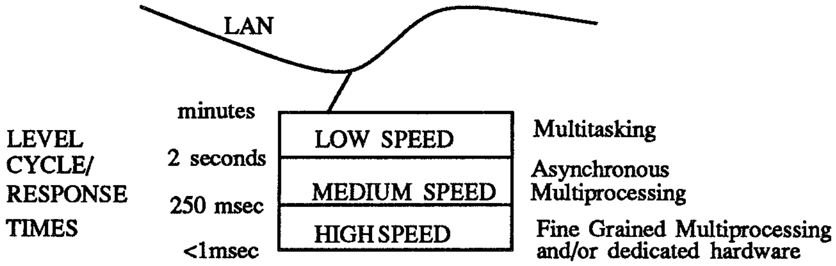


**Figure 2. Computing Resource Utilization Based on Cycle Response Time**

## 3.0 Synchronization and Communication

Synchronization is a basic building block that a multiprocessor system must contain. Synchronization serves the dual purpose of enforcing the correct sequencing of processes and ensuring the mutually exclusive access to certain shared, writable data. Synchronization is usually supported by some special-purpose hardware. These basic synchronization primitives are then used to build higher level synchronization and communication tools in software or microcode. Communication and synchronization are difficult to separate because communication primitives can be used to implement synchronization protocols and vice versa. In general, communication can be defined as transferring or exchanging information between processes, whereas synchronization is a special form of communication in which the data is control information.

### 3.1 Synchronization

Synchronization provides mechanisms for coordinating *control flow* and *data flow* between multiple processes. Synchronization tools evolve from basic hardware primitives to complex software and microcode schemes. Parallel machines include hardware primitives capable of enforcing mutual exclusion on a resource. Semaphores and message passing are software extensions built upon these primitives. For control flow, parallel processes are synchronized to wait until all processes are ready. For data flow, synchronization limits access to a shared resource to one process at a time. This section will cover synchronization of control flow, while the following section on communication will cover synchronization of data flow.

At the user programming level, control synchronization between the master routine and the slave subroutine in sequential processing is guaranteed because the master waits until the slave is done before resuming execution. In concurrent machines, the master could continue processing or wait for a rendezvous at a later point in time. In parallel programming, the concept of coroutines appear as a more fundamental synchronization structure than subroutines, which can be regarded as a special case [HOA78]. Synchronization in a parallel machine must handle the problem of parallel processes finishing at different times, waiting for all to complete (synchronizing), and then continuing.

Synchronization schemes are based on some indivisible hardware action limited to only one process at a time. For example, a basic semaphore consisting of a test-and-set can be used to mediate requests. These primitive operations are subsequently used as the building blocks for higher level multiprocessing synchronization mechanisms such as *monitors*, *semaphores*, *single-writer/multiple-read blocks*, *master/slave* and *critical sections*. *Priority-level* access is a further enhancement of synchronization. A *barrier synchronization* function is a software or microcoded synchronization tool placed so that processes arriving at the barrier must wait until all pertinent processes arrive. Processes either busy-wait or block until the barrier is lifted. *Global broadcast synchronization* uses some signal to establish the beginning of a time period in which all processes across all boards in the system are allowed to write to their common memory buffers [ALB81]. Global broadcast synchronization is best suited where the standard deviation between cycles is small so that updates can occur at a known intervals with little processing time wasted.

### 3.2 Communication

Interprocessor communication can be characterized as a read-write sequence with two additional facets, synchronization for the exchange of information and destructive/non-destructive read/write execution. For example, writing to a queue is non-destructive (assuming enough storage), while removing it is destructive. Writing to a variable is destructive,while reading it is non-destructive. Figure 3 summarizes the combinations of non-destructive and destructive execution during communication and the styles of communication that result.

| | Destructive Write | Non-Destructive Write |
|---|---|---|
| Destructive Read | non-queued message passing | queued message-passing |
| Non-destructive Read | variables, shared memory | mail (assumes explicit deletemechanism) |

**Figure 3. Styles of Communication**

The execution may also have a time-out feature, so that the process does not wait indefinitely for new information. The information exchange can be synchronous or asynchronous. The following sections more closely explore the methodologies and rationale of different communication techniques.

### 3.2.1 Evaluating Communication

To evaluate the effectiveness of interprocessor communication, the performance measures of latency and throughput are used. To characterize a robot control system as real-time implies a guaranteed maximum latency for interprocessor communication. *Latency* is defined as the elapsed time before a message is acknowledged. *Throughput* is defined as the number of bytes one process can send to another process in one second, especially important for systems that share large amounts of data. However, 20 bytes of information with a 20 msec update rate cannot be handled with a communication protocol that is efficient for moving large amounts of data but which requires a 100 msec setup time.

Additionally, the concepts of flexibility, data integrity, and extensibility are as equally important in evaluating a communication scheme but are more intangible. Data integrity and flexibility can best be explained with an example. On sequential machines, interprocess communication is performed by calling or invoking a subroutine and passing the appropriate parameters on the stack by value or by sharing global variables. How the parameters are exchanged controls the method of communication. Passing parameters "by value" places a copy of the parameter onto the stack and can be considered message passing since each processes' internal variables are independent of the other processes. Passing parameters "by reference" pushes a pointer or address onto the stack so that data is shared between the processes. In a uniprocessing architecture, passing parameters "by reference" saves excessive copying. However, in parallel processing such copying is more flexible since a user cannot be assured that the two processes both have access to the pointers or addresses, and that the addresses are valid (in a dual ported memory scheme, on-board versus off-board addresses differ). However, complete copying is slower and this highlights the fact that a robot control system must address the issues of flexibility and data integrity in conjunction with performance measures. This coupling of evaluation leads to numerous communication design alternatives aimed at meeting different priorities.

### 3.2.2 Communication Designs

Different designs are available in order to perform real-time communication. Gauthier, et al provide an in depth survey of the various interprocessor communication designs [GAU87]. Overall, communication depends on the system architecture, either tightly or loosely coupled. Tightly coupled systems have a public memory architecture, where processors communicate through *shared memory*. Loosely coupled systems have private memories where processors generally cannot read each others memory and must have an explicit *message-passing* communication channel.

*Message passing* systems have memory attached privately to each processor (at least conceptually), so that processors communicate only through explicit transmissions of whole messages [LYO87]. Message passing is more generic in that messages can be passed not only across backplanes but across machines. Message passing is typically slower, as it is generally implemented at the subroutine level and requires coordination at the receiving end.

*Shared memory* is a special form of message passing, that provides equal access to all processors. Shared-memory offers many advantages, including ease of sharing data, rapid communication, and high performance. The advantages that make shared-memory architectures attractive result from the tight coupling along the critical path between processors and memory. This leads to a notable increase in performance. Shared memory offers such a highly efficient and straightforward method for communicating among parallel processes that it is commonly used for parts of real-time systems [PAU86, KOR86, KAZ87]. Assuming a shared memory communication scheme, the disadvantages that arise are: 1) guaranteeing mutual exclusion of shared resources across processors and 2) maintaining a consistent view of addressing across processors. Another major disadvantage of shared memory is that as the number of processors grow, the arbitration of accesses to the common memory burdens the system and degrades performance. The use of multiple communication links can ease this but increases the problem of internal versus external addressing for common memory mapping.

### 3.2.3 Communication Synchronization : Asynchronous vs Synchronous

In general, communication mechanisms can be characterized as either synchronous (blocking) or asynchronous (non-blocking). Routinely, blocking implies surrendering the processor and waiting for some condition to occur before continuing execution. Whether to block provides the fundamental distinction between a synchronous "lock-step" communication exchange and an asynchronous "free-wheeling" communication exchange. Synchronous communication uses an explicit handshake for acknowledgment. In an asynchronous communication protocol, the sending process does not have to wait or block for the receiving process to acknowledge receipt of the message. Since asynchronous messages can arrive at random times, queues are attached to save messages. Acknowledgment is optional, but can be installed as part of the mechanism. The message itself must contain instructions whether an acknowledgment is necessary.

One advantage of synchronous communication is simplicity (no queues or queuing monitor) and thus low overhead. Another advantage is the savings in space and time that result because data can be directly copied from the sending processes' buffer to the receivers with no intermediate storage action required. Finally, the one-to-one correspondence of a synchronous transmission provides a stricter notion of accountability and determinism that enhances software reliability. The major disadvantage of synchronous communication is the time spent synchronizing and acknowledging, i.e. waiting, for each communication. Another disadvantage is the lack of flexibility and subsequent extra software penalty for handling dynamic reconfigurations such as many-to-one communication channels. For example, in a dynamic environment, a synchronous communication exchange requires a priori knowledge by both parties of the other's existence in order to synchronize.

The advantages of asynchronous communication are speed and flexibility. The lack of synchronization and acknowledgment steps improves performance. The disadvantage is that error detection may be overlooked. Flexibility results in that the message may embody more of the communication mechanism, i.e. destination of an acknowledgment. With information embedded in the message rather than the a priori synchronous connection, a new client can be dynamically added to a server by including a destination address within the message. The disadvantages of asynchronous communication is the lack of accountability for errors which requires programmer discipline to foresee and handle errors.

These two models are basically equivalent in that the synchronization (i.e. control information exchange) before the communication can be considered a bi-directional asynchronous communication. Further, asynchronous communication is usually extended to include an acknowledgment step. In addition, synchronous communication can use a table lookup for dynamic reconfiguration. From a software validation standpoint, synchronous communication is preferred, but the system may not tolerate the extra amount of overhead. In general, for connections that are statically predefined one-to-one exchanges, synchronous communication provides the most reliable scheme. For connections mapped as many-to-one and dynamically reconfigurable, asynchronous communication provides the cleaner implementation.

### 3.2.4 Communication Duration : Connection Ties

Communication connection ties can be either temporary (datagrams), or permanent (virtual circuits) [POS80]. The communication ties can be established dynamically through a system broadcast [FRI86] or by statically defined connections [NAR86]. A dynamic connection is established by the sending process broadcasting a system-wide request for the receiver location, who responds in turn with its location. Communication is then direct. Dynamic connections offer flexibility, but should be done upon system start-up for permanent communication links. Dynamic connection for temporary links would not be useful for a system with stringent real-time control demands. Static connections use tables to link sending and receiving processes. Static connections are fast, but may require the overhead of some centralized server.

## 4.0 Analysis of Design Alternatives

Given the rich set of communication design strategies, some general heuristics assist in focusing the communication design. A survey of existing multiple processor robot control systems reveals a wide range of implementations. The requirement to satisfy real-time constraints has lead to numerous implementations using tightly coupled architectures with several processor boards on a common bus [PAU86, NAR86, KOR86, KAZ87, GAU87, SCH87]. The target operating system of these implementations vary in degree of system complexity from the basic use of interrupts and handshaking operations to handle interprocessor communication[PAU86], to a limited operating system with special communication features [NAR86], to an extended real-time operating system [SCH85].

These implementations share common system architectural characteristics with a hierarchical control system. Although few of these control systems are directly labeled as hierarchical, most systems have at least a two-level hierarchy with a low-level, real-time subsystem handling real-time motion control and a slower high-level subsystem responsible for planning. This basic hierarchical decomposition in each of these applications can be generalized into two virtual control loops exchanging commands and status. Implicitly, these applications have additional levels within the hierarchy, but these levels do not directly correspond to a virtual control loop implementation because of the concept that returning status is implicit in a serial execution of the processes. These implicit levels could be partitioned into separate concurrent processes that communicate through established interfaces rather than serial routines that communicate via subroutine calls. Further, with concurrent operation, the concept of sampling the return status rather than just receiving a final status embodies the feedback nature of a virtual control loop.

The concurrent hierarchical model of a robot control system containing virtual control loops has been implemented with a purely executor style of task decomposition for a robot control system at the National Institute of Standards and Technology [BAR82]. The flow of control was based on state-table transitions. Where planning was appropriate, static plan definitions were used. The system offered several benefits. First, the system was sensory-interactive and adapted to perturbations in the environment in real-time even though the world model was limited to basic feature recognition. Second, hierarchical decomposition created well-defined interfaces that allowed substitution of different implementations of a level without major impact on the higher or lower levels that lead to proposed interface standards [FITZ85]. Finally, the system was able to execute in real-time and supply robot updates within a fixed interval on the order of milliseconds.

Table 1 summarizes some of the advantages and disadvantages of the previously discussed communication features. Double lines separate independent communication features, from which one of the alternative designs must be chosen. Sections between the single lines demarcate the selections for each feature.

Although one of the communication features must be chosen for a given level, it generally does not constrict one to that choice for the entire system. Indeed, a very flexible system would incorporate some of each of these using the response time criteria as an initial guide to the selection.

| Issue | Feature | ADVANTAGES | DISADVANTAGES |
|---|---|---|---|
| Coupling | Tightly Coupled<br>Shared Memory | high-speed,<br>simple | Rarely portable, interconnect limits, contention problems |
| | Loosely Coupled/<br>Message Passing | Flexible, generic<br>better security,<br>cheaper for larger number of processors | Slower, limited data bandwidth |
| Synchronization | Synchronous Comm. | Simple, guaranteed comm. | Slow, requires time-out |
| | Asynchronous Comm. | Flexible, fast | Overhead - queuing |
| Duration | Temporary Channel | multiplex resources | repeated overhead |
| | Permanent Channel | low overhead | 1:1 communication best |
| Connectivity | Dynamic connectivity | flexible, reconfigurable | slower, larger overhead,<br>special hardware |
| | Static connectivity | simple, fast | centralized server necessary |
| | Centralized connectivity | better accountability | extra decode overhead each message |
| | Decentralized connectivity | unbundled, demon model | slower, resource intensive |

**Table 1. Communication Feature Summary**

The algorithmic view of data shared between processes also affects the communication interfaces. The most straightforward algorithm is double buffering using shared memory where only one reader and one writer grab complete control of the common memory until finished. A more complicated algorithm (n readers, one writer ) allows several readers access to the memory at once, but only one writer at a time. For an example of an n-readers, 1-writer algorithm, consider the case were two levels in the hierarchy are communicating command and status to each other. Not only could one level read a command, but a diagnostic process and a data performance logger could also sample the command buffer with minimal overhead.

The client/server model is a widely used communication protocol. This method fits into a message passing type of interface. The server offers a level of abstraction from the user (i.e. the client) who queries the server. The server hides the client from the physical implementation details and allows the level of discourse to be of a logical and abstract nature. The client/server offers the strongest rationale for using a message passing scheme (even if it is simply a subroutine call). Within the control hierarchy, the world model acts as a server to its clients, the planner and the executor. Clients request service from the world model. If the world model is busy, this request is queued. When ready, a client is serviced by the world model. The world model translates all logical queries concerning the system into a corresponding physical representation and responds with an answer. Thus, the world model shields its clients from understanding its physical representation of the world.

Given these two communication interfaces, a general rule applies :

*When one process has knowledge of the data structure used by the other processes, then access is direct. If the processes use different data structures or one views the data logically while the other handles physical implementation, a server is necessary in order to communicate information.*

The amount of data moved between processes is another factor in the design. Moving large amounts of data or shifting processes to and from private processor memories can be cumbersome and slow. A shared memory scheme is more appropriate here since attempting to use a message passing technique as a server to mediate shared data can be much slower, by a factor of thirty, than direct access [KAZ87]. Thus, if the data structure is conceptually consistent across processors, shared memory improves performance.

## 5.0 Conclusion

This paper has addressed some design issues associated with developing a hierarchical control system for an intelligent machine composed of a broad set of functional requirements. This set of requirements is too large to handle in a conventional sequential programming environment because of the stringent timing constraint imposed on performance. The use of a hierarchical structuring of modules for a control system creates levels that operate in parallel and can be characterized as virtual control loops. Virtual control loops use software to emulate a feedback control loop by sampling as inputs the command from a neighboring upper level, comparing the input to the sensory sampled environment, computing a goal-directed action, and outputting this action to the neighboring lower level, all within a fixed response time. Mapping this hierarchical control system onto a parallel pipelined computing system is a reasonable method of implementation. It offers flexibility and reliability at a reasonable cost. Work has begun on the design and implementation of a concurrent hierarchical control system that includes planning, extensive world modeling, and high level sensory processing. This system will use many of the aforementioned communication features in a hybrid approach. The concept of the response time dictating the choice of communication schemes (based on a percentage of the response time devoted to communication) will be further explored as a possible aid to limit the iterative process of allocating processes to processors. By using a systematic ordering by time of the various communication primitives available, one can quickly determine the type of system support appropriate for a level, given the response time of that level. The discussion is beyond the scope of this paper, and will be the topic of future publications by the authors.

## References

[ALB81] ALBUS, J.S., BARBERA, A.J., NAGEL, R.N., "Theory and Practice of Hierarchical Control", *Twenty-third IEEE Computer Society International Conference*, 1981, pp.18-39.

[ALB87] ALBUS J.S., McCAIN H.G., AND LUMIA R., "NASA/NBS Standard Reference Model For Telerobot Control System Architecture (NASREM)", NBS Technical Note 1235, National Bureau of Standards, Gaithersburg, Md., July 1987.

[BAR82] BARBERA, A.J., FITZGERALD, M.L., AND ALBUS, J.S., "Concepts for a Real-Time Sensory-Interactive Control System Architecture", *Proceedings of the 14th Southeastern Symposium on System Theory*, April 1982.

[DUB88] DUBOIS, M., SCHEURICH, C. AND BRIGGS, F.A., "Synchronization, Coherence, and Event Ordering in Multiprocessors", *IEEE Computer*, Februrary 1988, pp. 9-21.

[FITZ85] FITZGERALD, M.L., BARBERA, A.J., "A Low-Level Control Interface for Robot Manipulators", NBS-Navy NAV/SIM Workshop of Robots Standards, June 6-7, 1985.

[FRI86] FRIEDLANDER, C.B., AND WEDDE, H.F., "Distributed Processing Under the Dragon Slayer Operating System".

[GAG86] GAGLIANELLO, R.D., AND KATSEFF, H.P., "A Distributed Computing Environment for Robotics", *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 1986, pp. 1890-1895.

[GAU87] GAUTHIER, D., FREEDMAN, P., CARAYANNIS, G., AND MALOWANY, A.S., "Interprocess Communication for Distributed Robotics", IEEE Journal of Robotics and Automation, Vol. RA03, No. 6, Dec. 1987, pp. 493-504.

[HOA78] HOARE, C. R., "Communicating Sequential Processes", *Communications of the ACM*, Vol. 21, No. 8, August 1978, pp. 666-677.

[KAZ87] KAZANZIDES P., WASTI H., AND WOLOVICH W.A., "A Multiprocessor System for Real-Time Robotic Control: Design and Applications", *Proceedings of the IEEE International Conference on Robotics and Automation*, New York, NY, 1987, pp. 1903-1908.

[KOR86] KOREIN, J.U., MAIER, G.E., TAYLOR, R.H., AND DURFEE, L.F., "A Configurable System for Automation Programming and Control", *Proceedings of the IEEE International Conference on Robotics and Automation,* San Francisco, CA, April 1986, pp. 1871-1877.

[KRU87] KRUSKAL, C.P., SMITH C.H., "On the Notion of Granularity", National Bureau of Standards Report, July 1987.

[LYO86] LYON, G., "Programming The Parallel Processor", Second Symposium on the Role of Language in Problem Solving, Applied Physics Laboratory of Johns Hopkins University, April 2-4, 1986.

[LYO87] LYON, G., "On Parallel Processing Benchmarks", National Bureau of Standards Report , NBSIR 87-3580, June 1987.

[MIC88] MICHALOSKI, J.L., WHEATLEY, T.E., AND LUMIA, R. "Exploiting Computational Parallelism with a Hierarchical Robot Control System", to be published.

[NAR86] NARASIMHAN, S., SIEGEL, D., HOLLERBACH, J.M. , BIGGERS, K., AND GERPHEIDE, G., "Implementation of Control Methodologies on the Computational Architecture of the Utah/MIT hand", *Proceedings of the IEEE International Conference on Robotics and Automation,* San Francisco, CA, April 1986, pp. 1884-1889.

[PAU86] PAUL, R.P, AND ZHANG, H., "Design of a Robot Force/Motion Server", *Proceedings of the IEEE International Conference on Robotics and Automation,* San Francisco, CA, April 1986, pp. 1878-1883.

[POS80] POSTEL, J., "Internetwork Protocol Approached", IEEE Transactions on Communications, Vol. COM-28, Number 4, April 1980, pp. 604-611.

[SCH85] SCHWAN, K, BIHARI,T, WEIDE, B, AND TAULBEE, G., "GEM: Operating System Primitives for Robots and Real-Time Control Systems", *Proceedings of the IEEE International Conference on Robotics and Automation*, 1985, pp. 807-813.

[SCH87] SCHWAN, K, BIHARI,T, WEIDE, B, AND TAULBEE, G., "High-Performance Operating System Primitives for Robotics and Real-Time Control Systems", *ACM Transactions on Computer Systems*, Vol. 5, No. 3, August 1987, pp. 189-231.

N90-29892

# The JPL Telerobot Manipulator Control and Mechanization Subsystem (MCM)

S. Hayati, T. Lee, K. Tso, P. Backes, E. Kan
Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California, U.S.A.

J. Lloyd, McGill University, Montréal, Québec, Canada

## Abstract

*The Manipulator Control and Mechanization (MCM) subsystem of the JPL/NASA Telerobot system provides the real-time control of the robot manipulators in autonomous and teleoperated modes and real-time I/O for a variety of sensors and actuators. Substantial hardware and software are included in this subsystem which interfaces in the hierarchy of the Telerobot System with the other subsystems. These other subsystems are Run Time Control (RTC), Task Planning and Reasoning (TPR), Sensing and Perception (S&P), and Operator Control subsystem (OCS) (see Figure 1). This paper describes the architecture of the MCM subsystem, its capabilities, and details of various hardware and software elements. Important improvements in the MCM subsystem over the first version [1] are dual arm coordinated trajectory generation and control, addition of integrated teleoperation, shared control capability, replacement of the Unimate controllers with JPL-built motor controllers [2], and substantial increase in real-time processing capability.*

## 1 Introduction

The Jet Propulsion Laboratory, National Aeronautics and Space Administration (JPL/NASA) telerobot program began in 1985 [3]. The goal of this program is to develop a telerobot system that has the capability to perform typical space-related robotics activities under human supervision. It is understood that the system will initially have more capabilities in teleoperation and as it develops, more autonomous capabilities will be realized. This telerobot system consists of five subsystems [4]. Since each subsystem is specialized in one particular area of robotics, and each subsystem will in general evolve based on current technology, no rigid guidelines were imposed on the subsystems to utilize particular hardware or software. It was determined in the early stages that as long as all the subsystems could communicate with each other on a common protocol, then they could develop at their own pace.

Figure 1 shows a schematic drawing of the overall system. Two paths from the operator to the robots are recognizable in this figure: the Teleoperation path and the Autonomous path. In

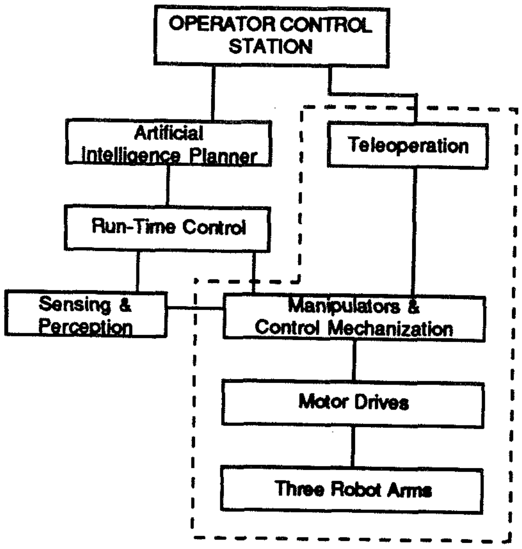


Figure 1: JPL System Architecture

the Teleoperation path, the operator controls the robot arms using the six-degree-of-freedom force reflecting hand controllers (FRHC) [5]; minimal interaction with other subsystems is required. In the autonomous path, the operator can perform supervisory control by issuing high level commands via the TPR and the RTC subsystems to invoke certain general *macros* resident in the MCM subsystem.

The scenario of the operation of this telerobot system is a 'mixed' mode of operation. As an example, assume that the operator needs to perform a bolt removal. She tells the TPR subsystem that she needs to operate one of the arms in the teleoperation mode. This command is then channeled through the RTC subsystem to the MCM subsystem instructing it to operate in teleoperation mode. As a result of this action, now both the TPR and RTC are aware that the environment and the position of the robots might change. They "listen" to the MCM subsystem as the teleoperation continues so that they can "watch" over the shoulders of the operator making sure that she does not move the arms to some forbidden area based on their a priori knowledge of the environment. She then uses the TPR subsystem to request the S&P subsystem to locate the position of the bolt using a designation and fit mechanism [7,8]. This process enables the operator to locate the position of the bolt head to within a few mm. At this time the operator can request the RTC subsystem to initiate a relative calibration process with the S&P subsystem. The S&P subsystem then uses the *camera arm* to look at the robot end-effector and the bolthead to locate more precisely their relative displacement. This process eliminates the effects of the absolute positioning errors of the manipulator arms. The operator then issues a high level command for bolt removal. The TPR subsystem breaks this command into several subcommands which are sent to the RTC subsystem and RTC issues appropriate commands to the MCM subsystem to position the arm above the bolt and then to execute the MCM bolt removal macro. MCM macros are written in a general way so one macro can be used for a number of geometrically similar objects and tasks. The RTC subsystem also checks all possible joint limit violations before it sends a command to the MCM subsystem. MCM removes the bolt and reports the success or failure status to the higher level systems. The MCM subsystem is responsible for providing the following basic manipulation functions:

- autonomous and teleoperated control of two arms,
- force reflection to the hand controller and display to the operator,
- execution of free or guarded motions,
- execution of macro tasks capable of accepting different parameter sets,
- frequent trading of control between the teleoperated and autonomous task executions, and
- shared control.

In section 2 we provide the basic architecture of the MCM subsystem. Section 3 describes the new multi-arm RCCL and its port to a Sun 4/200 computer. Section 4 details the teleoperation and shared control portions of the system and conclusions are given in section 5.

## 2 MCM Architecture

The MCM subsystem is a real-time system which controls the robots and the hand controllers, acquires sensor data for control and communicates with other subsystems. Guidelines for the design of this subsystem are as follows:

- MCM's computational architecture must clearly distinguish between the *local* and *remote* sites. The *local* is where the human operates the system and *remote* is where the manipulators operate. In principle, these two sites can be only a few feet or hundreds of miles apart.
- The architecture should allow for future expansions of the system's computing power and addition of more peripherals.
- The architecture must use the same remote computational environment to operate the system in either teleoperation, autonomous, or shared control modes.
- System software must be flexible to add/modify new basic functionalities such as a user-defined trajectory generator. This feature is required to design shared control [6] capabilities so the operator and the autonomous system can *share* in the execution of various tasks.
- The software must be as arm independent as possible.
- The system must allow for rapid software development both at the system and user levels.
- The computing environment (i.e., hardware and software) must allow the system to migrate to new faster processors as they become available without major rewrite of the software.

Figure 2 shows a block diagram representation of the MCM subsystem. The system is physically divided into two sites: local and remote. In the local site where the operator is stationed, there are two identical systems (left and right) each consisting of a hand controller (FRHC), a VME chassis with 68020 processors for computations of kinematics and force reflection, an RGB monitor for force/torque display, a JPL-built Universal Motor Controller (UMC), and a Sun computer for the

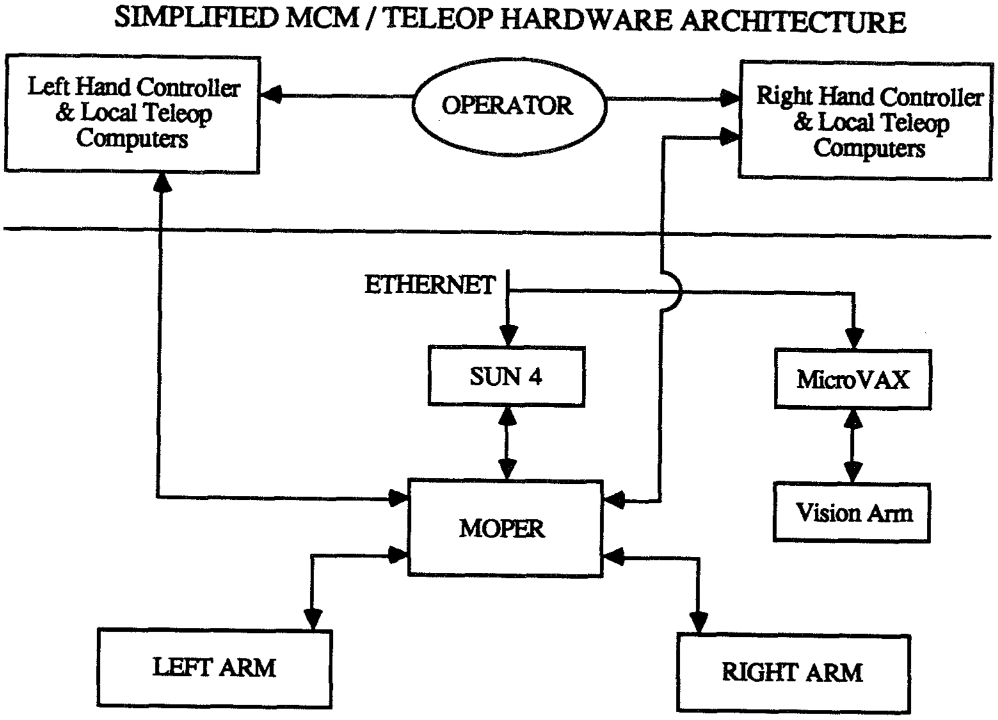

SIMPLIFIED MCM / TELEOP HARDWARE ARCHITECTURE
Left Hand Controller & Local Teleop Computers
OPERATOR
Right Hand Controller & Local Teleop Computers
ETHERNET
SUN 4
MicroVAX
Vision Arm
MOPER
LEFT ARM
RIGHT ARM

Figure 2: High-Level MCM Architecture

operator interface. The remote site consists of a Sun 4/200 computer to run the enhanced version of multi-arm RCCL [9,10,11] which is capable of generating trajectories for multiple coordinated manipulators. This is used to run two PUMA 560's. A MicroVax II which runs the same enhanced RCCL is used to move a third Puma 560 for pointing a pair of cameras for the S&P subsystem. The reason for not using the Sun 4 to run this third arm is the unavailability of one additional UMC rather than any limitation in the new Sun 4-based RCCL software. Each arm is equipped with a Lord wrist force/torque sensor, servoed gripper from TRI Inc., and the Unimation teach pendant which operates under RCCL. Signals from local hand controllers, force/torque sensors, servoed grippers, and teach pendant all are read and passed to the Sun 4 through the MOPER. This chassis is equipped with four processors[1] and several parallel and serial communication cards. Figure 3 shows a block diagram representation of the MCM hardware and their connections.

## 2.1 Real Time Operations

The system works based on interrupts generated by a processor card in the MOPER chassis. This regular interrupt causes the MOPER to gather information from the UMC's (i.e., the state of the arms such as joint encoder values, joint potentiometers, and the status of the motor power), the incremental motion of the hand controllers (either in Cartesian or joint form), and force/torque sensor information. This data is then transmitted to the SUN 4/200 via a shared memory card (BIT 3). This data is used in RCCL to check for joint or velocity or some other user-defined limit violations.

[1] All processor cards except the ones in the UMC's are single board computers base on Motorola 68020 microprocessor and 68881 floating point co-processor with 1 meg. RAM. The development environment is VxWorks with the Wind kernel.

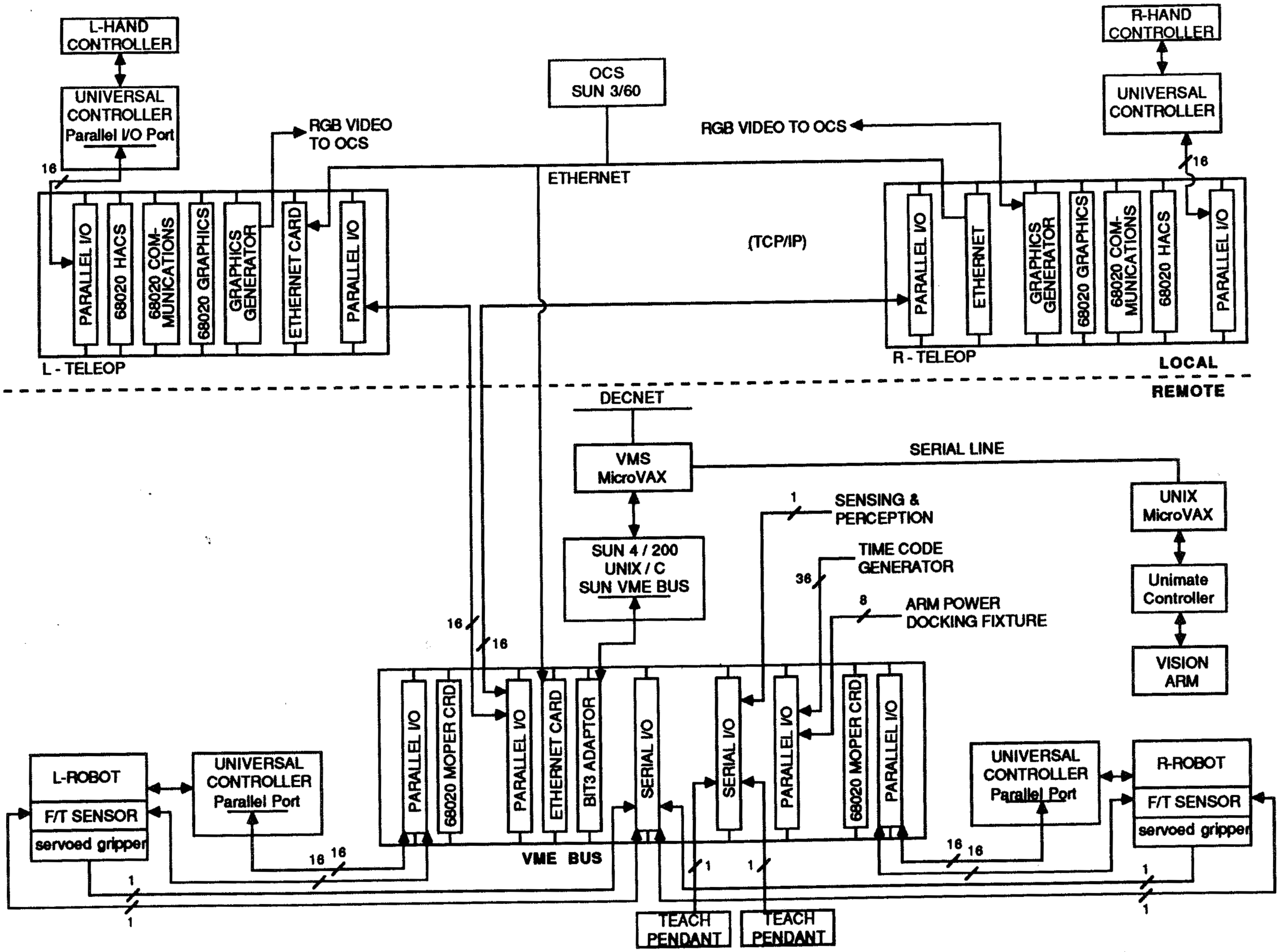


Figure 3: MCM Hardware Block Diagram

Since RCCL allows the trajectories to be modified in real time, this data is utilized to modify the nominal trajectory to provide sensory-based motion. During this time interval, MOPER transmits force/torque sensor information to the local teleop chassis for Kinesthetic and visual feedback to the operator. It also receives commands from the Sun 4 and after interpretation sends appropriate commands to the UMC's to move the robot arms. In addition to the I/O function, MOPER generates the main timing interrupts to the system. This means the Sun 4, which runs the RCCL code as well as the local teleoperation chassis, operates in the slave mode. This system heartbeat can be increased or decreased based on the complexity of the user written software in RCCL. Unlike the Unimate controller restriction where one is limited to sampling time intervals of 7, 14, 28, 56 msec. etc., the sample rate can be any value with a resolution of fractions of msec.

Note that the system is designed such that at every time interval, all the information including the teleop is available in the Sun 4/200 RCCL environment. This makes it very easy to implement traded and shared control. By traded control we mean a mechanism by which an operator switches back and forth between autonomous and teleoperated modes. Shared control refers to a 'mix' mode of operation where certain directions are controlled by the operator and the remaining ones by the autonomous system. This applies both for position and force subspaces [6].

## 2.2 MCM Executive

The remote portion of the MCM system is utilized in two modes of operations: Stand alone and Integrated modes. In the former, one writes C programs in the RCCL environment and executes tasks. This mode is used to develop macros, test and package them so the operator can call them in the supervisory (autonomous) mode of operation from higher levels. In the integrated mode, the MCM executive receives commands from RTC using a communication software called Network Interface Package (NIP) developed at the Stanford Research Institute (SRI). NIP is a layer above DECNET which uses Ethernet hardware to communicate between various subsystems. When a NIP message arrives at the MCM executive, it determines the destination of the command and then directs the data to the appropriate hardware (three puma arms, and two servoed grippers). The MCM executive can receive several commands and queue those that require the same hardware. The higher level systems can always issue cancel commands which will be executed immediately and the queue will be flushed. To make sure that adequate information is provided to the higher subsystems, MCM reports the status of the system approximately once a second.

The NIP software only runs under the VMS operating system. Since the MCM subsystem is Unix based, the NIP commands must first be received by a *gateway* MicroVax running the VMS operating system and then transmitted via DECNET to the SUN 4 computer. The MCM executive can receive its commands either in the form of NIP from the network or from an operator interface provided by the MCM executive. The format of this interactive interface is very similar to MATLAB and is an *extension* to ARC (A Robot Calculator) supported by the new RCCL. ARC is an interactive scalar and matrix calculator which has access to RCCL's numerous robot specific library functions (such as Jacobian, forward kinematics and so on). The extension to ARC attaches it to the actual robots so an analyst can interactively perform certain analysis and then try the results on the robots without compiling or leaving the analysis environment.

# 3 Multi-Arm RCCL: Multiple Robots and Processors

RCCL (Robot Control C Library) was originally developed at Purdue University [9] and later was improved at McGill University [12]. It was later ported to a MicroVax II [10] and used at JPL, RCA Corporation and McGill University. This implementation still lacked several important features that were needed by the telerobot project's requirements, i.e.,

1. Multi-arm coordination and control capability
2. Adequate computing speed
3. State-of-the-art hardware and software environment for sensor integration
4. Robot independence

The new multi-arm RCCL now implemented has overcome these deficiencies.

## 3.1 Coordination of Multiple Robots

The original RCCL was designed to program only one arm. Several basic dual arm capabilities are needed to use two or three robots in a convenient manner. These are: independent, synchronized, and coordinated operations. Execution of a single task could easily use all three modes of operations. For example, assume that a pair of robots are utilized to unbolt several bolts from a panel and move it to a different location. If the tool-crib is only reachable by one of the robots, then one robot must pick up a tool and transfer it to the second one. This will require synchronization between the robots. At this point both robots can independently execute unbolting operations. After the bolts are removed the robots must transfer the panel by operating in the coordinated mode.

Coordinated motion requires two Cartesian trajectory generators to realize the trajectories such that certain kinematic constraints are satisfied at all times. To do this, the kinematic ring equation structure was generalized to allow for expressing the kinematic relations for an object which is not a robot [11]. The ring equations for each robot can be kinematically constrained to be attached to this object. When the object moves, so do the robots. The forces of interaction caused by uncertainties in the Denavit Hartenberg [15] parameters and positioning control errors are accommodated by including *COMPLY* transforms which are modified by control laws in functions using force/torque sensor data. Figure 4 shows the planning and control stages of the multi-arm RCCL system.

## 3.2 Multi-Processor RCCL

Computing power is one of the main limitations both for single and multiple robot control environments. RCCL operates in two levels. The planning level processes the user written code and queues motions for the control level to process. The Control level which provides a real time environment for set point computation is called RCI (Robot Control Interface). The planning level runs in the background (asynchronous) whenever the CPU is available. The control level is that portion of the

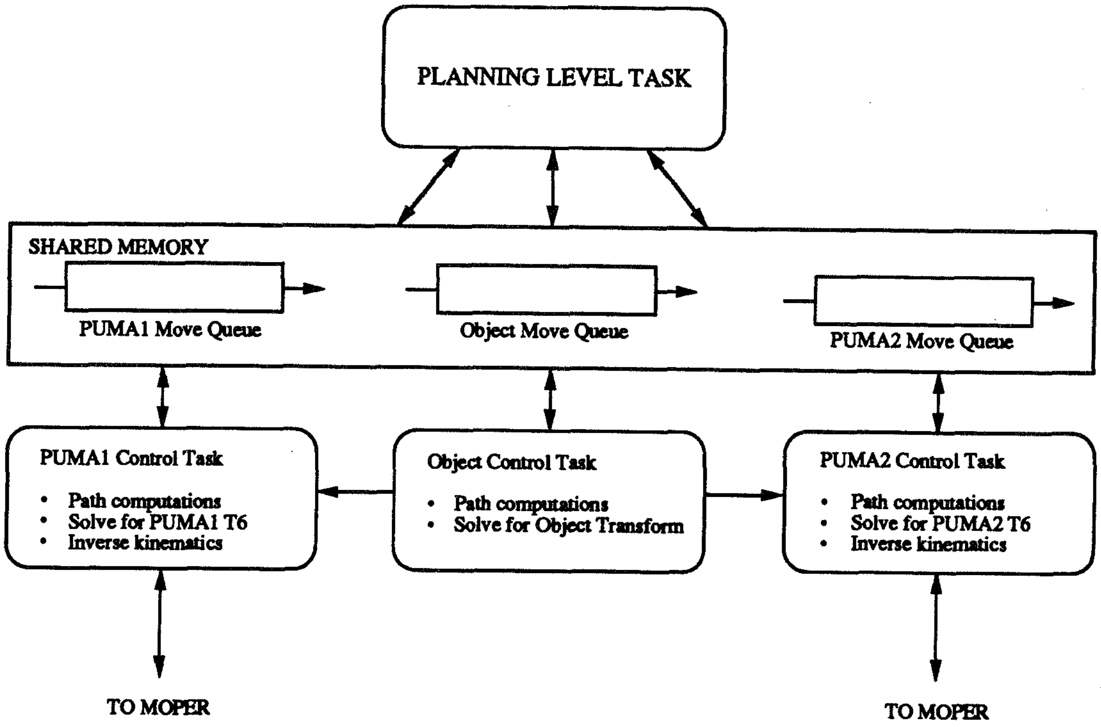


Figure 4: Task Diagram of a Multi-Robot RCCL System

code which determines the sampling rate for the set point generator. The simpler the control level code, the faster the sampling. For example, in the MicroVax II implementation, the sampling rate was 30 Hz. The CPU needed about 14 msec to perform the trajectory generation in the Cartesian mode.

In the new RCCL, the control level software can run either on one or more cpu's. For example, the RCA version uses one MicroVax CPU for the planning level for two robots and two MicroVax II slave cpu's for the two control levels. In the JPL implementation, since the code has been ported to a Sun 4/200 computer, everything runs on a single CPU (note that the Sun 4/200 is approximately 6 times faster that a MicroVax II). Presently, the sampling time for single and dual arm are 200 and 100 Hz, respectively. Note that this is the sampling time for trajectory generation and not for the joint servo control which runs at 1000 Hz. The multiprocessor RCCL makes it very easy to port the control portion of the code to a set of RISC-based boards which at the present time are approximately 2.5 times faster than the Sun 4 SPARC chip.

## 4 Teleoperation and Shared Control

Teleoperation remains one of the most reliable ways to manipulate objects in a non-assembly type environment. Space operation assembly and repair requires either sophisticated autonomous robotic capabilities or reliable teleoperation. The JPL telerobot approach is to develop both capabilities and advance from pure teleoperation to supervised autonomy and shared control. Effective teleoperation requires telepresence, meaning that the operator feels that he is at the remote site and is operating the robot rather than the hand controller. Kinesthetic force reflection is an important aid to the operator whenever the robot contacts an environment. Transmission time delay which is unavoidable in space environments decreases the effectiveness of force reflection. The JPL telerobot architecture is designed so that teleoperation experiments with or without time delay can be performed. The adverse effect of time delay can be eliminated by executing tasks in traded or shared control [6].

Teleoperation is implemented by using one UMC and one FRHC for each of the local teleoperation VME chassis. The joint positions are sensed from the UMC's. Cartesian space increments are computed using the hand controller Jacobian and sent to the remote site. This information is received by the MOPER and transmitted to the Sun 4 computer. An RCCL program is written to servo the arm to its present position indefinitely. This null trajectory is then *modified* according to the information received from the hand controller's Cartesian motion. This implementation has two advantages: 1- The trading of control between the autonomous and teleoperation is as simple as executing different MCM *macros*, and 2- shared control can readily be developed. Shared control is accomplished by issuing motions from the autonomous side and modifying them by the motion of the hand controllers. Force feedback to the operator is performed by reading the Lord force/torque sensor *raw stain gauges reading* once every 1.4 msec, transferring the data to the local teleoperation chassis, and applying joint torque computed based on the hand controller Jacobian.

Teleoperation software provides a user friendly menu implemented on the Sun View software. The operator can choose to operate in joint or Cartesian (world and tool) modes of operation and with or without force reflection in the Cartesian mode. For more details see [16]

## 5 Conclusions

This paper has described the MCM subsystem of the JPL/NASA Telerobot program. MCM is a complete autonomous and teleoperation environment which can be utilized both as a research tool and as a subsystem in the integrated environment of a telerobot. It offers single and dual arm autonomous, teleoperated, and shared control capabilities using state-of-the-art software and hardware for fast real-time computations.

### Acknowledgments

The research described in this document was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with National Aeronautics and Space Administration.

## References

[1] Hayati, S., Wilcox, B., 'Manipulator and Control Mechanization: A Telerobot Subsystem,' Proc. of Workshop on Space Telerobotics, Pasadena, Ca., January 20-22, 1987, pp. 219-227.

[2] Bejczy, A., and Szakaly, Z., 'Universal Computer Control System (UCCS) for Space Telerobots,' Proceedings of the 1987 IEEE International Conference on Robotics and Automation, Raleigh, NC, Mar. 30 - Apr. 3, 1987, pp. 318-324.

[3] Shanker, P., 'NASA Research and development for Space Telerobotics,' *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 24, No. 5, Sep. 1988, pp. 523-534.

[4] Matijevic, J., Zimmerman, W., Dolinsky, S., 'The NASA/OAST Telerobot Testbed Architecture,' Proceedings of NASA Conference on Space Telerobotics, Pasadena, California, Jan. 31 - Feb. 2, 1989.

[5] Bejczy A. K., Salisbury J. K., 'Controlling Remote Manipulators through Kinesthetic Coupling,' *Computers in Mechanical Engineering*, Vol. 2, No.1, July 1983, pages 48-60.

[6] Hayati, S. and Venkataraman, S., 'Design and Implementation of a Robot Control System with Traded and Shared Control Capability,' Proceedings of the 1989 IEEE Conference on Robotics and Automation, Scottsdale, Arizona, May 14-19, 1989.

[7] Kan, E., et. al., 'The JPL Telerobot Operator Control Station: Part I - Hardware,' Proceedings of NASA Conference on Space Telerobotics, Pasadena, California, Jan. 31 - Feb. 2, 1989.

[8] Kan, E., et. al., 'The JPL Telerobot Operator Control Station: Part II - Software,' Proceedings of NASA Conference on Space Telerobotics, Pasadena, California, Jan. 31 - Feb. 2, 1989.

[9] Hayward V., and Paul, R., 'Robot Manipulator Control Under Unix RCCL,' International Journal of Robotics Research, Vol. 5, No. 4, Winter 1987, pp. 94-111.

[10] Lee, J., Hayati, S., et. al., 'Implementation of RCCL, a Robot Control C Library, on a MicroVax II,' Proceedings of the SPIE conference on Advances in Intelligent Robotics Systems, Vol. 726, Cambridge, MA., Oct. 1986., pp. 26-31.

[11] Lloyd, J., Parker, M., McClain, R., 'Extending the RCCL programming environment to multiple robots and processors,' Proceedings of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, PA, April 1988, pp. 465-469.

[12] Lloyd, J., 'Implementation of a Robot Control Development Environment,' Master's Thesis, Department of Electrical Engineering, McGill University, Montreal, Canada, Dec., 1985.

[13] Wilcox, B., et. al., 'Autonomous Sensor-Based Dual-Arm Satellite Grappling,' Proceedings of NASA Conference on Space Telerobotics, Pasadena, California, Jan. 31 - Feb. 2, 1989.

[14] Hayati, S., Tso, K., and Lee, T., 'Generalized Master/Slave Coordination and Control for a Dual Arm Robotic System,' Proceedings of the 2nd International Symposium on Robotics and Manufacturing:Research, Education, and Applications, Albuquerque, New Mexico, Nov. 16-18, 1988, pp. 421-430.

[15] Paul, R., *Robot Manipulator: Mathematics, Programming, and Control*, MIT Press, Cambridge, Mass. 1981.

[16] Lee, T., 'Implementation and Design of a teleoperation System Based on a VME Bus/68020 Pipelined Architecture,' Proceedings of NASA Conference on Space Telerobotics, Pasadena, California, Jan. 31 - Feb. 2, 1989.

# On Discrete Control of Nonlinear Systems With Applications to Robotics

**Mansour Eslami**

Department of Electrical Engineering and Computer Science (M/C 154)
The University of Illinois at Chicago
Chicago, Illinois 60680-4348

## Abstract

Much progress has been reported in the areas of modeling and control of nonlinear dynamic systems in a continuous-time framework. From implementation point of view, however, it is essential to study these nonlinear systems directly in a discrete setting that is amenable for interfacing with digital computers. But to develop discrete models and discrete controllers for a nonlinear system such as robot is a nontrivial task. Robot is also inherently a variable-inertia dynamic system involving additional complications. Not only the computer-oriented models of these systems must satisfy the usual requirements for such models, but these must also be compatible with the inherent capabilities of computers and must preserve the fundamental physical characteristics of continuous-time systems such as the conservation of energy and/or momentum.

In this exploratory presentation we will review preliminary issues regarding discrete systems in general and discrete models of a typical industrial robot that is developed with full consideration of the principle of conservation of energy. Subsequently, we will review some research on the pertinent tactile information processing. Finally, we will review system control methods and how to integrate these issues in order to complete the task of discrete control of a robot manipulator.

## 1. Introduction

The real-time applications of robotics technology in space and/or in hostile environments require sophisticated use of teleoperational equipment that are capable of maneuvering difficult task in such situations. These nonlinear and inertia-variable systems exhibit a number of challenging problems for us that must be carefully analyzed before implementing a particular design. The overall nonlinear control problem can be divided into the following sub-problems:

Discrete Systems
Discrete Robot Models
Tactile Information
Control Methods
Integration and Future Research Directions.

In this exploratory paper we will describe some general results on these issues and we will present some preliminary thoughts on the future directions of research in this area.

## 2. Discrete Systems

The overall dynamic system is, of course, nonlinear and thus it must be looked at first in the most general sense. Certainly, and with the advent of computer-controlled methodologies, it is preferable to study the modeling of a general dynamical system directly in discrete forms, that are easily amenable for interfacing with digital computers. This approach is often superior to that of discretization of a continuous-time process (model) which inevitably yields errors of significant magnitudes specially for a nonlinear system. Although the robot manipulator is a special case of a nonlinear dynamic system, we can not escape some commonalities that must be reviewed in the context of a general nonlinear discrete dynamic system.

According to the authoritative survey paper of Jury and Tsypkin [24] a discrete automatic system can be subdivided into the following types.

(a) Relay automatic systems wherein the quantization takes place in jumps in both system parameters and maybe system structure resulting in a nonlinear problem.

(b) Pulse (or impulse) automatic systems also known as sampled-data systems. In this case the system is described by a set of finite-difference equations in time. Here we treat the parameters as constants, or in a realistic but much complicated way as time-varying variables. The theory for time-varying situation is not yet fully developed and this is an exciting and promising research area.

(c) Digital automatic system that combines both cases (a) and (b). Here unavoidably digital computers must be used as part of the overall systems.

Analysis of linear pulsed system entails certain classical methods that are extensively developed and documented in the literature. A special characteristic of linear pulsed systems is that they have a finite critical amplification which is independent of the order of the system [24]. Only recently stability robustness under large parameter variations of this class of systems has been established in [12]. The main analytical *tool* to analyze these dynamic systems is theory of difference equations that is well developed in the early textbook of Levy and Lessman [33].

The synthesis of pulsed system also entails properties that are now considered classical, but nevertheless very important. One issue is concerned with pulse correction or compensation that corresponds to introducing a discrete filter that changes (or corrects) the sequence of pulses [24]. An interesting observation that is reported in [24] and is credited to a 1964 Rutman's paper (in Russian) is as follows. When synthesizing an optimal discrete control system, if we combine discrete correction policy that changes the shape of control pulses with some external control action (which is resulted from the sensory information *per se*), then we get the best result. This simple concept has been extensively studied by a number of researchers in a number of different disciplines. In a very interesting paper on nonlinear discrete-time systems Geromel and Cruz [14] show that the additive perturbation (the same as the external input) is much richer than the gain perturbation (that is designed using feedback automatic policies). They also show that a system with high tolerance to gain perturbation may present a low tolerance to additive perturbation and *vice versa*. Therefore a compromise must be made between weighting these two inputs in a given application and this is, in our opinion, a very interesting result. In an earlier work we used the term auxiliary input to emphasize similar situations, for example [36].

Generally speaking, the synthesis of discrete linear system is well developed for constant parameter cases. It is extremely complicated and to the large extent undeveloped for the case in which the parameters are changing. This is a very exciting problem to investigate and no bonafide procedure is documented for its treatment. It seems that the general theory of averaging, which was introduced for continuous-time differential equations, for example in [19], may be the starting point to deal with these types of problems. Recently, Astrom also suggests that this approach is very promising to understand parameter variation behavior of a dynamic system [4]. The extension of this concept to discrete system is not fully developed, although some results are reported in [2].

## 3. Discrete Robot Models

The discretization of robot dynamics, which is inherently a variable-inertia dynamic system, involves additional complications. Not only the computer-oriented models of these systems must satisfy the usual requirements for such models, but these must also be compatible with the inherent capabilities of computers and must preserve the fundamental physical characteristics of continuous-time systems. Neuman and Tourassis [42] have developed a discrete-time dynamic model for robot that guarantees conservation of energy and momentum at each sampling time. The principle of conservation of energy plays the central role in discretization of a nonlinear mechanical system. Here we will briefly describe the discrete model which is reported in [42] for illustration.

Consider the generalized state of an n-degree-of-freedom robot as follows.

$$D(x^p)x^a + C(x^p, x^v)x^v + g(x^p) = u(t). \tag{1}$$

Here $x^p \in R^n$ is the generalized position vector, $x^v = \dot{x}^p \triangleq v$ is the velocity vector and $x^a = \dot{x}^v \triangleq a(x^p, x^v, u)$ is the acceleration vector. The symmetric positive-definite matrix $D(x^p)$ represents an inherently variable-inertia matrix of the robot. The Coriolis and centrifugal forces are shown by $C(x^p, x^v)x^v$; and $g(x^p)$ is the vector of gravity. The external or applied input torque (or force) is shown by u(t). Equation (1) can also be written in a general form as follows.

$$D(x^p)\, a(x^p, x^v, u) = f(x^p, x^v, u), \tag{2}$$

where $f(x^p, x^v, u)$ represents the remaining nonlinear terms and is the *net* generalized force which is acting on the system. Needless to say that a closed-form solution of (2) is very difficult to attain and the final answer is very complicated to compute. There are, however, a number of functions of position and velocity vectors whose values remain constant throughout the motion and depend only on the initial conditions [42]. These functions are called the *integrals-of-the-motion* [29]. For an n-degree-of-freedom (n-DOF) robot there are (2n − 1) independent such functions. To find all of these functions is equivalent to solving for the entire system of differential equations of the dynamic system.

To discretize a nonlinear mechanical system, Greenspan [17] and [18] has defined a new concept for the work which guarantees the conservation of energy. In this analysis Greenspan has assumed that the mechanical system has constant mass/inertia and the acceleration remains constant in each sampling interval. Neither of these assumptions are true in robotics. These constraints have led Neuman and Tourassis [42] to develop a new discrete model for a 3-DOF cylindrical robot which is based on generating the *integrals-of-the-motion* of this system while preserving the principle of the conservation of energy.

To develop the Neuman and Tourassis discrete model we present the following set of notation first. The *work* that is done by a force f (according to the classical mechanics) is equal to f·dx, where dx is displacement along the motion direction and *dot* represents the inner product. Also corresponding to (2) at instant k we have Da(k) = f(k), where D is a constant inertia matrix. The principle of the conservation of energy with constant $D(x^p) \equiv D$ states that the work at instant k is

$$E(k+1) - E(k) \triangleq w(k) = \tfrac{1}{2}v^T(k+1)Dv(k+1) - \tfrac{1}{2}v^T(k)Dv(k). \tag{3}$$

Here E(k) is the total energy at the kth-sampling time. The Greenspan discrete mechanics uses a trapezoidal (smoothing) formula for the position x(k) and a forward Euler formula for the velocity v(k). These equations and an expression for the work can be stated as follows.

$$x(k+1) = x(k) + \frac{T}{2}[v(k+1) + v(k)], \tag{4a}$$

$$v(k+1) = v(k) + Ta(k), \tag{4b}$$

$$w(k) = f(k)\cdot[x(k+1) - x(k)]. \tag{4c}$$

In this study the input is updated at each sampling time and is maintained constant in that period. Our task is to use the above schemes which may not satisfy the principle of the conservation of energy. Or to devise a new scheme to discretize the robot nonlinear dynamics.

To illustrate the Neuman and Tourassis procedure a cylindrical robot dynamics is chosen [9]. The coordinate vector of this robot is $q = [\theta, z, r]$, where $\theta$ is the rotation angle, z is the vertical translation and r is a radial translation. For this arrangement the kinematic equations and the equation-of-the-motion are, respectively, as follows.

$$\dot{\theta} = \omega, \qquad \dot{z} = \nu, \quad \text{and} \qquad \dot{r} = v, \tag{5}$$

$$[J + j(r)]\dot{\omega} + \frac{\partial j}{\partial r}\omega v = F_\theta(t), \tag{6}$$

$$M\dot{\nu} + Mg = F_z(t), \tag{7}$$

$$m\dot{v} - ½\frac{\partial j}{\partial r}\omega^2 = F_r(t). \tag{8}$$

Here J is the constant inertia for vertical column; j(r) is the variable inertia of the radial link (note that: $j(r) = mr^2 - m_pRr$); m is the mass of the radial link with the mass of payload $m_p$, that is concentrated at the tip of radial link; M is the vertically translated mass of column and radial link; and $F_\theta$, $F_z$, $F_r$ are the external forces/torques that drive the $\theta$, z, and r DOF, respectively [42]. The inertial matrix $D(r) = \text{diag}[J + j(r)Mm]$ of the cylindrical robot is diagonal and depends explicitly upon the radial displacement. In (6) the term $(\partial j/\partial r)\omega v$ corresponds to the Coriolis torque and in (8) the term $(\partial j/\partial r)\omega^2$ corresponds to the centrifugal force. We note that in this case the vertical motion (7) is decoupled but (6) and (8) are two coupled-nonlinear differential equations.

If we assume that in each sampling period $F_\theta(k)$ remains constant, then a direct differentiation of (9) below shows that (6) can be expressed as follows.

$$\frac{d}{dt}\{[J + j(r)]\omega\} = F_\theta(k). \tag{9}$$

The complete integrability of this equation is a consequence of the fact that the total energy of the cylindrical robot is independent of the rotation $\theta$. Here the *integral-of-the-motion* reinforces the conservation of the *angular* momentum $p_\theta = [J + j(r)]\omega$.

Similarly, the vertical z-coordinate can be integrated to yield

$$\frac{d}{dt}(M\nu) = F_z(k) - Mg, \tag{10}$$

that shows the conservation of the *linear* momentum $p_z = M\nu$. Finally, with some algebra we can show that from (8) and (6) we get

$$½\frac{d}{dt}\{[J + j(r)]\omega^2 + mv^2\} = \frac{d}{dt}[F_\theta(k)\theta + F_r(k)r]. \tag{11}$$

This equation guarantees the conservation of energy for the coupled rotational and radial degree of freedom.

In summary, we have three *integrals-of-the-motion* for a 3-DOF cylindrical robot. Integration of these equations will lead directly to the discrete-time robot model which are exact and as follows.

$$\{J + j[r(k+1)]\}\omega(k+1) - \{J + j[r(k)]\}\omega(k) = TF_\theta(k), \tag{12}$$

$$M[\nu(k+1) - \nu(k)] = T[F_z(k) - Mg], \tag{13}$$

$$½\{J + j[r(k+1)]\}\omega^2(k+1) - ½\{J + j[r(k)]\}\omega^2(k) + ½m[v^2(k+1) - v^2(k)] =$$

$$[\theta(k+1) - \theta(k)]F_\theta(k) + [r(k+1) - r(k)]F_r(k). \tag{14}$$

These equations are developed with no constraints upon T.

Finally, the following smoothing formulae are used to simplify (12) to (14) to yield (18) to (20).

$$\theta(k+1) - \theta(k) = \frac{T}{2}[\omega(k+1) + \omega(k)], \tag{15}$$

$$z(k+1) - z(k) = \frac{T}{2}[v(k+1) + v(k)], \tag{16}$$

$$r(k+1) - r(k) = \frac{T}{2}[v(k+1) + v(k)], \tag{17}$$

$$TF_\theta(k) = \{J + j[r(k+1)]\}\omega(k+1) - \{J + j[r(k)]\}\omega(k), \tag{18}$$

$$TF_z(k) = M[v(k+1) - v(k)] + TMg, \tag{19}$$

$$TF_r(k) = m[v(k+1) - v(k)] - \tfrac{1}{2}T\{\frac{j[r(k+1)] - j[r(k)]}{r(k+1) - r(k)}\}\omega(k)\omega(k+1). \tag{20}$$

These equations are the discrete state-space model of a cylindrical robot which guarantee the conservation of energy and momentum in each sampling period. The main approximation comes from the first-order smoothing equations. From algorithmic point of view the Neuman and Tourassis discrete model at each sampling time requires that a system of 2n linear and nonlinear coupled equations be computed and/or analyzed. At each sampling instant the above formulation lends itself to a nested two-loop iterative algorithm consisting of an outer loop (which sets the system of 2n equations) and an inner loop (which solves the system of 2n equations). This approach eliminates the indeterminacies that may occur in (20) when r(k+1) becomes r(k). The inner loop corresponding to the kth iteration of the outer loop starts with the current values of all variables. Then from (15) to (20) the next set of values of variables at the (k+1)th-sampling time is generated. This procedure is followed accordingly and when this algorithm identifies that r(k+1) = r(k), it automatically replaces c(k+1) by c(k), where $c(k+1) = \tfrac{1}{2}\{j[r(k+1)] - j[r(k)]\} / [r(k+1) - r(k)]$ is the centrifugal coefficient in (20). This procedure thus eliminates the indeterminacies by using the last computed value of centrifugal coefficient. The algorithm then repeats. Recently this result of Neuman and Tourassis [42] has been revisited by Lee and Tsay [32], who have introduced a shift transformation matrix based on general discrete orthogonal polynomials. Lee and Tsay have shown that using these polynomials we can transform the difference equations (15) to (20) into a set of algebraic equations resulting in computational simplicity. In other words, instead of using a nested two-loop iterative algorithm we can solve these equations simultaneously and we can reduce the computational burden significantly.

The above model can also be used in a number of inverse problems in order to compute, for example, the desired forces/torques for a desired trajectory. This approach can be also used to compute a sampling time if a desired trajectory is specified. Although in general there are other issues regarding the choice of sampling strategy which must be considered as we will briefly mention them.

Finally, in order to assess the fidelity of a given discrete model we need to compare its response to a known input with the response which will be resulted from other discretization methods. To this effect Neuman and Tourassis [42] have suggested to look at the step response of (18) to (20) with that resulted from discretizing the robot model based on a number of other discretization methods. The initial condition is chosen zero and T = 10 ms with a 3 seconds total simulation time. In particular, Neuman and Tourassis have suggested to select $|\Delta E(k)|/|E(k)|$ and/or $|\Delta p_\theta(k)|/|p_\theta(k)|$ as an appropriate criterion for evaluating the performance of each algorithm. The rationale behind the residual terms is that, in general, we can not compute the exact value of responses analytically. In this study, it has been shown that the discrete dynamic robot model outperforms constant fixed-sampling-period algorithms including the forth-order-Runge-Kutta algorithm. The Neuman and Tourassis result compares well with the fifth and sixth-order Runge-Kutta-Verner algorithm [42].

Recently, Tumeh has also looked at robot dynamic discretization by using discentralized equivalent joint model that each is developed in a discrete-time form [53]. This work has some potential use but it is based on the assumption of constant inertia in each sampling time which is not desirable.

The research in this area must be continued in order to generate a more versatile discrete model which are applicable to a number of industrial robot manipulators.

## 4. Tactile Information

Recent experimental studies by Sharpe [52] show that the most exacting tasks are those which can be performed through the feel and not visually. Sharpe has concluded from his experimental studies that for any dexterous skilled operation there must be a normal positional forward bandwidth (20 to 30 Hz) and a high frequency bandwidth force reflection (5 to 10 KHz). To achieve this task a new class of bilateral control system that uses force feedback manipulator called covariant bilateral control (CBC) is then introduced. This method enables us to design various sub-systems independently, and to control a nonlinear and resonant system more effectively than before. The use of *feel* enables us to adapt the level of damping of the overall system by altering the response to the sensing information and it also enables us to sense gravity and inertia effects and to respond to these effectively.

The inherent time delays that are introduced in a bilateral servomechanism causes many concerns regarding stability, fidelity and usefulness of the force reflection. Sharpe's experimental work has shown that the application of CBC in position controlling a resonant load with adjustable delays in the forward and return paths, enables a manipulator operating in earth orbit be conceivably controlled with force reflection from an earth station. This task can be done using a high frequency force feedback information in an adaptive manner. The sub-systems rely on the local control loops and therefore their stability are not disturbed by the transmission delays. The stability of these sub-systems must be studied separately. The experimental work has shown that up to a delay of 6.0 ms the full force feedback loop-gain could be used. For higher delays the acceptable gain setting reduced as delay was increased. With a delay of 120 ms the forces tend to fall below an acceptable threshold [52]. In general, the conclusion is that the CBC manipulator has many interesting properties and is capable of providing a good *feel* even with sub-systems that are having nonlinearity, backlash and delay. Clearly the force fidelity is a function of the quality of the proper sub-system. With force feedback the operator can control the behavior of a resonant system while this control is not possible even when the operator is capable of viewing what is happening to the system in a conventional sense [52]. Also we note that the force reflected onto the operator is measured in terms of various electrical quantities associated with each subsystems. The effects of delay is very clear to the operator, but the amount of the delay is very difficult to quantify. This fact implies that the operator's response is instinctive. The very fast force feedback response (500Hz) allows the human operator to adapt to needs in controlling an otherwise uncontrollable higher-order resonant system.

## 5. Control Methods

In an ideal situation, in which the manipulator model and the workspace are known precisely and we have sufficient sensory information, simple controllers will enable us to perform most of our difficult tasks. In a real-world environment, however, the manipulator model and/or parameters are not accurately known and we do not have precise sensory information. Thus we need to devise sophisticated feedback controllers and we need to put together coherently a number of issues which are complementing each other in order to perform a nontrivial telerobotic task.

To study the discretization of a general nonlinear discrete control problem we must choose: first, a proper (or *optimal*) sampling strategy; and secondly a method to deal with the parameter variations. We believe to develop an optimal sampling strategy we should set a constrained optimization scheme in which we define a performance index as a function of sampling time and subsequently that function, subject to the physical (or hardware) realization constraints, will be minimized. A simple version of this problem is reported in [39]. We define a sampling time *optimal* if is physically (or hardware) realizable and it also gives enough times for various data acquisition or computational tasks to be completed. This sampling strategy must also give enough times for controllers to execute their functions.

To account for parameter variations we should use the discrete model reference adaptive control to lay out a foundation for various controller designs. Although this choice may conflict with our sampling method. Subsequently, we must look at parameter variations using the averaging concepts to study its impact on the overall system stability. There are a number of unanswered questions in discrete adaptive control which must be looked at simultaneously and in that sense this is an interesting research area.

Therefore in short the control method must entail certain constrained optimization methods to develop the *optimal* sampling strategy and discrete model reference adaptive control with averaging concepts to design controllers. These general methodologies are well established in their own rights. But a number of important questions regarding discrete nonlinear control problems must now be answered before we can successfully complete the task of real-time implementing these control algorithms and before we can successfully assemble a number of perhaps conflicting issues together.

## 6. Integration and Future Research Directions

Using the methodologies that we have cited before, analytical methods to yield the *optimal* sampling strategies for real-time implementation, specially in conjunction with the force control; and the robustness of system performance regarding the large parameter changes must all be developed. Subsequently, synthesis methods that incorporate this information to correct or to compensate for discretization errors as well as all the other usual disturbances that exit in a complicated dynamic system such as a space teleoperator must be established.

The utilization of discrete averaging theory to deal with parameter variations will advance the state of knowledge in design of discrete controllers for a general nonlinear system. This new result will enable us to address many uncertainties that we usually left out due to the lack of information regarding: friction; unprecise inertia; and unmodeled dynamics or other external uncertainties in our system.

## 7. References

[1] G. Ambrosino, G. Celentano, and F. Garofalo, "Robust model tracking control for a class of nonlinear plants," *IEEE Trans. Auto. Contr.*, vol. AC-30, no. 3, pp. 275-279, March 1985.

[2] B.D.O. Anderson, *et al.*, *Stability of Adaptive Systems*. Cambridge, MA: MIT Press, 1986.

[3] R.P. Anex, Jr. and M. Hubbard, "Modelling and adaptive control of a mechanical manipulator," *ASME Trans. J. Dynam. Syst., Meas. Contr.*, vol. 106, no. 3, pp. 211-217, Sept. 1984.

[4] K.J. Astrom, "Adaptive feedback control," *Proc. IEEE*, vol. 75, no. 2, pp. 185-217, Feb. 1987.

[5] A. Balestrino, G. De Maria, and L. Sciavicco, "An adaptive model following control for robot manipulators," *ASME Trans. J. Dynam. Syst., Meas. Contr.*, vol. 105, no. 3, pp. 143-151, Sept. 1983.

[6] A. Balestrino, G. De Maria, and A.S.I. Zinober, "Nonlinear adaptive model-following control," *Automatica*, vol. 20, pp. 559-568, 1984.

[7] A.K. Bejczy, "Robot arm dynamics and control," Jet Propulsion Lab., Pasadena, CA, Tech. Memo. 33-669, Feb. 1974.

[8] A.K. Bejczy and R.P. Paul, "Simplified robot arm dynamics for control," in *Proc. 20th IEEE Conf. on Decision and Contr.*, pp. 261-262, Dec. 1981.

[9] M. Brady *et al.*, Eds., *Robot Motion: Planning and Control.* Cambridge, MA: MIT Press, 1982.

[10] L.S. Cooper and D. Steinberg, *Introduction to methods of optimization.* Philadelphia: W.B. Saunders Co., 1970.

[11] J.E. Dennis, "A user's guide to nonlinear optimization algorithms," *Proc. IEEE,* vol. 72, no. 12, pp. 1765-1776, Dec. 1984.

[12] M. Eslami, "On sensitivity of nonlinear digital systems," in *Proc. 18th Allerton Conf.*, pp. 424-432, Oct. 1980.

[13] E. Freud, "Fast nonlinear control with arbitrary pole-placement for industrial robots and manipulators," *Int. J. Robotics Research,* vol. 1, pp. 65-78, 1982.

[14] J.C. Geromel and J.J. da Cruz, "On the robustness of optimal regulators for nonlinear discrete-time systems," *IEEE Trans. Auto. Contr.*, vol. AC-32, no. 8, pp. 703-712, Aug. 1987.

[15] E.G. Gilbert and I.J. Ha, "An approach to nonlinear feedback control with applications to robotics," *IEEE Trans. Syst., Man, Cybern.* vol. SMC-14, no. 6, pp. 879-884, Nov./Dec. 1984.

[16] B.S. Gottfried and J. Weisman, *Introduction to Optimization Theory.* NJ: Prentice-Hall, 1973.

[17] D. Greenspan, *Introduction to Numerical Analysis and Applications.* Chicago: Markham, 1971.

[18] D. Greenspan, "A new explicit discrete mechanics with applications," *J. Franklin Inst.*, vol, 294, no. 4, pp. 231-240, Oct. 1972.

[19] J.K. Hale, *Ordinary Differential Equations.* New York: Wiley, 1969. Reprint, R.E. Krieger, 1980.

[20] E.J. Haug, *Computer-Aided Analysis and Optimization of Mechanical Systems Dynamics.* New York: Springer-Verlag, 1984.

[21] P. Henrici, *Error Propagation for Difference Methods.* New York: Wiley, 1963.

[22] F. Hildebrand, *Introduction to Numerical Analysis.* New York: McGraw-Hill, 1956.

[23] D.M. Himmelblau, *Applied Nonlinear Programming.* New York: McGraw-Hill, 1972.

[24] E.I. Jury and Ya. Z. Tsypkin, "On the theory of discrete systems," *Automatica,* vol. 7, pp. 89-107, 1971.

[25] R.E. Kalman and J.E. Bertram, "Control system analysis and design *via* the 'second method' of Lyapunov: Part I, Continuous-time systems," *ASME Trans. J. Basic Eng.*, pp. 371-393, June 1960.

[26] R.E. Kalman and J.E. Bertram, "Control system analysis and design *via* the 'second method' of Lyapunov: Part II, Discrete-time systems," *ASME Trans. J. Basic Eng.*, pp. 394-400, June 1960.

[27] A.J. Koivo and T.H. Guo, "Adaptive linear controller for robotic manipulators," *IEEE Trans. Auto. Contr.*, vol. AC-28, no. 2, pp. 162-171, Feb. 1983.

[28] I.D. Landau and L.S. Lozano, "Unification of discrete time explicit model reference adaptive control designs," *Automatica,* vol. 17, no. 4, pp. 593-611, 1981.

[29] L.D. Landau and E.M. Lifshitz, *Mechanics.* New York: Pergamon Press, 1976.

[30] Y.D. Landau,[1] *Adaptive Control - The Model Reference Approach.* New York: Marcel Dekker, Inc., 1979.

[31] J.P. LaSalle, "Stability theory for difference equations," in *Studies in Ordinary Diff. Equ.*, MAA, vol. 14, J.K. Hale, Ed., pp. 1-31, 1977.

[32] T.-T. Lee and Y.-F. Tsay, "Analysis of discrete dynamic robot models," *IEEE J. Robotics Automation,* vol. RA-3, no. 6, pp. 583-590, Dec. 1987.

[33] H. Levy and F. Lessman, *Finite Difference Equations.* New York: MacMillan, 1961.

[1] The same author as I.D. Landau.

[34] K.Y. Lim and M. Eslami, "Adaptive controller designs for robot manipulator systems using Lyapunov direct method," *IEEE Trans. Auto. Contr.*, vol. AC-30, pp. 1229-1233, Dec. 1985.

[35] K.Y. Lim and M. Eslami, "Adaptive controller design for robot manipulator systems yielding reduced Cartesian error," *IEEE Trans. Auto. Contr.*, vol. AC-32, no. 2, pp. 184-187, Feb. 1987.

[36] K.Y. Lim and M. Eslami, "Robust adaptive controller designs for robot manipulator systems," *IEEE JRA*, vol. RA-3, no. 1, pp. 54-66, 1987.

[37] R. Lozano and I.D. Landau, "Redesign of explicit and implicit discrete time model reference adaptive control schemes," *Int J. Contr.*, vol. 33, no. 2, pp. 247-268, 1981.

[38] J.M. Maciejowski, "Asymptotic recovery for discrete-time systems," *IEEE Trans. Auto. Contr.*, vol. AC-30, no. 6, pp. 602-605, June 1985.

[39] S.M. Melzer and B.C. Kuo, "Sampling period sensitivity of the optimal sampled data linear regulators," *Automatica*, vol. 7, pp. 367-370, 1971.

[40] T. Mori, N. Fukuma, and M. Kuwahara, "On the discrete Lyapunov matrix equation," *IEEE Trans. Auto. Contr.*, vol. AC-27, no. 2, pp. 463-464, April 1982.

[41] R.L. Morris and C.P. Neuman, "Model reference adaptive control with multiple samples between parameter adjustments," *IEEE Trans. Auto. Contr.*, vol. AC-26, no. 2, pp. 534-537, April 1981.

[42] C.P. Neuman and V.D. Tourassis, "Discrete dynamic robot models," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, no. 2, pp. 193-204, March/April 1985.

[43] J.T. Oden, Ed., *Computational Methods in Nonlinear Mechanics*, vol. 1. New York: Springer-Verlag, 1975.

[44] J.T. Oden, Ed., *Computational Methods in Nonlinear Mechanics*, vol. 2. New York: Elsevier, 1980.

[45] K. Ogata, *Discrete-Time Control Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[46] K. Ohno, "A new approach to differential dynamic programming for discrete time systems," *IEEE Trans. Auto. Contr.*, vol. AC-23, no. 1, pp. 37-47, Feb. 1978.

[47] J.M. Ortega and W.C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. New York: Academic Press, 1970.

[48] D. Paviani and D.M. Himmelblau, "Constrained nonlinear optimization by heuristic programming," *Operations Research*, vol. 17, pp. 872-882, 1969.

[49] E. Polak, D.E. Mayne, and D.M. Stimler, "Control system design *via* semi-infinite optimization," *Proc. IEEE*, vol. 72, no. 12, pp. 1777-1794, Dec. 1984.

[50] J.R. Rice, *Numerical Methods, Software and Analysis*. New York: McGraw-Hill, 1983.

[51] C. Samson, "Robust nonlinear control of robotic manipulators," in *Proc. 22nd IEEE Conf. on Decision and Contr.*, pp. 1211-1216, Dec. 1983.

[52] J.E.E. Sharpe, "Technical and human operational requirements for skill transfer in teleoperations," in *Proc. Int. Symp. Teleoperation and Contr.*, pp. 175-187, July 1988.

[53] Z.S. Tumeh, "A gain scheduling approach in decentralized manipulator control using discentralized equivalent joint models," in *Proc. IEEE Conf. on Decision and Control*, pp. 616-621, Dec. 1988.

[54] J.S. Vandergraft, *Introduction to Numerical Computations*. New York: Academic Press, 1978.

[55] M.W. Walker and D.E. Orin, "Efficient dynamic computer simulation of robotic mechanisms," *ASME Trans. J. Dynam. Syst., Meas. Contr.*, vol. 104, pp. 205-211, Sept. 1982.

[56] X. Xianya and R.J. Evans, "Discrete-time adaptive control for deterministic time-varying systems," *Automatica*, vol. 20, pp. 309-319, 1984.

[57] N.N. Yaneko, *The Methods of Fractional Steps*. New York: Springer-Verlag, 1971.

[58] T. Yoshimura and M. Tomizuka, "Application of model reference adaptive techniques to a class of nonlinear systems," *ASME J. Dynam. Syst., Meas. Contr.*, vol. 102, no. 2, pp. 158-163, June 1981.

[59] L. Zheng, "Discrete-time adaptive control of deterministic fast time-varying systems ," *IEEE Trans. Auto. Contr.*, vol. AC-32, no. 5, pp. 444-447, May 1987.

# A SPATIAL OPERATOR ALGEBRA FOR MANIPULATOR MODELING AND CONTROL

G. Rodriguez, K. Kreutz, and A. Jain
Jet Propulsion Laboratory/California Institute of Technology
4800 Oak Grove Drive, MS 198-330, Pasadena, CA 91109

## Abstract

A recently developed spatial operator algebra, useful for modeling, control and trajectory design of manipulators is discussed. The elements of this algebra are linear operators whose domain and range spaces consist of forces, moments, velocities, and accelerations. The effect of these operators is equivalent to a spatial recursion along the span of a manipulator. Inversion of operators can be efficiently obtained via techniques of recursive filtering and smoothing. The operator algebra provides a high-level framework for describing the dynamic and kinematic behavior of a manipulator and control and trajectory design algorithms. The interpretation of expressions within the algebraic framework leads to enhanced conceptual and physical understanding of manipulator dynamics and kinematics. Furthermore, implementable recursive algorithms can be immediately derived from the abstract operator expressions by inspection. Thus, the transition from an abstract problem formulation and solution to the detailed mechanization of specific algorithms is greatly simplified. This paper discusses the analytical formulation of the operator algebra, as well as its implementation in the Ada programming language.

## 1. Introduction: A Spatial Operator Algebra

A new approach to the modeling and analysis of systems of rigid bodies interacting among themselves and their environment has recently been developed in [1-10]. This work develops a framework for clearly understanding issues relating to the kinematics, dynamics and control of manipulators in dynamic interaction with each other, while keeping the complexity involved in analyzing such systems to manageable proportions.

The analysis of robot arms given in [1-10] has shown that certain linear operators are always present in the equations describing the dynamical and kinematical behavior of such arms. As a class, these operators are called "spatial operators" since they show how forces, velocities, and accelerations propagate through space from one rigid body to the next. Not only do such operators have obvious physical interpretations, but they are implicitly equivalent to tip-to-base or base-to-tip recursions which, if needed, can be immediately turned into implementable algorithms by projecting them onto appropriate coordinate frames.

Compositions of spatial operators, when allowed to operate on functions of the joint velocities and accelerations, result in exactly the dynamical equations of motion which arise from a Lagrangian analysis. The fact that the operators have equivalent recursive algorithms is a generalization of the well-known equivalence between the Lagrangian and recursive Newton-Euler approaches to manipulator dynamics [17]. The operator-based formulation of robot dynamics leads to a complete integration of these two approaches, so that analytical expressions can be shown to almost always have implicit, and obvious, recursive equivalents which are straightforward to mechanize.

The essential ingredients of the operator algebra are the operations of addition and multiplication [11, 12]. There is also an "adjoint," or "*", operator which can operate on elements of the spatial algebra. If a spatial operator $A$ is "causal," in a sense to be described below, then its adjoint $A^*$ is "anticausal." Operator inversion is also defined in the spatial operator algebra. For an arbitrary finite dimensional linear operator, inversion is achieved by the traditional techniques of linear algebra. However, many important spatial operators encountered in multibody dynamics belong to a class that can be factored as the product of a causal operator, a diagonal operator, and an anticausal operator. For these operators, inversion can often be achieved using the inward/outward sweep solutions of spatially recursive Kalman filtering and smoothing described in [1,3,6,8].

That the equations of multibody dynamics can be completely described by an algebra of spatial operators is certainly of mathematical interest. However, the significance of this result goes beyond the mathematics and is useful in a very practical sense. The spatial algebra provides a convenient means to manipulate the equations describing multibody behavior at a very high level of abstraction. This liberates the user from the excruciating detail involved in more traditional approaches to multibody dynamics where often one "can't see the forest for the trees." Furthermore, at any stage of an abstract manipulation of equations, spatially recursive algorithms to implement the operator expressions can be readily obtained by inspection. Therefore the transition from abstract operator mathematics to practical implementation is trivial to perform and requires only a simple mental exercise. Although, when applied to the dynamical analysis of an $N$ link manipulator the algebra typically leads to $O(N)$ recursive algorithms, numerical efficiency is not the main motivation for its development. What the algebra primarily offers is a powerful mathematical framework that because of its simplicity is believed to be superior for addressing advanced control issues.

Another characteristic of the spatial algebra is that it is closed, in the sense that it completely describes the dynamics of multiple rigid bodies. There are no situations (at least none have been found to date) in which it is necessary to go outside the algebra in order to solve rigid multibody dynamics problems. This implies that the user can reliably use the algebra knowing that the language of discourse can cover all problems that can be modeled as a "world" of multiple rigid bodies dynamically interacting. The algebra therefore puts a mathematical wrapping around this world. It provides a self-contained framework to formulate, analyze, and understand higher-level modeling and control issues.

To show the power and use of the spatial operator algebra, several applications of the algebra to robotics will be presented: 1) An operator representation of the manipulator Jacobian matrix; 2) The robot dynamical equations formulated in terms of the spatial algebra, showing the equivalence between the recursive Newton-Euler formulations to robot dynamics in a far more transparent way than [17]; 3) The operator factorization and inversion of the manipulator mass matrix which immediately results in $O(N)$ recursive forward dynamics algorithms; 4) The joint accelerations of a manipulator due to a tip contact force; 5) The recursive computation of the equivalent mass matrix as seen at the tip of a manipulator (the operational space mass matrix of Khatib [13]); 6) Recursive forward dynamics of a closed chain system. Finally, we will discuss in general terms additional applications and current research involving the application of the spatial operator algebra.

## 2. The Jacobian Operator

After defining a link spatial velocity to be $V(k) = col[\omega(k), v(k)]$, the recursion which describes the relationship between joint angle rates, $\dot{\theta} = col[\dot{\theta}(1), \cdots, \dot{\theta}(N)]$, and link velocities, $V = col[V(1), \cdots, V(N)]$, is [8]

```
$V(N+1) = 0$
LOOP $k = N, \cdots, 1;$
    $V(k) = \phi^T(k+1,k)V(k+1) + H^T(k)\dot{\theta}(k)$
END LOOP;
$V(0) = \phi^T(1,0)V(1)$
```

$H^T(k) = h(k)$ where $h(k) \in R^6$ is the unit vector in the direction of the $k^{th}$ joint axis, and $\phi^T(k+1,k)$ is the Jacobian which transforms velocities across a rigid link. This recursion represents a base-to-tip recursion which shows how link velocities propagate outward to the tip, point "0" on link 1, from the base "link $N+1$," assuming that the base has zero velocity. Note that the link numbering convention used here, and in [1-10], increases from the tip to the base unlike the numbering convention described in most robotics textbooks [14].

Summation of the above recursion leads to

$$V(k) = \sum_{i=k}^{N} \phi^T(i,k)H^T(i)\dot{\theta}(i)$$

where the facts that $\phi(i,i) = \mathcal{I}$ and $\phi(i,j)\cdot\phi(j,k) = \phi(i,k)$ have been used. Also note that $\phi^{-1}(i,j) = \phi(j,i)$. This naturally suggests that we define the "operators" $H^* = diag[H^T(1), \cdots, H^T(N)]$, $B^* = [\phi^T(1,0), 0, \cdots, 0]$ and

$$\phi \triangleq \begin{pmatrix} \mathcal{I} & 0 & 0 & \cdots & 0 \\ \phi(2,1) & \mathcal{I} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ \phi(N,1) & \phi(N,2) & \cdots & \cdots & \mathcal{I} \end{pmatrix}$$

This results in $V(0) = B^*\phi^*H^*\dot{\theta}$ or

$$J = B^*\phi^*H^* \tag{2.1}$$

The Jacobian operator in (1.1) is seen to be the product of three operators $B^*$, $\phi^*$ and $H^*$. The operator $H^*$, being block diagonal, is called "memoryless" while the operator $B^*$ projects out the link 1 velocity $V(1)$ of the composite velocity $V$ and propagates it to the tip location at point 0. The operator $\phi$ is lower block triangular, which we denote as "causal," making $\phi^*$ upper block triangular and hence "anticausal." $\phi^*$ represents a propagation of link velocities from the base to the tip, which is viewed as the anticausal direction, as opposed to the tip-to-base recursion represented by $\phi$ which is denoted as causal.

The action of the Jacobian operator on the joint angle rates $\dot{\theta}$ then is as follows: $H^*\dot{\theta}$ results in relative spatial velocities between the links along the joint axes: The action of $\phi^*$ then anticausally propagates these relative velocities from the base to the tip to form the link spatial velocities $V = col[V(1), \cdots, V(N)]$; $B^*$ then projects out $V(1)$ from $V$ and propagates it to the tip forming $V(0)$.

The well-known dual relationship to $V(0) = J\dot{\theta}$ is $T = J^*f(0) = H\phi Bf(0)$, where $f(0) = col[N(0), F(0)]$ is a spatial force which represents the tip interaction with the environment [14]. The action of $J^*$ on $f(0)$ is as follows: $B$ takes $f(0)$ to $col[f(1), 0, \cdots, 0]$. The effect of $\phi$ is to propagate $f(1)$ causally from link 1 to the base forming the spatial forces acting at each link represented by $f = col[f(1), \cdots, f(N)]$. Finally, $H$ projects each component of $f$, $f(k)$, onto joint axis $H^T(k) = h(k)$ to obtain the joint moments $T = col[T(1), \cdots, T(N)]$.

The key points to note here are that $J$ and $J^*$ have operator factorizations which have immediate physical interpretations and obvious recursive algorithmic equivalents. Working with the factorized version of $J$, one can manipulate expressions involving $J$ in novel ways while maintaining the physical insight provided by the factors and the ability to produce equivalent recursive algorithms at key steps of a calculation. For example, using the techniques of the spatial operator algebra, one can find algorithms for efficient recursive construction of $J$, $JJ^*$, $J^*J$, and (when an arm is nonredundant and nonsingular) $(J^*J)^{-1}$. See [5] and [15].

## 3. An Operator Formulated Robot Dynamics

Consider the following equations of motion for a serial-link manipulator in a gravity-free environment with the tip imparting a spatial force $f(0)$ to the external environment:

$$\mathcal{M}\ddot{\theta} + \mathcal{C} + J^*f(0) = T \tag{3.1}$$

$\mathcal{C}$ denotes "bias" torques due to the velocity dependent coriolis and centrifugal effects. Equation (3.1) is precisely the form that arises from a Lagrangian analysis of manipulator dynamics. Equation (3.1) has an operator interpretation which arises from the following spatial operator factorizations of $\mathcal{M}$, $\mathcal{C}$, and $J^*$

$$\mathcal{M} = H\phi M\phi^*H^* \tag{3.2a}$$

$$\mathcal{C} = H\phi(M\phi^*a + b) \tag{3.2b}$$

$$J^* = H\phi B \tag{3.2c}$$

These factorizations are derived in [1,3,6,8]. The quantity

$$M = diag[M(1), \cdots, M(N)]$$

is made up of the spatial inertia $M(k)$ associated with each link of the manipulator. $M$, being block diagonal, is interpreted as a memoryless operator. For a given link $k$, $M(k)$ has the form

$$M(k) = \begin{pmatrix} I(k) & m(k)\widetilde{p}(k) \\ -m(k)\widetilde{p}(k) & m(k)\mathcal{I} \end{pmatrix}$$

where $I(k)$ is the inertia tensor of link $k$ about joint $k$, $m(k)$ is the link $k$ mass, and $p(k)$ is the 3-vector from joint $k$ to the link $k$ mass center. The "tilde" operator is defined by $\widetilde{x}y = x \times y$ for any 3-vectors $x$ and $y$. In (3.2b), $a = col[a(1), \cdots, a(N)]$ and $b = col[b(1), \cdots, b(N)]$ are quadratic functions of the link spatial velocities. The operators $H$, $\phi$, and $B$ were described in the previous section.

When (3.1) is given an operator interpretation via (3.2), it is immediately apparent that (3.1) is functionally identical to the Newton-Euler recursions given by [8,14,16]

```
                    $\alpha(N+1) = 0$
LOOP $k = N, \cdots, 1;$
                    $\alpha(k) = \phi^T(k+1,k)\alpha(k+1) + H^T(k)\ddot{\theta}(k) + a(k)$
END LOOP;
                    $f(0) = f_{ext}$
LOOP $k = 1, \cdots, N;$
                    $f(k) = \phi(k,k-1)f(k-1) + M(k)\alpha(k) + b(k)$
                    $T(k) = H(k)f(k)$
END LOOP;
```

where $\alpha = col[\alpha(1), \cdots, \alpha(N)]$, and $\alpha(k) = \dot{V}(k)$ denotes the spatial acceleration of link $k$.

It is now evident that the equivalence between the Lagrangian and recursive Newton-Euler formulations of manipulator dynamics [17] has been made trivial. Furthermore, the factorizations given by (3.2) allow us to manipulate the dynamical equations of motion in ways not previously apparent. The fact that each factor has an interpretation as a causal, memoryless, or anticausal recursion of spatial quantities means that at any point of the mathematical analysis one can interpret expressions in a deeply physical way or immediately produce an equivalent recursive algorithm.

The true power of the spatial algebra applied to manipulator dynamics will become clearer in the following sections. It will be shown that an important alternative factorization to (3.2a) exists which results in new causal, memoryless, anticausal operators with corresponding equivalent recursions. Also, we will discuss the existence of powerful operator identities which allow one to manipulate kinematical and dynamical equations in ways which would be otherwise impossible, all the while keeping the correspondence of abstract mathematical expressions to equivalent implementable algorithms.

## 4. Operator Inversion of the Manipulator Mass Matrix

From (3.2a), the well known fact that $\mathcal{M}$ is symmetric positive definite can be easily shown. It is also well-known that a symmetric positive definite operator is a covariance for some Gaussian random process. A deeper result is that the factorization given by (3.2a) shows that $\mathcal{M}$ has the structure of a covariance of the output of a discrete-step causal finite-dimensional linear system whose input is a Gaussian white-noise process. This a very important fact, for it is well-known that such an operator can be factored and inverted efficiently by the use of standard techniques from filtering and estimation theory. Applications of these techniques to the manipulator mass matrix can be found in [1,3,6,8] and are partially summarized in this section.

First, we present an important alternative factorization to (3.2a). To this end, we define

$$S \triangleq \begin{pmatrix} 0 & & & & 0 \\ \mathcal{I} & 0 & & & 0 \\ 0 & \mathcal{I} & 0 & & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots \quad \mathcal{I} & 0 \end{pmatrix} \quad \text{and} \quad \Phi \triangleq \phi S = \begin{pmatrix} 0 & & & & 0 \\ \phi(2,2) & 0 & & & 0 \\ \phi(3,2) & \phi(3,3) & 0 & & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi(N,2) & \phi(N,3) & \phi(N,4) & \cdots \quad \phi(N,N) & 0 \end{pmatrix}$$

Also define

$$\Delta_\phi \triangleq diag[\phi(2,1),\cdots,\phi(N+1,N)]$$

and

$$D \triangleq HPH^*, \quad G \triangleq PH^*D^{-1}, \quad L \triangleq \Delta_\phi G,$$

where $P = diag[P(1),\cdots,P(N)]$. The diagonal elements $P(k)$ are obtained by the following causal discrete-step Riccati equation

$$P(1) = M(1)$$

LOOP $k = 2,\cdots,N$;

$$P(k) = \psi(k,k-1)P(k-1)\psi^T(k,k-1) + M(k)$$

END LOOP;

where

$$\psi(k,k-1) \triangleq \phi(k,k-1) - L(k-1)H(k-1) \tag{4.1}$$

$P(k)$ is always symmetric positive definite and hence $D$, which is diagonal with the positive diagonal elements $D(k) = H(k)P(k)H^T(k)$, is always invertible.

In an analogous fashion to the definitions of the operators $\phi$ and $\Delta_\phi$ we define the operators $\psi$ and $\Delta_\psi$ with the component operators $\psi(k,k-1)$ being given by Eq. (4.1), $\psi(k,k) \triangleq \mathcal{I}$, and

$$\psi(i,j) \triangleq \psi(i,i-1)\psi(i-1,i-2)\cdots\psi(j+1,j)$$

for $i \geq j$. Note that the $\psi(i,j)$'s form a semigroup.

With these definitions, we have that

$$\Delta_\psi = \Delta_\phi(I - GH)$$

**Lemma 4.1:** An alternative factorization of $\mathcal{M} = H\phi M\phi^* H^*$ is

$$\mathcal{M} = (\mathcal{I} + H\Phi L)D(\mathcal{I} + H\Phi L)^* \tag{4.2}$$

where $\mathcal{I} + H\Phi L$ is causal (lower triangular), and $D$ is memoryless, diagonal and invertible.

**Proof:** See Appendix.

The factorization (4.2) is equivalent to viewing the mass operator $\mathcal{M}$ as the covariance of a filtered innovations process, $Y$. In stochastic estimation theory, the innovations representation is given by the causal operator $\mathcal{I} + H\Phi L$ operating on an innovations process $\epsilon = diag[\epsilon(1),\cdots,\epsilon(N)]$ which can be taken to be an independent Gaussian sequence. The action of $\mathcal{I} + H\Phi L$ on $\epsilon$,

$$Y = (\mathcal{I} + H\Phi L)\epsilon$$

is equivalent to a causal tip-to-base recursion. The importance of the innovations operator $\mathcal{I} + H\Phi L$ is that it is trivially and causally invertible and that its inverse is precisely a discrete-step Kalman filter viewed as a whitening filter. To demonstrate this, define

$$\Psi \triangleq \psi S$$

**Lemma 4.2:** The causal (lower triangular) operators $\mathcal{I} + H\Phi L$ and $\mathcal{I} - H\Psi L$ are mutual causal inverses of each other

$$(\mathcal{I} + H\Phi L)^{-1} = \mathcal{I} - H\Psi L \tag{4.3}$$

**Proof:** See Appendix.

The relationship $\epsilon = (\mathcal{I} + H\Phi L)^{-1}y = (\mathcal{I} - H\Psi L)y$ is equivalent to a causal tip-to-base recursion. This recursion is precisely a discrete-step Kalman filter. Lemmas 4.1 and 4.2 result in

**Theorem 4.1:** The operator $\mathcal{M}^{-1}$ has the following anticausal-memoryless-causal operator factorization

$$\mathcal{M}^{-1} = (\mathcal{I} - H\Psi L)^* D^{-1} (\mathcal{I} - H\Psi L) \tag{4.4}$$

Application of Theorem 4.1 to the bias-free robot equations of motion given by Equation (3.3) immediately yields the following $O(N)$ forward dynamics algorithm

**Algorithm FD.**

$$T' = T - H\phi[M\phi^* a + b + Bf(0)] \tag{4.5a}$$

$$\ddot{\theta} = (\mathcal{I} - H\Psi L)^* D^{-1} (\mathcal{I} - H\Psi L) T' \tag{4.5b}$$

Equation (4.5a) represents an $O(N)$ Newton-Euler recursion to remove the bias torques. Equation (4.5b) leads to the following $O(N)$ recursive algorithm

```
                    $\hat{z}(0);\quad T'(0) = 0$
LOOP $k = 1, \cdots, N$;
                    $\hat{z}(k) = \psi(k, k-1)\hat{z}(k-1) + L(k-1)T'(k-1)$
                    $\epsilon(k) = T'(k) - H(k)\hat{z}(k)$
                    $\nu(k) = D^{-1}(k)\epsilon(k)$
END LOOP;
                    $\lambda(N+1) = 0$
LOOP $k = N, \cdots, 1$;
                    $\lambda(k) = \psi^T(k+1, k)\lambda(k+1) + H^T(k)\nu(k)$
                    $\ddot{\theta}(k) = \nu(k) - L^T(k)\lambda(k+1)$
END LOOP;
```

It can be shown that the forward dynamics algorithm given by Equations (4.5) is equivalent to that of [18], but derived by vastly different means. Similarly, it can be shown that $P(k)$ defined above is an articulated body inertia as defined by [18], but discovered independently, and in a much different context, in [1].

In addition to the operator factorizations at our disposal, there exist a multitude of operator identities relating the various operator factors. This greatly enhances the ability to obtain any number of important results. For instance, it is shown in [8] how these identities can be used to obtain a variety of $O(N)$ forward dynamics algorithms, all of them significantly different. It is seen that the algorithm given by Equations (4.5) above is but one in a whole class of such algorithms available from an application of the spatial operator algebra. Furthermore, in [8] it is shown how these algorithms can be easily extended to the closed-chain system made up of several arms grasping a common rigid object.

## 5. Applications of Spatial Operator Identities

Above, we have referred to the availability of identities relating elements of the spatial operator algebra. In [8], many such relationships are derived. In this section, we will focus on the application of one such identity as representative of how these identities can be used to perform high-level manipulations which result in novel algorithms useful in dynamical analysis and control.

The identity of interest is

**Identity 5.1:**

$$(\mathcal{I} - H\Psi L)H\phi = H\psi \tag{5.1}$$

**Proof:** See Appendix.

The action of $\psi$ on a composite spatial quantity $Z = \psi Y$ is equivalent to a causal tip-to-base recursion.

**Application 1: Tip Force Correction Accelerations**

From Eq. (3.1) it is evident that

$$\ddot{\theta} = \ddot{\theta}_f + \Delta\ddot{\theta}$$

where

$$\ddot{\theta}_f = \mathcal{M}^{-1}(T - \mathcal{C}) = \mathcal{M}^{-1}T'$$

can be determined from the forward dynamics algorithm (4.5). Our first application of Identity (5.1) is to find a simple relationship between tip contact forces and the resulting joint accelerations, $\Delta\ddot{\theta}$, due solely to such tip forces. From Eqs. (3.1) and (3.2), we have $\mathcal{M}\Delta\ddot{\theta} = -J^* f(0)$ or

$$H\phi M\phi^* H^* \Delta\ddot{\theta} = -H\phi B f(0)$$

With Equation (4.4) this becomes

$$\Delta\ddot{\theta} = -(\mathcal{I} - H\Psi L)^* D^{-1}(\mathcal{I} - H\Psi L)H\phi B f(0) \tag{5.2}$$

Application of Identity (5.1) then results in

$$\Delta\ddot{\theta} = -(\mathcal{I} - H\Psi L)^* D^{-1} H\psi B f(0) \tag{5.3}$$

Eq. (5.3) is significantly simpler than (5.2). It shows how the effect of the tip force propagates from the tip to the base of a manipulator, producing link accelerations which then propagate from the base to the tip.

**Application 2: Composite Body Inertia Reflected to the Manipulator Tip**

The next application of Identity (5.1) will be to produce an $O(N)$ recursive algorithm for computing the Operational Space mass matrix $\Lambda$ of Khatib [13]. Knowledge of $\Lambda$, together with the Operational Space coriolis, centrifugal, and gravity terms, enables the use of Operational Space Control - a form of feedback linearizing control described in [13]. The ability to obtain the Operational Space dynamics recursively avoids the need to have explicit analytical expressions which can be quite complex. Although we will only discuss the recursive construction of the Operational Space mass matrix $\Lambda$, the entire Operational Space dynamics can be computed via $O(N)$ recursions using the techniques of the spatial operator algebra, allowing for recursive implementation of Operational Space Control.

If the dynamics of an $N$-link manipulator are reflected to the tip locations, the resulting composite body inertia has the form

$$\Lambda = (J\mathcal{M}^{-1}J^*)^{-1}$$

For a manipulator whose workspace is $R^6$, the inversion of the $6 \times 6$ operator $J\mathcal{M}^{-1}J^*$ entails a constant cost which is independent of the number of manipulator links. The real work is to obtain an efficient $O(N)$ algorithm for the construction of $\Omega(0) = J\mathcal{M}^{-1}J^*$. Equations (1.1) and (4.4) reveal that

$$\Omega(0) = J\mathcal{M}^{-1}J^* = B^*\phi^*H^*(\mathcal{I} - H\Psi L)^* D^{-1}(\mathcal{I} - H\Psi L)H\phi B \tag{5.4}$$

Application of Identity (5.1) to Eq. (5.4) immediately results in

$$\Omega(0) = J\mathcal{M}^{-1}J^* = B^*\psi^*H^*D^{-1}H\psi B \tag{5.5}$$

It is quite straightforward [8] to show that the following $O(N)$ anticausal base-to-tip recursive algorithm is equivalent to (5.5)

$$\Omega(N+1) = 0$$

LOOP $k = N, \cdots, 1$;

$$\Omega(k) = \psi^T(k+1,k)\Omega(k+1)\psi(k+1,k) + H^T(k)D^{-1}(k)H(k)$$

END LOOP;

$$\Omega(0) = \phi^T(1,0)\Omega(1)\phi(1,0)$$

**Application 3: Closed Chain Forward Dynamics**

Figure 1a represents a closed chain of rigid bodies connected by revolute joints which are all actuated. Figure 1a can be viewed as a graph whose nodes are links and whose edges are joints. A spanning tree can be found for this graph which is equivalent to cutting the chain at some point, say point $c$ of Figure 1a. The root of this tree is indicated by the arrow.

Imagine that the chain is physically cut at $c$ and designate the root link to be the "Base." This results in Figure 1b. For simplicity, assume that the base is immobile. This assumption results in no real loss of generality – see, e.g., ref. [8]. Cutting the chain has resulted in arms 1 and 2 with $N_1$ and $N_2$ links respectively. We can now assign the causal/anticausal directions to each arm. (Note that this assignment propagated back to Fig. 1a corresponds to the existence of a directed graph associated with Fig. 1a.)

The fact that the tips of arms 1 and 2 are always constrained to remain in contact corresponds to the boundary conditions

$$f_2(0) = -f_1(0) \equiv f(0) \qquad (5.6a)$$

$$\alpha_1(0) = \alpha_2(0) \qquad (5.6b)$$

With (5.6a), the dynamical behavior of arms 1 and 2 is given by

$$\mathcal{M}_1\ddot{\theta}_1 + \mathcal{C}_1 = T_1 + J_1^* f(0) \quad , \quad \mathcal{M}_2\ddot{\theta} + \mathcal{C}_2 = T_2 - J_2^* f(0) \qquad (5.7)$$

subject to (5.6b). Note from $V(0) = J\dot{\theta}$ and $\alpha(0) = \dot{V}(0)$ that

$$\alpha_1(0) = J_1\ddot{\theta}_1 + \dot{J}_1\dot{\theta}_1 \quad , \quad \alpha_2(0) = J_2\ddot{\theta}_2 + \dot{J}_2\dot{\theta}_2 \qquad (5.8)$$

From (5.7)

$$\dot{\theta}_1 = \ddot{\theta}_{1f} + \Delta\ddot{\theta}_1 \quad , \quad \ddot{\theta}_2 = \ddot{\theta}_{2f} + \Delta\ddot{\theta}_2 \qquad (5.9)$$

where

$$\ddot{\theta}_1 = \mathcal{M}_1^{-1}(T_1 - \mathcal{C}_1) \quad , \quad \ddot{\theta}_2 = \mathcal{M}_2^{-1}(T_2 - \mathcal{C}_2) \qquad (5.10)$$

and

$$\Delta\ddot{\theta}_1 = \mathcal{M}_1^{-1}J_1^* f(0) \quad , \quad \Delta\ddot{\theta}_2 = \mathcal{M}_2^{-1}J_2^* f(0) \qquad (5.11)$$

If $f(0)$ can be determined, $\ddot{\theta}_1$ and $\ddot{\theta}_2$ can be found from (5.9)-(5.10) via the recursive algorithms presented earlier.

To find $f(0)$, define

$$\alpha_{1f}(0) = J_1\ddot{\theta}_{1f} + \dot{J}_1\dot{\theta}_1 \quad , \quad \alpha_{2f}(0) = J_2\ddot{\theta}_f + \dot{J}_2\dot{\theta}_2 \quad , \quad \Delta\alpha_f(0) = \alpha_{1f}(0) - \alpha_{2f}(0) \qquad (5.12)$$

These quantities can be computed via $O(N_1)$ and $O(N_2)$ recursive algorithms [8]. Eqns. (5.8) and (5.12) yield

$$\alpha_1(0) = \alpha_{1f}(0) + J_1\Delta\ddot{\theta}_1$$

$$\alpha_2(0) = \alpha_{2f}(0) + J_2\Delta\ddot{\theta}_2$$

which with the boundary condition (5.6b) gives

$$\Delta\alpha_f(0) = J_1\Delta\ddot{\theta}_1 - J_2\Delta\ddot{\theta}_2 \quad . \qquad (5.13)$$

From (5.11), (5.13) becomes

$$\Delta\alpha_f(0) = (\Lambda_1^{-1} + \Lambda_2^{-1})f(0) \tag{5.14}$$

where

$$\Lambda_1^{-1} = J_1\mathcal{M}_1^{-1}J_1^* \quad , \quad \Lambda_2^{-1} = J_2\mathcal{M}_2^{-1}J_2^* \tag{5.15}$$

Thus

$$f(0) = \Lambda_c\Delta\alpha_f(0) \tag{5.16a}$$

$$\Lambda_c^{-1} \equiv \Lambda_1^{-1} + \Lambda_2^{-1} \tag{5.17a}$$

As discussed previously, $\Lambda_1^{-1}$ and $\Lambda_2^{-1}$ can be found via $O(N_1)$ and $O(N_2)$ recursive algorithms respectively. Noting that the inversion of $\Lambda_c^{-1}\epsilon R^{6\times 6}$ involves a flat cost independent of $N_1$ and $N_2$, we see that we have produced an $O(N_1 + N_2)$ recursive algorithm for finding the forward dynamics of the system of Figure 1a. $\Lambda_c$ is the composite body inertia of the closed chain system reflected to point $c$.

The spatial algebra perspective enables the generation of efficient recursive algorithms for computing the composite body inertia of a system of several arms grasping a common object which is of complexity $O(N) + O(\ell)$, when no arm is at a kinematical singularity, or, more generally, $O(N) + O(\ell^3)$, where $N$ is the total number of links in the system and $\ell$ is the number of arms grasping the object.

For additional applications of the spatial operator algebra similar to those of this section, see for example [5,8]. In [5] an operator expression for $(J^*J)^{-1}$ is obtained for nonredundant arms which is used in a recursive scheme for solving the manipulator inverse kinematics problem. In [8], many additional examples may be found along with an extensive listing of operator identities. For instance, in [8] it is shown how one can easily find the composite body inertia matrix for a system consisting of several arms grasping a commonly held rigid body.

## 6. Conclusions

A powerful new spatial operator algebra for describing the kinematical and dynamical behavior of multibody systems has been presented. Abstract dynamical equations of motion, such as arise from a Lagrangian analysis, can be reinterpreted as equivalent operator formulated equations. From this perspective, the distinction between abstract expressions, the interlink physical relationships of spatial quantities (velocities, accelerations, and forces), and recursive algorithms which propagate spatial quantities from link-to-link entirely vanishes. One consequence of the operator algebra is that the equivalence between the Lagrangian and Newton-Euler formulations of dynamics is trivial and transparent.

Important elements of the spatial operator algebra were presented, in particular those which arise from natural factorizations of critical kinematical and dynamical quantities. These factorizations allow one to manipulate equations of motion in hitherto unknown ways, greatly increasing our powers of analysis. This is particularly true given the existence of important identities and inversions which relate the spatial operators. A key result is the operator factorization and inversion of the manipulator composite body inertia given by Fact 4.1 and Theorem 4.1.

Various applications of the spatial algebra to kinematics, dynamics, and control were presented, including the development of a recursive forward dynamics algorithm, which essentially comes for free once the key step of obtaining the innovations factorization (4.1) is seen.

The potential payoff of the spatial algebra in terms of providing a framework which can manage the complexity associated with multibody systems is immense. For example, compare the abstract simplicity of the development of the forward dynamics algorithm in this paper with those developed by other means which often require a morass of notation and development. In Sec. 5, we touched only lightly on some of the current areas where the spatial operator algebra is being applied. We believe that this algebra can provide a complete framework for describing multibody systems. This will greatly aid in the ultimate generation of "smart" programs which can reason about the behavior of the dynamical world by the use of a suitable hierarchy of abstraction.

## 7. Acknowledgement

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

## 8. References

[1] G. Rodriguez, "Kalman Filtering, Smoothing, and Recursive Robot Arm Dynamics," JPL Publication 86-48, Dec. 1986.

[2] G. Rodriguez, "Filtering and Smoothing Approach to Dual Robot Arm Dynamics," Proc. $1^{st}$ Int. Symp. on Robotics and Manufacturing, Albuquerque, NM, Nov. 1986.

[3] G. Rodriguez, "Spatially Random Models, Estimation Theory, and Robot Arm Dynamics," IEEE Symp. on Intelligent Control, Philadelphia, PA, Jan. 1987.

[4] G. Rodriguez, "Kalman Filtering, Smoothing, and Topological Tree Dynamics," VPI/SU Symp. on Control of Large Structure, Blacksburg, VA. June 1987.

[5] G. Rodriguez and R. E. Scheid, Jr., "Recursive Inverse Kinematics for Robot Arms via Kalman Filtering and Bryson-Frazier Smoothing," AIAA Guidance, Navigation, and Control Conference, Monterey, CA, Aug. 1987.

[6] G. Rodriguez, "Kalman Filtering, Smoothing, and Recursive Robot Arm Forward Dynamics," IEEE J. Robotics and Automation, Vol. RA-3, Dec. 1987.

[7] G. Rodriguez, "Recursive Forward Dynamics for Multiple Robot Arms Moving a Common Task Object," JPL Publication 88-6, Feb. 1988.

[8] G. Rodriguez and K. Kreutz, "Recursive Mass Matrix Factorization and Inversion: An Operator Approach to Open- and Closed-Chain Multibody Dynamics," JPL Publication 88-11, March 1988.

[9] K. Kreutz and A. Lokshin, "Load Balancing and Closed–Chain Multiple Arm Control," 1988 American Control Conference, Atlanta, GA, June 1988.

[10] G. Rodriguez, M. Milman and K. Kreutz, "Dynamics and Coordination of Multiple Robot Arms," $3^{rd}$ Int. Conf. on CAD/CAM and Robotics, Detroit, MI, Aug. 1988.

[11] P. Roman, Modern Mathematics for Physicists and Other Outsiders, Pergamon Press, New York, 1975.

[12] W. Rudin, Functional Analysis, McGraw-Hill, New York, 1973.

[13] O. Khatib, "The Operational Space Formulation in the Analysis, Design, and Control of Manipulators," $3^{rd}$ Int. Symp. Robotics Research, Paris, 1985.

[14] J. Craig, Introduction to Robotics, Addison-Wesley, Reading, Mass., 1986.

[15] K. Kreutz, "Ada Packages Supporting the Dual Arm Robot Programming Library," JPL Engr. Memo. 347-88-245 (internal document), June 1988.

[16] J. Y. S. Luh, M. Walker, and R. Paul, "On-line Computational Scheme for Mechanical Manipulators," J. Dyn. Sys. Meas. and Control, Vol. 120, 1980.

[17] W. Silver, "On the Equivalence of the Lagrangian and Newton-Euler Dynamics for Manipulators," Int. J. Robotics Research, Vol. 1, 1982.

[18] R. Featherstone, "The Calculation of Robot Dynamics Using Articulated- Body Inertias," Int. J. Robotics Research, Vol. 2, 1983.

## Appendix

Prior to proving Lemma 4.1, Lemma 4.2 and Identity 5.1, we define

$$\tilde{\phi} \triangleq \phi - \mathcal{I} \qquad \text{and} \qquad \tilde{\psi} \triangleq \psi - \mathcal{I}$$

and establish the following identities.

**Lemma A1:**

(a) $$\psi^{-1} - \phi^{-1} = SLH \tag{A.1}$$

(b) $$\phi\psi^{-1} = I + \Phi LH$$

$$\psi^{-1}\phi = I + SLH\phi \tag{A.2}$$

(c) $$\tilde{\psi}PH^* = 0 \tag{A.3}$$

(d) $$\psi M \psi^* = P + \tilde{\psi}P + P\tilde{\psi}^* \tag{A.4}$$

(e) $$H\psi M \psi^* H^* = D \tag{A.5}$$

**Proof:**

(a) It is easy to verify that

$$\tilde{\phi} = \Phi\Delta_\phi \qquad \text{and} \qquad \tilde{\psi} = \Psi\Delta_\psi$$

It then follows that

$$\phi^{-1} = I - S\Delta_\phi \qquad \text{and} \qquad \psi^{-1} = I - S\Delta_\psi \tag{A.6}$$

Thus, $\psi^{-1} - \phi^{-1} = S(\Delta_\phi - \Delta_\psi)$. However, we have seen from the definition of $\Delta_\psi$ that

$$\Delta_\psi = \Delta_\phi(I - GH) \tag{A.7}$$

and the result follows immediately.

(b) These two identities are obtained by respectively pre- and post-multiplying (A.1) by $\phi$.

(c) It follows from the definition of $G$ that $(I - GH)PH^* = 0$. Thus, using (A.7)

$$\tilde{\psi}PH^* = \Psi\Delta_\psi PH^* = \Psi\Delta_\phi(I - GH)PH^* = 0$$

(d) From the definition of $P$ it follows that $P$ satisfies the following equation:

$$M = P - (S\Delta_\psi)P(S\Delta_\psi)^*$$

which using (A.6) and some simple manipulations yields the result.

(e) We have that,

$$\begin{aligned} H\psi M\psi^* H^* &= H(P + \tilde{\psi}P + P\tilde{\psi}^*)H^*, \quad \text{using (A.4)} \\ &= HPH^*, \quad \text{using (A.3)} \\ &= D \end{aligned}$$

**Proof of Lemma 4.1:**

$$\begin{aligned} \mathcal{M} &\triangleq H\phi M\phi^* H^* \\ &= H(\phi\psi^{-1})\psi M\psi^*(\phi\psi^{-1})^* H^* \\ &= (I + H\Phi L)H\psi M\psi^* H^*(I + H\Phi L)^*, \quad \text{using (A.2)} \\ &= (I + H\Phi L)D(I + H\Phi L)^*, \quad \text{using (A.5)} \end{aligned}$$

**Proof of Lemma 4.2:**

$$\begin{aligned} (I + H\Phi L)^{-1} &= (I + H\phi SL)^{-1} \\ &= I - H\phi(I + SLH\phi)^{-1}SL, \quad \text{using a standard matrix identity} \\ &= I - H\phi(\psi^{-1}\phi)^{-1}SL, \quad \text{using (A.2)} \\ &= I - H\psi SL \quad = I - H\Psi L \end{aligned}$$

**Proof of Identity 5.1:**

Premultiplying (A.1) by $\psi$ we see that $\psi\phi^{-1} = I - \Psi LH$. Hence,

$$\begin{aligned} (I - H\Psi L)H\phi &= H(I - \Psi LH)\phi \\ &= H(\psi\phi^{-1})\phi \quad = H\psi \end{aligned}$$

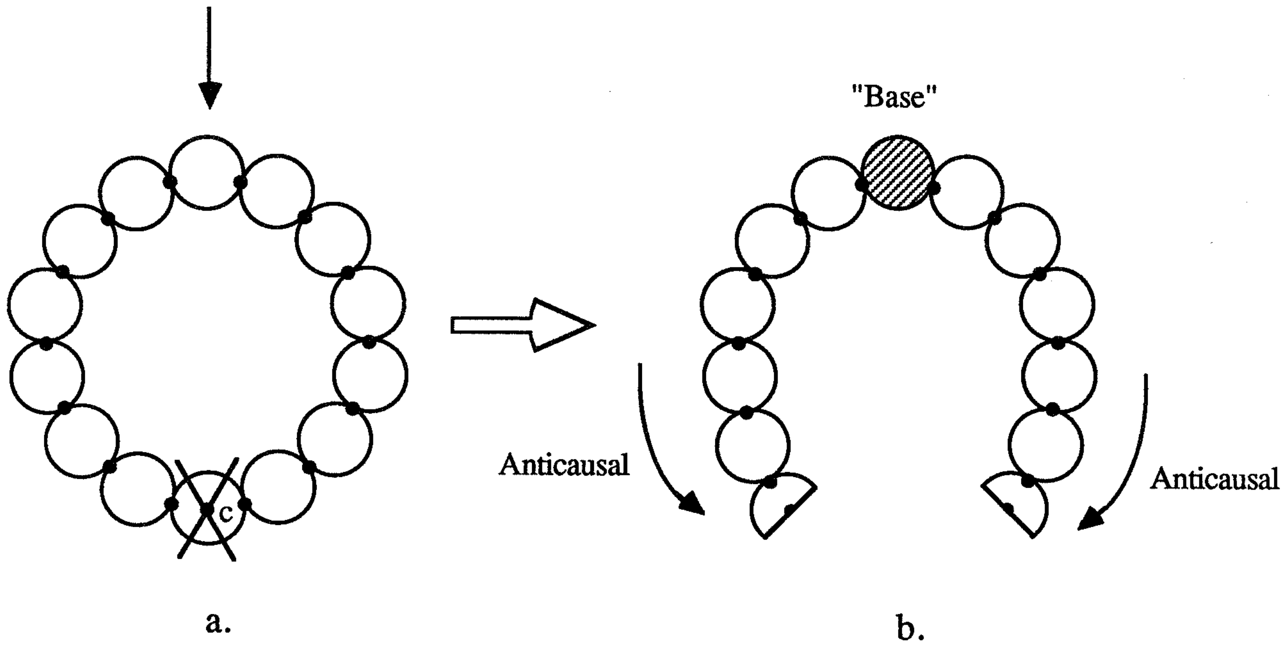


Figure 1: Closed Chain System

# FLIGHT EXPERIMENT CONCEPTS

# FLIGHT EXPERIMENTS IN TELEROBOTICS-ORBITER MIDDECK CONCEPT

Lyle M. Jenkins
Johnson Space Center
Houston, TX 77058

## Abstract

Most uncertainties of operation of a telerobot in the space environment relate to the absence of gravity effects and not to the vacuum effects. A flight experiment concept is proposed for the middeck of the Space Shuttle that provides direct access for the crew. Telerobot dexterous manipulation issues in task performance, mechanism response, system duty cycles and operator interface can be effectively addressed. A pair of replica-type master controllers would be adapted for slave manipulator functions. A variety of test setups and control modes can obtain data on zero g operation of a telerobot.

## 1. Introduction

The operation of a telerobot in the space environment will bring up a number of issues that cannot be effectively addressed in ground-based simulations [1].

The difficulty and expense of flight testing is an effective deterrent to research in the vacuum and zero g. Even so, the risk in applying telerobotics to dexterous manipulation tasks can be reduced by validating simulations and answering questions of task performance, mechanism design, system flexibility and interface with the operator in a flight test. The majority of flight test issues relate to zero g and not to vacuum effects. This provides an opportunity to experiment in the middeck of the Space Shuttle cabin. The proposed concept involves replica-type master controllers, developed for good bilateral force reflection, adapted for slave manipulator function. Through a variety of control modes, many significant test objectives can be resolved. With direct access to the task site, the crew can change the test setups and accommodate mistakes in task performance to reduce the probability of test failure.

## 2. Experiment Objectives

The objective of a flight experiment is to develop data for resolution or support of research issues that cannot be adequately simulated in ground-based laboratories. For a telerobot, these issues can be categorized as task performance, manipulator characterization and operator interface. Tasks involve some degree of dexterous manipulation. Maneuvering and positioning orbital replacement units (ORU), attachment of structural fasteners, electrical connectors, fluid transfer lines, and handling of tools are examples of tasks. Operations in zero g may also involve unique provisions for object retention, containing contaminents or dealing with large surface areas such as insulation.

The mechanical functions of a manipulator may change in zero g, affecting positional accuracy and repeatability and cannot be effectively simulated in ground-based tests. The interaction of joint and arm flexibility with control may be significant in response to forces generated in task performance. The direction and magnitude of force application may change to a degree that will affect the accomplishment of a specific task. Dual-arm activities are certainly a question when gravity is not acting on the arms and workpiece.

The operator's control of the manipulators in zero g cannot be adequately simulated on Earth. Position control modes and bilateral force reflection control are particularly sensitive to operator restraints and controller configuration. Data must be developed on these interactions including dual-arm control. Viewing the worksite, whether direct or by TV, has not been evaluated under space flight conditions.

There are integrated telerobotic system issues that exhibit a high degree of uncertainty in the transition from ground to space flight operations. The manipulator duty-cycles with power requirements and heat dissipation should be measured when performing tasks in zero g. Actual flight testing should also develop the realities of training and operations integration in the use of a remote operating system for dexterous manipulation.

## 3. Concept Description

Several potential experiment concepts are described in reference [2]. These include a fixed-base telerobot attached to a carrier structure in the orbiter payload bay (Figure 1), a telerobot positioned by the Space Shuttle remote manipulator system (RMS) (Figure 2), and a telerobot representation in the middeck of the orbiter. The middeck flight experiment concept is derived from the performance of a force-reflecting "mini-master" controller and extrapolating that performance to zero g (Figure 3). The flight experiment uses two of the controllers set up in the middeck of the orbiter cabin. The controllers are modified to operate in a slave mode as well as in a master control mode. Three combinations of operation are proposed:

1. A workstation with dual arm controllers that controls a computer simulation with synthetic force input fed to the controllers (Figure 4).

2. Master controller driving the other controller as a slave in a bilateral force reflecting mode (Figure 5).

3. Dual-arm slave manipulators controlled by computer (Figure 6).

The test combinations would be set up in various combinations of workstation and worksite configurations to evaluate task performance.

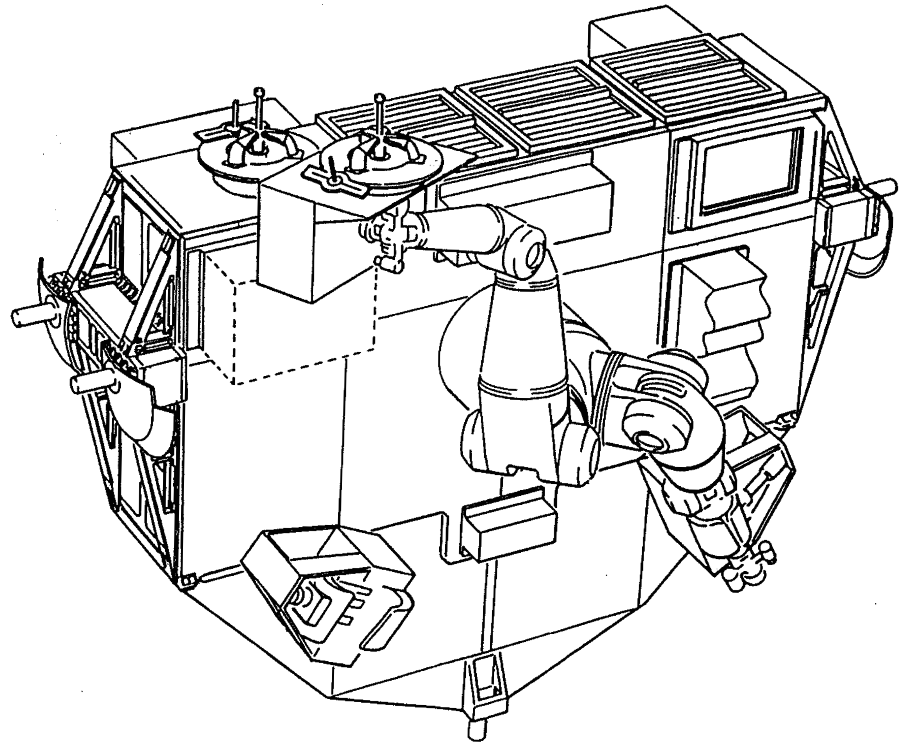


Figure 1.- Fixed-base telerobot in Orbiter payload bay.

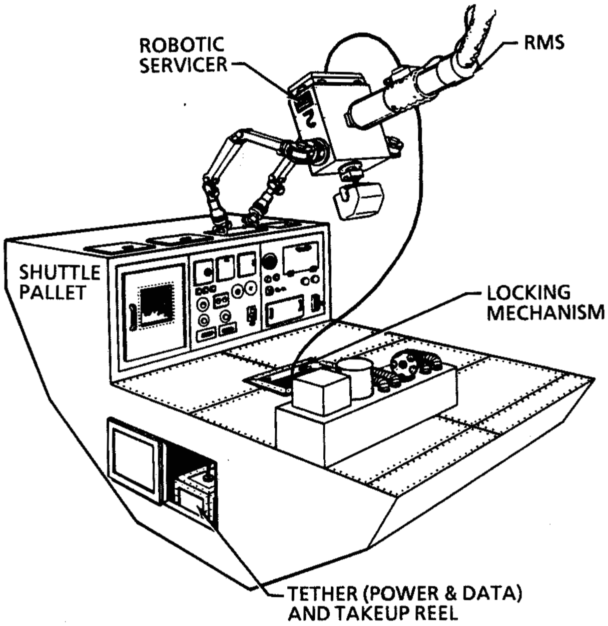


Figure 2.- RMS-positioned telerobot.

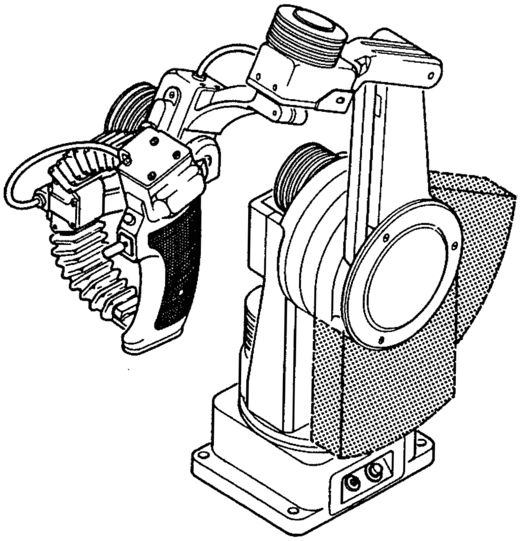


Figure 3.- Force reflecting mini-master controller.

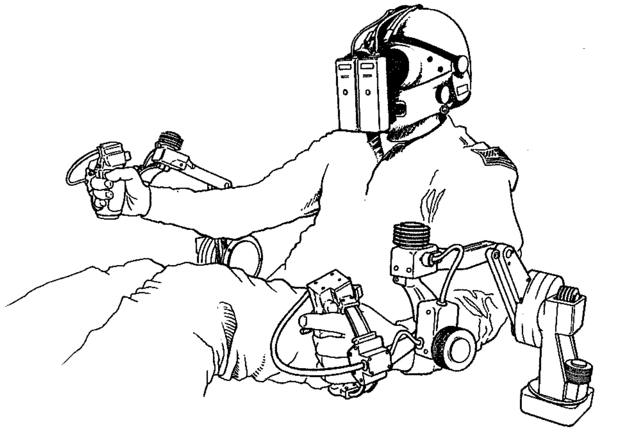


Figure 4.- Dual-arm controllers.

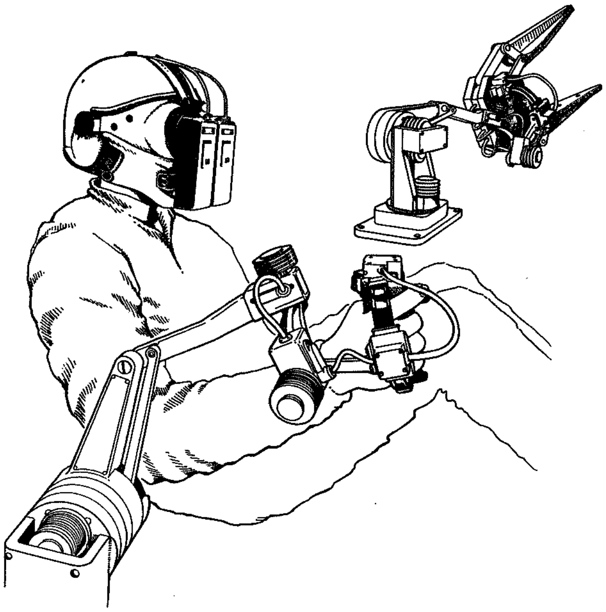


Figure 5.- Master controller and slave arm.

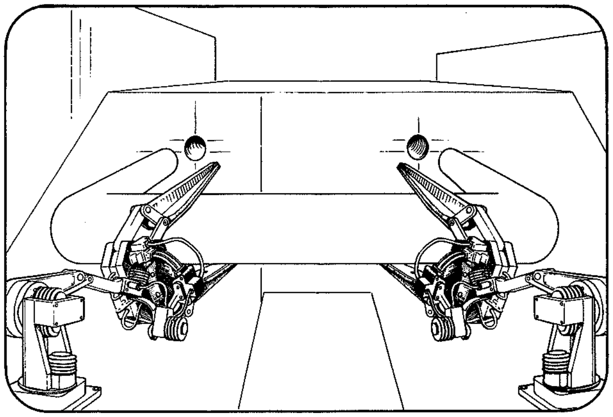


Figure 6.- Dual-arm slave manipulators.

## 4. Experiment Equipment Requirements

The primary equipment needed is a pair of mini-masters and their supporting electronics as controller/manipulators. End effectors must be installed on the controllers for the slave manipulator mode. A computer supports the controller/manipulators as well as an interface for the dual arm control simulation and a driver for the dual slave arm setup. Displays for the operator may be incorporated in a helmet for portability (Figure 7). Restraint systems for the operator should include several variations to provide a comparison of degrees of support for reacting controller loads (Figure 8). Closed circuit television functions to provide indirect viewing for the operator as well as to document most of the test results (Figure 9). The final piece of primary equipment is the task board.

Secondary equipment to support the experiment includes the power supply and structural interface adapters. Structural interfaces are required for the controller/manipulators, operator restraints and the task board. Lack of convection cooling may dictate the need for fans to circulate air past the controller/manipulators and the electronic equipment.

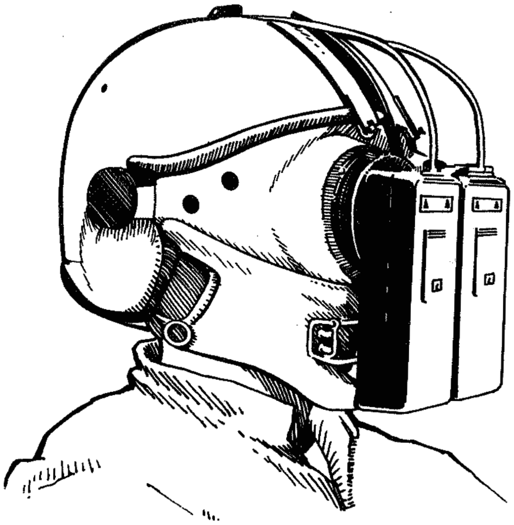


Figure 7.- Helmet mounted displays.

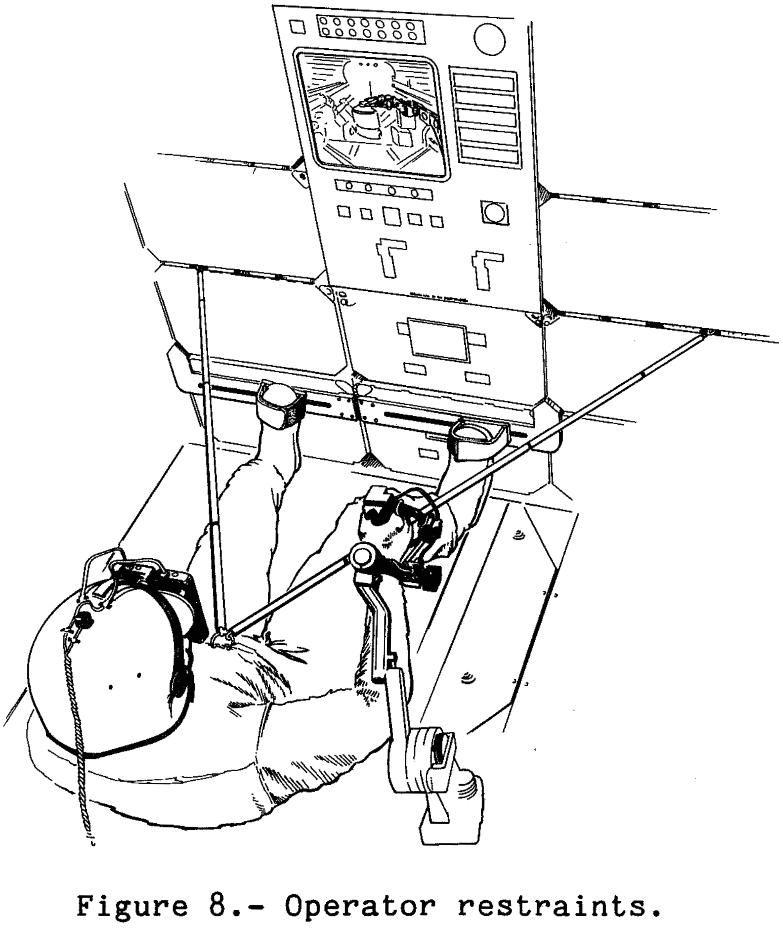


Figure 8.- Operator restraints.

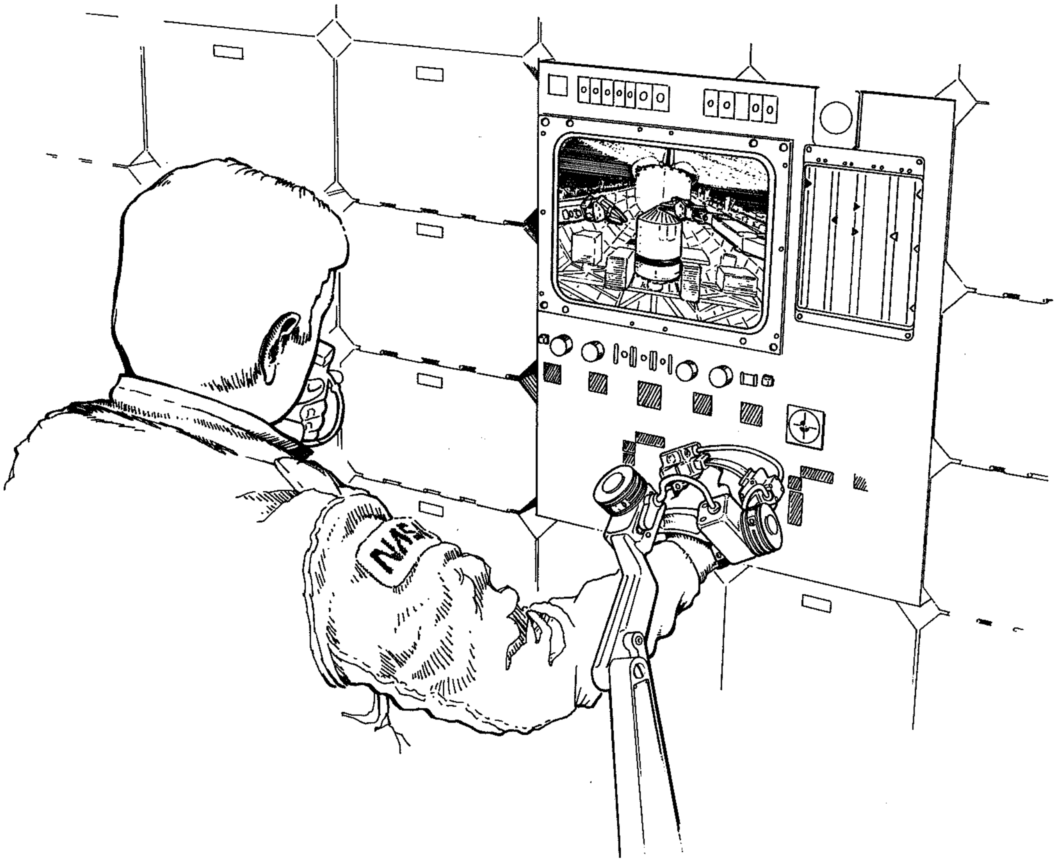


Figure 9.- Indirect viewing with CCTV.

## 5. Concept Analysis

The concept of a simple middeck test of dexterous manipulation in different telerobotic control modes has numerous advantages over a test setup in the cargo bay of the orbiter. The equipment, materials and function do not have to be certified for vacuum operation. The materials need only meet flammability and toxicity standards for use in the crew cabin. The small low-force manipulator arms do not represent a significant safety hazard for injury to the crew or damage to orbiter equipment. Tests, not under direct operator control may need safety isolation with light weight netting or simply avoidance of the test zone. The cabin location of the test setup allows hands-on access to alter test setups or to accommodate testing errors. This assures that useful data will be obtained. Repetitive testing with several operators can provide some degree of statistically significant results. Repeating tests using direct vision and television views of tasks gives an indication of the validity of similar testing in Earth-bound simulations.

A potential psychological disadvantage of the concept is the size of the manipulators. Most Earth-bound manipulators are massive to support payloads in the gravity field. Space manipulators can handle massive loads with low levels of force; therefore, they can be lighter than manipulators for Earth application. The perception of the small manipulator may detract from the impression of capability that is inherent in the test. The small manipulators will be mechanically different from the conventional design approach for manipulators. This will require careful design and test analysis to obtain duty-cycle test results that can be extrapolated to larger vacuum-rated manipulator designs.

## 6. Summary

Flight testing of telerobotic technology onboard the Space Shuttle can provide answers to uncertainties and issues that are a concern with the development of a telerobot system for space. The proposed middeck experiment provides a relatively low risk, low cost approach to early definition of telerobot system functions in space. The benefits of an operational space telerobotic system, such as the flight telerobotic servicer (FTS), in enhancing astronaut productivity and reducing the risk of extravehicular activity, deserve the greatest chance of success that can be achieved.

## REFERENCES

[1] R. deFigueiredo and L. Jenkins, "Space Robots," International Encyclopedia of Robotics, Dorf.

[2] L. Jenkins, "Telerobot experiment concepts in space," SPIE 1987 Symposium on Advances in Intelligent Robotic Systems, November 1, 1987.

# EXPERIMENTAL STUDY ON TWO-DIMENSIONAL FREE-FLYING ROBOT SATELLITE MODEL

Yoji UMETANI and Kazuya YOSHIDA

Department of Mechanical Engineering Science
Tokyo Institute of Technology
O-Okayama, Meguro, Tokyo 152, Japan

## Abstract

This paper treats the experimental study on a control method for a free-flying space robotic arm by means of a two-dimensional laboratory model. The authors' main target is to develop a new control method for trajectory tracking or target capturing, considering dynamical interaction between the manipulator arm and the base vehicle in space micro-gravity environment. In order to simulate the micro-gravity environment mechanically, the authors develop a laboratory model of a robot satellite supported on air bearings. The model comprises a base equipped with power and air supplies and a two-link manipulator arm. This model has relatively low gravitational and frictional disturbance in planar motion. An on-line RMRC control scheme with vision feedback is developed for exprimenting capture operations. This scheme utilizes the Generalized Jacobian Matrix which was proposed by the authors in a previous paper. In experiment, the acceleration environment of the model is evaluated firstly, then target capture operations are examined. The manipulator can properly chase and capture both a standing target and a moving target in spite of complex satellite/manipulator dynamical interactions. The experimental results confirm the validity of the Generalized Jacobian Matrix concept and the proposed control method.

## 1.Introduction

For a successful development of space projects, robotization and automation should be a key technology. Autonomous and dexterous robot systems could reduce the workload of astronauts and increase operational efficiency in many missions. One major characteristics of these space robotic systems, which clearly distinguishes them from on-earth operated ones, is the lack of a fixed base. Any motion of the manipulator arm will induce reaction forces and moments in the base, which disturb its position and attitude. If the arm were controlled for such task as target capturing without provision for this base disturbance, it would fail in the task. To cope with this problem, some approaches for modeling and control of space manipulators have been suggested.

Lindberg, Longmann and Zedd [1] proposed a method for simultaneous control of manipulator and satellite attitude. They derived a model of the dynamically interacting satellite/manipulator system to generate decoupled commands for manipulator joints and moment compensation devices.

On the other hand, modeling and control methods for free-flying systems have been developed which do not provide any attitude control for the satellite main body during manipulator operation. Vafa and Dubowsky [2] introduced the Virtual Manipulator concept in order to describe geometrically free-flying mechanical links, using thereby simmilar expressions as for ground-fixed ones. They applied this concept to analyze work spaces, as well as to solve the inverse kinematics.

Umetani and Yoshida [3,4] introduced the Generalized Jacobian Matrix concept for the expression of the free-flying behavior at kinematic level. Kinematics and inverse kinematics problems at velocity level can be treated in a simmilar way as for ground-fixed systems, by replacing the conventional Jacobian with the proposed new matrix. The concept was applied for resolved motion rate trajectory tracking control.

As an experimental study, Alexander and Cannon [5] recently developed a free-flying satellite robot simulator model with an arm controller, based on the computed-torque method, and obtained good results.

This paper treats the experimental investigation on the control of space free-flying manipulator systems. A laboratory model was developed, which is based on the same design concept as in Alexanders' work. The control scheme of this model utilizes the Generalized Jacobian Matrix. Target capture operations can be successfully demonstrated.

## 2.Free-Flying Robot Satellite Model

### 2.1 How to Simulate Micro-Gravity Environment

How to simulate micro-gravity environment in on-ground laboratory: that's always a serious problem for ground test experiments of space assemblies. In general, the following 4 methods could be available for this purpose.

(1) Experiment in an airplane flying along a parabolic trajectory or a free-falling capsule. In this case, we can observe pure mechanical behavior under the law of nature, but it costs a lot and is inconvenient.

(2) Experiment in a water pool with the support of buoyancy. This is especially good for training of astronauts' activities.

(3) Experiment on an air-cushion or air-bearings. In this case, however, the motion is restricted on a plane.

(4) Calculate the motion which should be realized in micro-gravity environment by using a mathematical model, then force a mechanical model to move according to the calculation. This method is adopted for the FTS test bed [6].

Among them, (1)-(3) are mechanical methods and, (4) is called as a hybrid simulation method combining mechanical models and mathematical ones. Each method has advantages and disadvantages and, we should carefully select the method so as to satisfy the purpose of the experiment.

In this paper, the authors adopt method (3), because they would like to observe the behavior of mechanical link systems under the law of nature by the simplest apparatus. Simulators utilizing air bearings have been developed in U.S.A. The most famous one is the test bed of SRMS [7] . It is designed to test the practical validity of the arm controller but not to investigate the free-flying behavior. So, the arm is fixed on the ground at the shoulder joint. A simulator model for the investigation of the free-flying behavior has been developed by Alexander and Cannon in Stanford Univ. [5] . The present laboratory model is designed with the same concept as Alexanders' model.

## 2.2 Design Concept of the Laboratory Model

The authors' goal of the experiment is to analyze the behavior of a mechanical link system in micro-gravity environment and to verify the proposed control scheme. In order to accomplish such an experiment, the laboratory model should be completely free-flying, with no mechanical disturbances for planar motion. To realize it, the model is significantly required

(a) to install the air supply for air bearings, and

(b) to be controlled autonomously or remotely.

And, as additional requirements, the model is desirable to be not to large, as light as possible, and easy to manufacture.

A conceptual structure of the model which satisfies these requirements is shown in Fig.1. A robot satellite model that has 2 jointed-link manipulator, is supported by 3 air pads. Each joint is actively rotated by a DC motor but there is no actuator for attitude control of the satellite main body. As a light and compact air supply, liquidized-gas bombs are installed on the satellite main body. A remote measurement and control system consists of satellite mounted subsystems; a communication port and a PD servo-controller and ground facilities; a CCD camera hanging from the ceiling, a Video Tracker (VT) and a personal computer (PC). The control loop is described briefly as follows: Tip, joints and tail of the model and a target object are marked with light emitting diodes. The motion of the model and the target is monitored by the CCD camera. Video signals of LED marks are transformed into position data by the VT and transmitted to the PC via a GPIB communication line. Each control command for both actuators is calculated in the PC and transmitted to the satellite via a wire-less communication system. Manipulator joints are locally controlled by the on-board PD servo-controller according to the tele-commands.

Due to the air supply by gas bombs and the remote measurement and control system, the laboratory model avoids air-tubes or wire connections from the earth, and is able to float on thin air films without any mechanical disturbances or external accelerations.

## 2.3 Specifications

Detail specifications of the laboratory model are listed as follows:

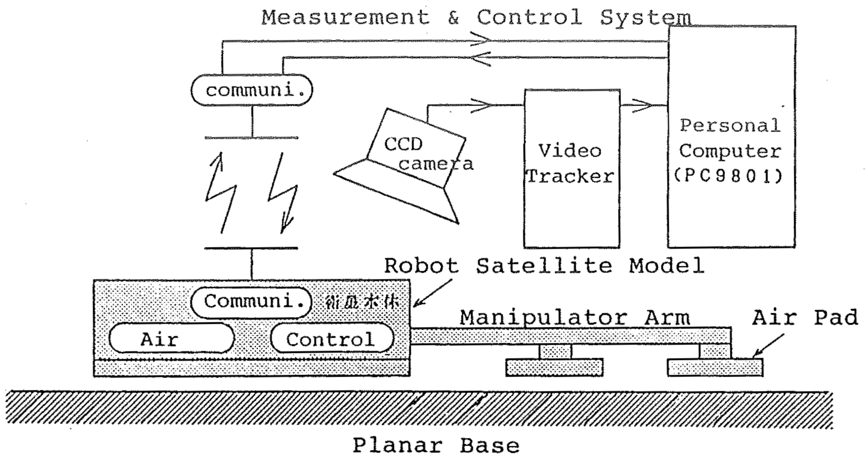


Fig.1. A conceptual structure of the laboratory model.

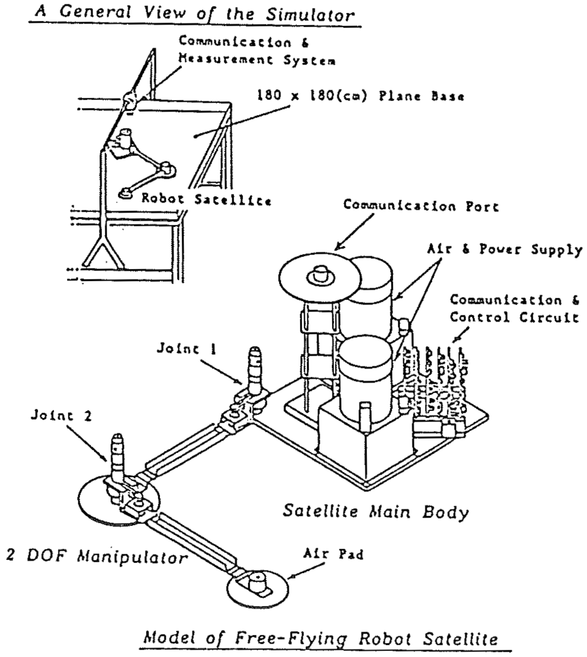
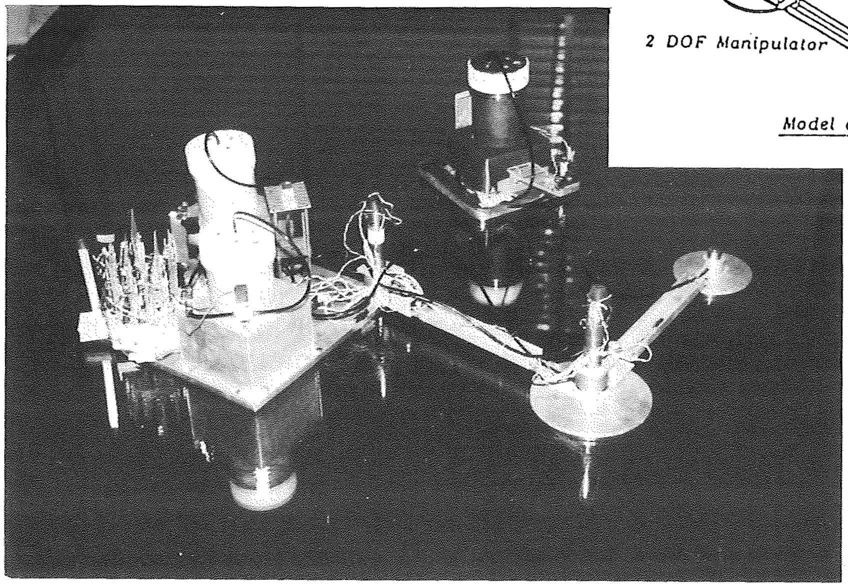

A General View of the Simulator
Communication & Measurement System
180 x 180(cm) Plane Base
Robot Satellite
Communication Port
Air & Power Supply
Communication & Control Circuit
Joint 1
Joint 2
Satellite Main Body
2 DOF Manipulator
Air Pad
Model of Free-Flying Robot Satellite

Photo 1 (↓) and Fig.2 (→).

A general view of the Robot Satellite model.



Robot Satellite

Satellite main body
- -dimension: 300x300mm
- -weight: 6.3 kg
- -equipments: gas-bomb type air supply
  rechargable battelies
  wire-less communication system
  local servo-controller

Manipulator
- -dimension: 700mm (350mm for each link)
- -weight: 1.4 kg
- -actuator: DC motor+planetary gear train
- -sensors: potentio-meters
  tacho-generators

Ground Facilities

Planar base table
- -dimension: 1800x1800mm
- -material: planar glass with multi-supports

Measurement and control system
- -512x492 CCD camera (NEC)
- -Video Tracker (G-3100:OKK Inc.)
- -16bit personal computer (PC-9801-VM2:NEC)

Photo 1 and Fig.2 show a general view of the developed laboratory model. Detail dimensions and inertia parameters of each link are listed in Table 1.

| body No. | 0 | 1 | 2 |
|---|---|---|---|
| a (m) | 0.190 | 0.162 | 0.124 |
| b (m) | 0.190 | 0.188 | 0.226 |
| m (kg) | 6.256 | 0.747 | 0.620 |
| I ($kgm^2$) | 0.169 | 0.0424 | 0.0141 |

Table 1. Dimensions and inertia parameters of each link.

## 3.Modeling and Control

An on-line resolved motion rate control (RMRC) scheme with vision feedback is developed for target capture operation by using the Generalized Jacobian Matrix (GJM). In this section, firstly the derivation of a mathematical model and the GJM is described, then an on-line control scheme is introduced.

### 3.1 Generalized Jacobian Matrix

A mathematical model and notations which correspond to the laboratory model are described in Fig.3. Bodies are numbered consecutively with "0" being the satellite main body and "2" the manipulator end-link. Tip position vector p and mass center of each link $r_i$ are described with respect to the origin of the inertial coordinate system. Only planar motion in X-Y plane and rotation around Z axis are considered in this paper.

For such a free-flying system, the momentum conservation equations hold true:

$$\sum_{i=0}^{2} m_i \dot{r}_i = \text{const.} \tag{1}$$

standing for translational momentum, and

$$\sum_{i=0}^{2} (I_i\omega_i+m_i r_i\times\dot{r}_i) = \text{const.} \qquad (2)$$

for rotational momentum, respectively. In these equations, $m_i$ is the mass of body i, $\omega_i$-its angular velocity and $I_i$ -its inertia matrix. Eq.(2) can be rewritten as

$$I_s\dot{\Omega} + I_m\dot{\Phi} = 0 \qquad (3)$$

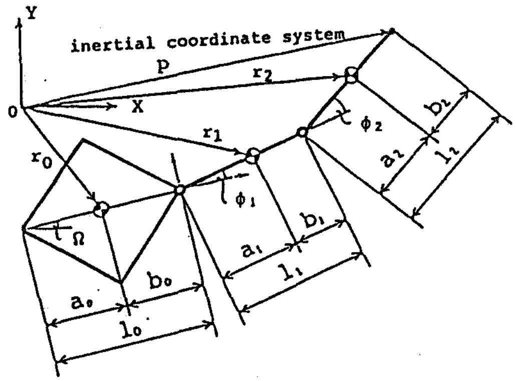


Fig.3. A mathematical model.

to distinguish the satellite rotational momentum $I_s\dot{\Omega}$ from the manipulator one $I_m\dot{\Phi}$, where $\dot{\Omega}$ is the attitude angular velocity and $\dot{\Phi}$ is the 2 x 1 joint velocity vector and $I_s$ , $I_m$ are defined in the Appendix. Initially, the system is assumed to be at rest, i.e. the right term of eq.(3) representing the initial momentum of the system is assumed to be zero.

On the other hand, for any manipulator, the relationship between the endtip velocity vector $\dot{p}$ and the joint velocity vector $\dot{\Phi}$ can be represented by the well known linear equation

$$\dot{p} = J\dot{\Phi} \qquad (4)$$

J being a Jacobian matrix. In the case of a satellite mounted manipulator, eq.(4) can be rewritten as

$$\dot{p} = J_s\dot{\Omega} + J_m\dot{\Phi} \qquad (5)$$

to distinguish the satellite-motion dependent endtip velocity $J_s\dot{\Omega}$ from the manipulator-motion dependent one $J_m\dot{\Phi}$. The satellite Jacobian $J_s$ and the manipulator Jacobian $J_m$ are defined in the Appendix. Note that both are functions of mass distribution in the satellite/manipulator system. From eqs. (3) and (5), we can eliminate the uncontrolled $\Omega$ variables:

$$\dot{p} = J^*\dot{\Phi} \quad ; \quad J^* = J_m - J_s I_s^{-1} I_m . \qquad (6)$$

$J^*$ is named the Generalized Jacobian Matrix for satellite mounted manipulator arm.

An important result with the GJM is that the conventional control scheme for ground-fixed manipulators is directly applicable for space free-flying ones by replacing J with $J^*$ . For example, a Resolved Motion Control scheme for free-flying manipulators is simply described with the inverse of $J^*$ as

$$\dot{\Phi}_d = [J^*(\Phi)]^{-1}v \qquad (7)$$

where v is the commanded endtip velocity and $\dot{\Phi}_d$ is resolved joint velocity command. This scheme could properly control trajectory tracking or target capture operations, taking into account the satellite/manipulator dynamic interactions.

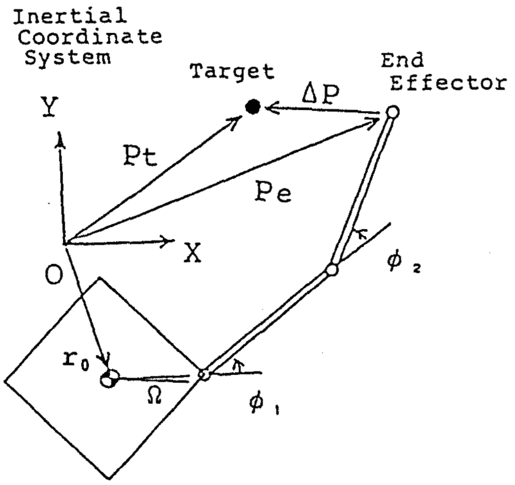


Fig.4. Parameter description for target capture operation.

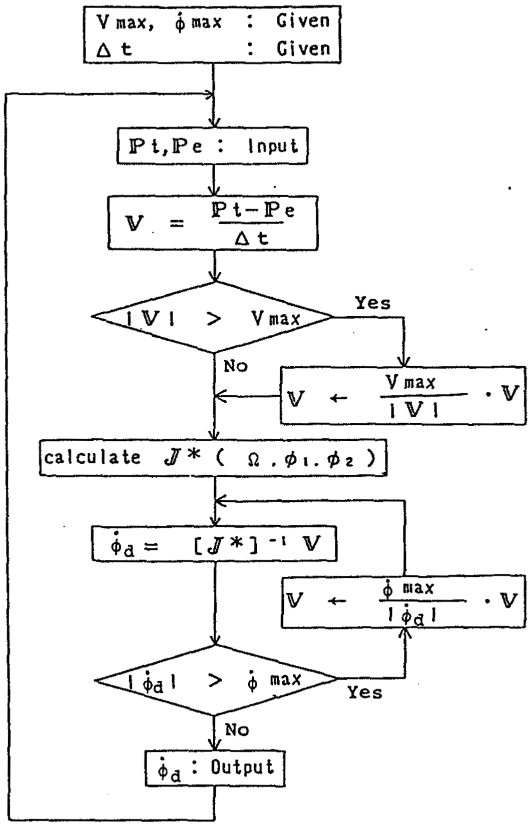

V max, $\dot{\phi}$ max : Given
Δ t : Given
Pt, Pe : Input
V = (Pt−Pe)/Δt
\|V\| > V max
Yes
No
V ← (V max / \|V\|) · V
calculate $J^*$ ( Ω . $\phi_1$ . $\phi_2$ )
$\dot{\phi}_d$ = [$J^*$]$^{-1}$ V
V ← ($\dot{\phi}$ max / \|$\dot{\phi}_d$\|) · V
\|$\dot{\phi}_d$\| > $\dot{\phi}$ max
Yes
No
$\dot{\phi}_d$ : Output

Fig.5.
Control flow-chart.

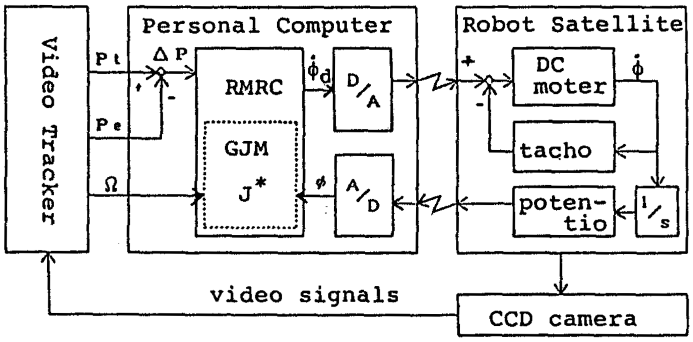


Fig.6. Control block diagram.

3.2 On-line RMRC Scheme based on the Generalized Jacobian Matrix

Parameters needed for capture control are defined in Fig.4. The following values are assumed to be measured at each sampling interval Δt during the operation: Ω -satellite attitude, $\phi_1, \phi_2$ -manipulator joint angles, $p_e$-position of manipulator endtip, and $p_t$ -position of the target. Endtip velocity command v is determined with a position error vector $\Delta p = p_e - p_t$ by

$$v = \frac{\Delta p}{\Delta t} = \frac{p_e - p_t}{\Delta t} . \qquad (8)$$

This value is then substituted in eq.(7), to obtain joint operation commands $\dot{\Phi}_d$ .

A flow-chart of this scheme with an appropriate data scaling process for avoiding high joint velocities near singular points, is shown in Fig.5. Also a block diagram is presented in Fig.6. The values of $p_e$ , $p_t$ and Ω are measured by the Video Tracker with video signals. Joint angles Φ are measured by potentiometers located at manipulator joints and transmitted via the tele-communication system. The generalized Jacobian Matrix is calculated with Ω and Φ , then the joint velocity commands are determined by the above mentioned procedure. The commands are transmitted to the robot satellite and joints are controlled with a local velocity feedback. Note that the designed control scheme is based on the vision data from the ground-fixed CCD camera, however, it is equivalent to measure p by a satellite mounted camera and by an on-board sensor. It means that this control scheme can be easily installed on the satellite.

Given conditions for the following experiment are:

| | | |
|---|---|---|
| maximum tip velocity | $v_{max}$ | : 0.1 m/sec, |
| maximum joint velocity | $\dot{\phi}_{max}$ | : 15.0 deg/sec, |
| data sampling interval | $\Delta t$ | : 0.2 sec. |

In this case, position sensing by the VT requires 1/30 seconds and more than 0.1 seconds are spent for the calculation of the GJM and its inversion. The calculation is executed by i8086+8087 processors using C language.

## 4.Experimental Results

### 4.1 Arm Slewing Maneuver

As a preliminary experiment, the measurement of friction between the planar base and air pads and an arm slewing maneuver have been made. The friction of air films is due to the viscosity of the air and is unavoidable. The order of the friction coefficient $\mu = \alpha$ (horizontal acceleration)/ g (vertical acceleration) is measured as $10^{-3}$. In other words, the laboratory model is allowed to work in $10^{-3}$g acceleration environment.

Fig.7 shows the experimental result of an arm slewing maneuver. In Fig.7 (a) the mass center of the system, which should be stationary during the maneuver, is a little bit moving, and in Fig.7 (b), the measured satellite attitude (in solid line) has also a small error from the calculated one by eq.(3) (in dot line). However, if the friction effect is considered, the momentum conservation will be proved, and the relationship between joint velocities and tip velocity described by eq.(6) will be also true during the maneuver. This fact shows the validity of the Generalized Jacobian Matrix concept.

### 4.2 Capture of a Standing Target

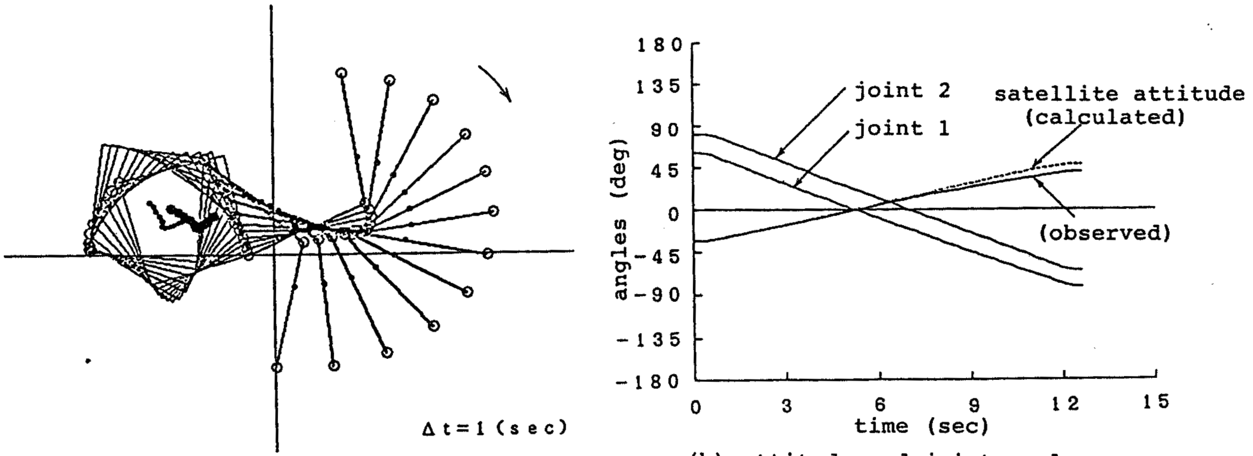
Capture operations of a standing target are successfully accomplished by the simple rate control with the GJM. Fig.8 shows a typical result of the operation. From the initial point to the target, the manipulator endtip travels straight and smoothly in spite of a large satellite attitude change.

### 4.3 Capture of a Moving Target

As for a smooth chase and capture of a moving target, the endtip velocity command is modified by the information of the target velocity $\dot{p}_t$.

$$v = \frac{p_e - p_t}{\Delta t} + \dot{p}_t \qquad (9)$$

With this small modification, the manipulator works very well for capture operations both of a standing target and a moving target. Fig.9 shows a typical experimental result of the capture of the moving target with $\dot{p}_t$= 0.05m/sec.



(a) motion of the system

(b) attitude and joint angles

Fig.7. Arm slewing maneuver.

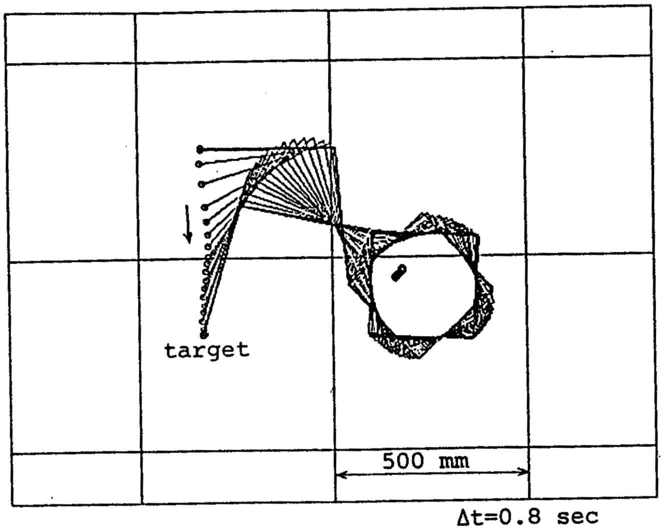


Fig.8.
Capture of a
standing target.

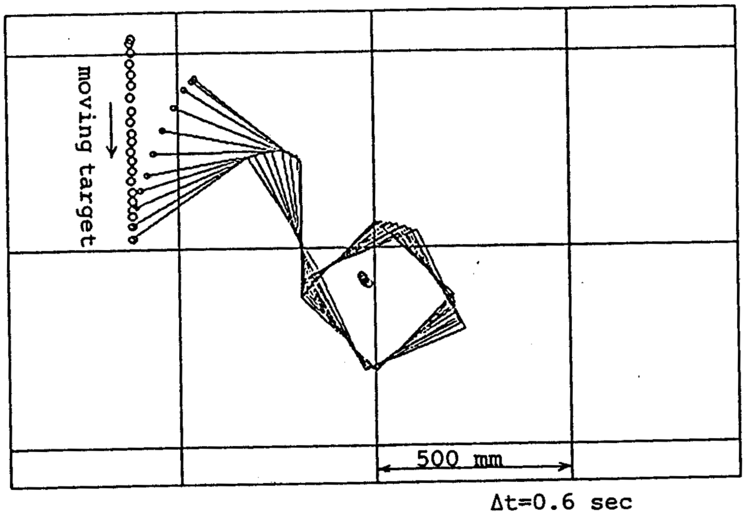


Fig.9.
Capture of a
moving target.

## 5.Conclusions

This paper presents the experimental investigation on the control of a space free-flying manipulator system. In order to simulate the micro-gravity environment, a laboratory model of a robot satellite supported on air bearings, is developed. The model is evaluated to have relatively low gravitational and frictional disturbance (of order $10^{-3}g$) for planar motion. An on-line RMRC scheme with vision feedback is developed for experimenting capture operations. This scheme is based on the Generalized Jacobian Matrix concept. Through experiments, it has been shown that the manipulator is able properly to chase and capture both a standing target and a moving target, in spite of the complex satellite/manipulator dynamical interaction. The results confirm the validity of the Generalized Jacobian Matrix concept and the proposed control method. Advantages of the approach are the relatively simple algorithm and the possibility for easy installation on practical systems.

## Appendix

The matrices $I_s$, $I_m$, $J_s$ and $J_m$ are defined as follows:

$$I_s=h_0+h_1+h_2+2C_{01}+2C_{12}+2C_{20},$$

$$I_m=[h_1+h_2+C_{01}+2C_{12}+C_{20},\ h_2+C_{12}+C_{20}]$$

where

$$h_0=I_0+M_0b_0^2. \qquad h_1=I_1+M_0a_1^2+M_2b_1^2+2M_1a_1b_1,\quad h_2=I_2+M_2a_2^2,$$

$$C_{01}=(M_0b_0a_1+M_1b_0b_1)\cos\phi_1,\quad C_{12}=(M_1a_1a_2+M_2b_1a_2)\cos\phi_2,\quad C_{20}=M_1b_0a_2\cos\phi_1+\phi_2,$$

$$M_0=m_0(m_1+m_2)/w,\quad M_1=m_0m_2/w,\quad M_2=(m_0+m_1)m_2/w,\quad w=m_0+m_1+m_2.$$

$$J_s=[-(s1+s2+s3)\ \ c1+c2+c3\ ]^t;$$

$$J_m=\begin{bmatrix} -(s2+s3) & -s3 \\ c2+c3 & c3 \end{bmatrix},$$

where

$$s1=m_0b_0\sin\Omega/w, \qquad c1=m_0b_0\cos\Omega/w$$

$$s2=(m_0l_1+m_1b_1)\sin(\Omega+\phi_1)/w, \qquad c2=(m_0l_1+m_1b_1)\cos(\Omega+\phi_1)/w,$$

$$s3=[(m_0+m_1)l_2+m_2b_2]\sin(\Omega+\phi_1+\phi_2)/w,\quad c3=[(m_0+m_1)l_2+m_2b_2]\cos(\Omega+\phi_1+\phi_2)/w.$$

## References

[1] R.W.Longman, R.E.Lindberg, M.F.Zedd, "Satellite-Mounted Robot Manipulators - New Kinematics and Reaction Moment Compensation", The Int. J. of Robotics Research, vol.6, No.3, pp.87-103. (1987)

[2] Z.Vafa, S.Dubowsky, "On the Dynamics of Manipulators in Space Using the Virtural Manipulator Approach", IEEE Conf. on Robotics & Automation, pp.579-585. (1987)

[3] Y.Umetani, K.Yoshida, "Continuous Path Control of Space Manipulators Mounted on OMV", Acta Astronautica, vol.15, No.12, pp.981-986. (1987)

[4] Y.Umetani, K.Yoshida, "Resolved Motion Rate Control of Space Manipulators with Generalized Jacobian Matrix", to be published in IEEE Int. J. of Robotics & Automation.

[5] H.L.Alexander, R.H.Cannon, "Experiments on the Control of a Satellite Manipulator", Proc. of Material Handling Reseach Focus, Georgia Institute of Technology. (1986)

[6] J.G.Watzin et al, "The Flight Telerobotic Servicer; a Focus for Automation and Robotics on the Space Station", AAS-88-043. (1988)

[7] C.G.Wagner-Bartak et al, "Shuttle Remote Manipulator System Hardware Test Facility", NASA-CP-2150, pp.79-93. (1980)

# The Astronaut and the Banana Peel: an EVA Retriever Scenario

Daniel G. Shapiro
Advanced Decision Systems

## Abstract

No matter the care and caution employed in Space Station activities, accidents will happen. To prepare for this problem, NASA is constructing a robot, the EVA Retriever (or EVAR), whose purpose is to retrieve astronauts and tools that float free of the Space Station. Advanced Decision Systems is at the beginning of a two-year project to develop research software capable of guiding EVAR through the retrieval process. This involves addressing problems in machine vision, dexterous manipulation, real time construction of programs via speech input, and reactive execution of plans *despite the mishaps and unexpected conditions* that arise in uncontrolled domains.

This paper concerns the problem analysis phase of our work. We use a walk through of an EVAR scenario to elucidate major domain and technical problems, and we conclude with an overview of our technical approach to prototyping an EVAR system.

## 1. The EVAR Task

In the event of an accident on the Space Station, NASA has decided to send a robot rescue vehicle in lieu of risking yet another astronaut in untethered, remote operations. The extra vehicular activity retriever (EVAR) is required to locate, rendezvous and return with an astronaut within 120 minutes, which is the duration of normal oxygen reserves. Calculations using an empirically observed maximum separation rate of 3.5 ft/sec and the known acceleration capability of the MMU show that the astronaut will at most be 5 km away.

Retrieval will be an interactive process, with the potential for teleoperated control of EVAR (moving the arms during grappling), autonomous behavior (search, navigation and spin cancelling), and user instruction ("extend your right arm while opening your hand"). As the scenario already assumes there has been an accident in space, there is little control over EVAR's operating environment. The robot will require a great deal of flexibility. Since the situation is also life threatening, EVAR must be capable of operating even if some of its systems are nonfunctional, and if external instrumentation has failed.

The basic scenario for retrieval of a free floating object has 7 parts:

1. Activation
2. Acquisition
3. Rendezvous
4. Grappling
5. Return
6. Transfer
7. Deactivation

In the nominal scenario, these parts occur in sequence, although there are many patterns for flow of control (for example, rendezvous and grappling can repeat if, say, a toolbox spills).

We will discuss acquisition, rendezvous and grappling in some detail. A more complete discussion of the

EVAR scenario can be found in [4].

## 2. EVAR Hardware and System Capabilities

EVAR is a man sized manipulator platform seated in the Manned Maneuvering Unit, which provides mobility. The current design is anthropomorphic, with 2 arms and a non-specified hand/tool attachment. It carries a laser range finder (with 128 x 128 resolution). Potential sensors include:

- A monocular TV camera (on the wrist and/or fixed to the chassis),
- a radar,
- a proximity sensor with spherical coverage,
- a star tracker (low probability),
- a Global Positioning System receiver (good to ~10 meters),
- gyros for detecting own acceleration and for computing velocity and position,
- a radio receiver.

EVAR will be capable of accepting voice commands.

The current processor configuration contains 4 transputers (currently 15 Mhz, T414 chips, to be augmented with five, 20Mhz, T800 chips). The current programming language is Occam, which will be changed to C. The total weight of EVAR (including the MMU) is 600-1000lbs. Note that fuel is highly constrained; total delta-v = 66 fps at system weight less than 600 pounds, or 1558 lb-sec usable impulse. Acceleration is 0.3 +/- 0.06 fps$^2$ for translation, and 10.0 +/- 4.0 deg/sec$^2$ rotation.

The manipulator system will have both force feedback and proprioception (the ability to directly sense the angle of arms and joints).

## 3. The Space Station Environment

The Space Station has many of the aspects of a construction zone, whose physical composition will evolve through time. With respect to navigation, this means the immediate environment is cluttered, although three dimensional maps for completed portions of the structure will be available. At larger distances, there are no obstacles to movement, but it becomes apparent that orbits are accelerating reference frames. This affects trajectory calculations. From the point of view of computer vision, the environment is quite restricted; lighting is stark (numbers of point sources, no diffuse lighting), and the objects are metallic, geometric, and well modeled. However, it may be day or night, and the earth or the sun may necessarily be in the image. It is possible to instrument the Space Station, astronauts and equipment to a reasonable degree for the purposes of retrieval.

## 4. Acquisition

The goal of *acquisition* is to positively identify the location of the escaped object; specifically, its distance, speed, and direction of travel. This task subdivides into *detection* (identifying the object's direction in space from EVAR), and *measurement* (identifying its velocity vector). Acquisition may be accomplished through a mixture of EVAR and non-EVAR resources. For example; user input, Space Station radar (if present), the EVAR camera, or laser range finder. Acquisition may also produce partial results; for example, a second astronaut may be able to identify the object's direction, but not its separation velocity.

There are several basic strategies for performing acquisition:

1. Scan for the lost object in place.

2. Move to an observation location and scan for the lost object.
3. Physically conduct a search pattern if scanning is not sufficient.
4. Move in an externally provided direction and scan for the object while moving.

EVAR will need to select among these options, with possible user interaction.

It is clear that instrumentation for both *detection* and *measurement* exists, but the actual selection becomes an issue when space station safety, mass, and environmental constraints are taken into account. In particular, current design does *not* call for radar (owing to its mass and interference with EM sensitive instruments), while backup instrumentation is also desired.

The most obvious options for instrumentation are as follows (D indicates ability to perform detection, M is for measurement):

Passive sensing:

- optical, by EVAR or tethered astronaut **D**
- triangulation of astronaut-carried beacon **D,M**
- interferometry, based on a beacon **D,M**
- motion detection against a star background **D**, with doppler from a beacon **D,M**

Note that passive detection will not work on all objects (small dark tools are generally hard to see, although they can be found with a plausible motion detector). Some form of prearranged beacon is required to support measurement with passive sensors.

There are two obvious active sensing techniques:

- radar (**D** + **M**)?
- laser range finder **D**

It is not clear that a single radar can perform both detection and measurement within Space Station power, time, and mass constraints. In the absence of radar, a motion detector with EVAR's laser range finder can perform acquisition, or we can employ a space station based strobe for optical acquisition (which has poor positional accuracy), with triangulation on the returned signal. Note that these backups may be less capable or less tolerant of environmental conditions than an appropriate radar.

## 5. Rendezvous

The *rendezvous* process involves trajectory calculation and path execution (in 3D), and culminates with a standoff maneuver (position holding action) next to the object to be rescued. The rendezvous phase may also involve completion of the detection and measurement tasks initiated during acquisition. EVAR must obtain a positive visual fix on the object during rendezvous.

Clohessy-Wiltshire equations provide the appropriate trajectory calculations for objects in orbit, unless the parameters of motion are incompletely known. Time efficient, fuel efficient, and "maximum probability of acquisition" trajectory calculations are all relevant tools. A technique is required for measuring and maintaining a constant distance from an irregular, rotating object.

The path execution function involves travel in the potentially cluttered environment near the space station, with avoidance of moving obstacles.

# 6. Grappling

*Grappling* begins when EVAR is close to the object, and stationary with respect to its center of mass. It ends with EVAR physically coupled to the object such that it can be towed back to the Space Station. The problems of grappling come from the fact that the object may be tumbling in space; its size, shape, the complexity of its movement, and its rotational energy all affect the grappling techniques and tools that can be safely applied. Once the physical connection is made, conservation of angular momentum dictates that EVAR will inherit some unknown spin (in 3 axes) about the EVAR-object center of mass. This spin should be cancelled, and the object tethered to EVAR before it returns to the space station.

The need to grapple with a tumbling object raises a few interesting questions. Can EVAR exploit properties of motion in free space to simplify the mechanics of interaction? What kinds of grappling tools can be/should be employed? What role does the astronaut play in the grappling process? Is he/she the object, or the director?

## 6.1. Properties of motion in free space

Unfortunately, in all but the simplest situations, there is no easy way to simplify the dynamics of interaction with a free-floating object. Exceptions are objects which don't rotate, or which spin perfectly about one axis, as in the case of a normal communications satellite. In precession, motion is still about one internal axis, but that axis moves in space. As a result, EVAR cannot adopt the strategy of matching spin along some fixed axis, and leisurely grapple with the object, achieving an apparent 0 velocity encounter.

In more detail, every object has 3 natural, or *principal* axes, one with maximum, one with minimum, and one with an intermediate value of rotational inertia. In space, a rigid object with no applied torques can have constant spin about the maximum or minimum axis. Spin about the intermediate axis is metastable (recall the high school experiment with a spinning tennis racket - there are certain spins it will not maintain).

The motion of non-ideal objects (astronauts) also evolves. A pure rotation in the presence of even minor perturbations becomes a tumble (motion about more than one axis). A tumble in the presence of dissipative forces becomes a pure rotation, but only after a substantial period of time; days, not minutes as would be required to aid retrieval.

The motion of the instantaneous axis of rotation of a tumbling object *is* constrained about its angular momentum vector (although not in an easily observable way). This suggests EVAR can simplify (to what degree?) the object's apparent motion by diagnosing and then aligning itself with the object's angular momentum vector. Intuitively, this will also simplify the net interaction; if EVAR and the object have parallel angular momentum vectors, grappling will exert no net torque, and the EVAR-object system will have the same orientation after all spin is cancelled.

## 6.2. Direct Coupling

In direct, or manual grappling, EVAR simply uses its manipulators to establish an immediate physical connection with the object. This exposes EVAR to impact, and will result in sudden momentum exchange, so the technique makes most sense for objects with small amounts of rotational energy.

The task of directly grasping a tumbling object is by no means trivial, and in many ways beyond the current state of the art. For EVAR to use its hands, it will need to model the object's shape (a scene analysis problem with dynamics), select a location to grab, diagnose its tumble (mathematically plausible for rigid objects, but sensor intensive), plan a path to the appropriate rendezvous (in time and space), and execute the grasp with some form of force feedback control. If it is a rotating wrench, only EVAR's limb needs to move. If it is a slowly rotating astronaut, the grasping action may involve EVAR translation, rotation, and use of both arms (and many joints) in a coordinated, and compliant two handed grasp. Some object recognition capability is also implicated; it better to grasp an astronaut by a leg than by the head, and some equipment (such as radio antennae) are too fragile to use as handholds.

There are several ways to simplify this task. One is to ignore object rotation by grasping quickly. This will be difficult for both teleoperation and autonomous control, especially when it involves complex hand or limb motions. A reasonable solution is to instill manipulator reflexes to make a set of jaws close quickly on physical contact or on signal.

The second solution is to employ grappling tools which can adapt to various combinations of object mass, movement, and shape. For example, if EVAR employed a simple butterfly net, it could entirely avoid the recognition, modelling, and dexterous manipulation tasks discussed above.

### 6.8. Loose Coupling

The objectives of loose coupling are to spread the EVAR-object momentum exchange over time, and to avoid dangerous interactions between EVAR and moving objects. This requires specially designed grappling tools. The ideal tool is insensitive to all of the object parameters (shape, identity and motion) discussed in the previous pages.

A representative list of tools is shown below. (This list is a compendium of suggestions spanning ideas from silly to clever, and buildable to completely impractical.) Several are illustrated in figure 6-1.

1. A rope with a lifesaver, deployed for the astronaut to grab.
2. A rod and reel apparatus, with a rotating end attachment to allow motion.
3. A powerful adhesive on the end of a stiff wire probe.
4. A contact activated clamp attached to the end of a manipulator or wire (as in 2 above).
5. A butterfly net.
6. A pellet gun carried by the astronaut, used to reduce spin.
7. An electrostatic aligner, with sprays for inducing a dipole field on the object.
8. A bolo, used to snare the target. In theory, the weights at the ends of the bolo spread out, causing the rotational inertia of the object to increase and its rotational velocity to decrease, simplifying further grappling.
9. A compressed gas apparatus, deployed onto the astronaut, mechanically designed to align the thrust opposite object rotation.
10. A modified TOW missile, guided remotely from the space station (an EVAR backup).
11. Rotating nets embedded in a pair of clappers.
12. The *Brupiro Grappler*: an annular ring containing movable, rotating snares, manipulated by an external, swivelling handle. This device has 3 degrees of freedom, and permits capture with no immediate momentum exchange.
13. A spherical shell with extensible arms that matches the tumble of the object at the instant of capture.
14. A momentum leech. The leech extends telescoping arms with attached masses until the angular velocity of system is damped. It then drops off and retracts its arms (spinning up in the process), while EVAR grapples with the now slowly moving astronaut.

Many of these tools have significant mechanical problems. The lifesaver is practical and inexpensive, but the astronaut could grab EVAR just as easily. With ideas 2 and 3, a tumbling object will attempt to roll up the line and tangle itself in the process; this produces astronauts packaged in large balls of string. The pellet gun (6) has a nasty side effect; it sends a barrage of high velocity material into the environment.

There is also a small problem with choosing a direction to fire, since tumbles are notoriously disorienting. A bolo might slow the astronaut's spin, until it reverts to its historical role as a weapon. No one has admitted to a design for the electrostatic aligner. The compressed gas jet (9) is mechanically complex, and will also apply an asymmetric torque which will produce odd motion (probably a spiral translation).

The remaining ideas are actually quite serious, although they require more elaborate machinery. The clapper is under design at NASA. The Brupiro grappler is the invention of this author (and Dr. Bruce Sawhill). The tumble matching shell is under devlopment by Capt. Don Idle at the University of Texas at Austin. Before dismissing the leech, note that an astronaut with 50 kg-m$^2$ of rotational inertia can be reduced to 1/10th his initial angular velocity with five, 1kg weights at the end of 10 meter poles.

### 6.4. User Interaction During Grappling

If we assume that the lost astronaut is not terribly disoriented, he/she will almost certainly insist on being active throughout EVA retrieval. This suggests some form of a vocabulary for verbally directing EVAR.

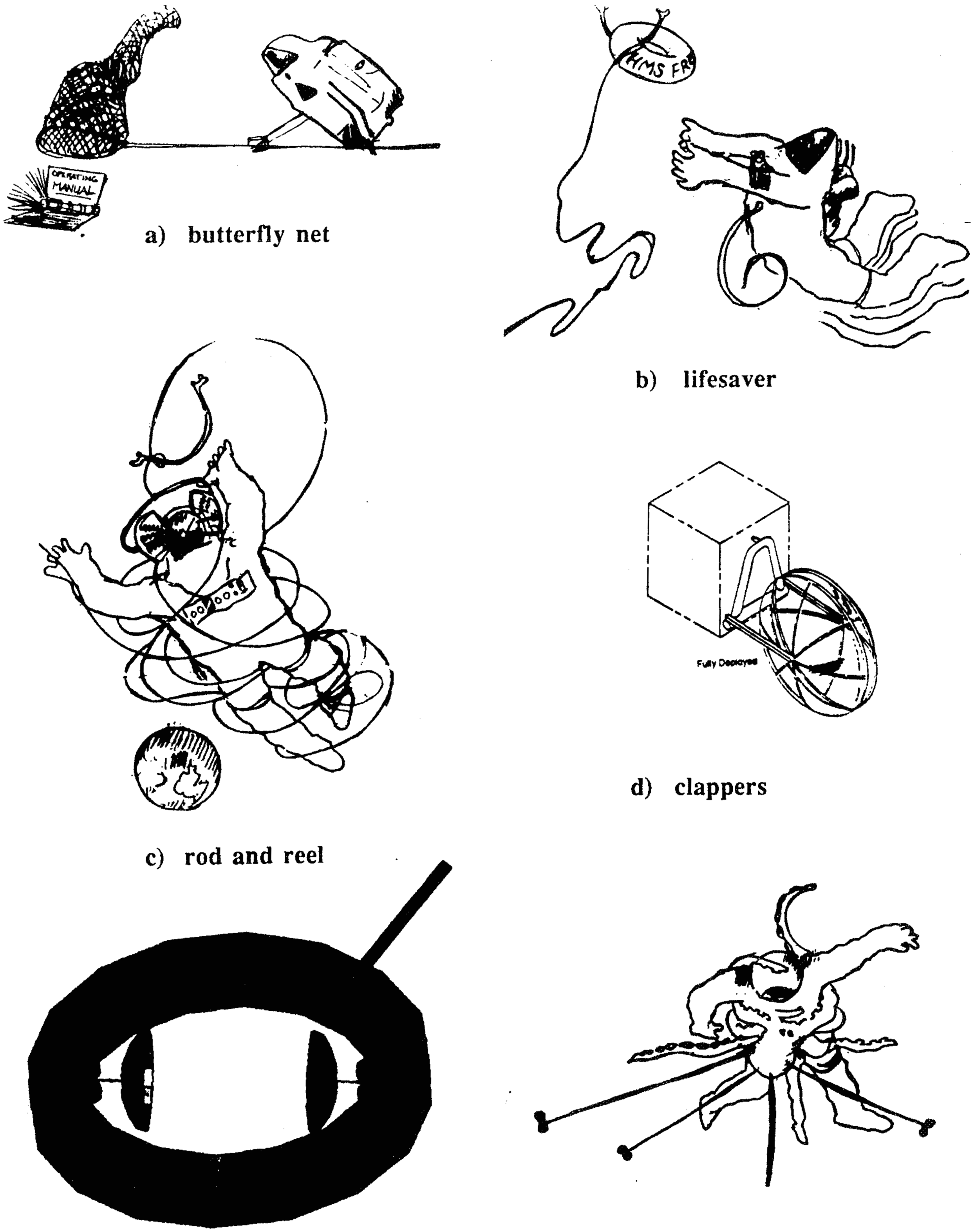
A simple <action> <object> <direction> <magnitude> grammar is an obvious (though primitive) start; define nouns for pieces of robot anatomy (wrist, elbow, shoulder, arm, hand, body), verbs for actions (move, open, close, flex, rotate), and keywords for direction, magnitude and speed (forward, backward, clockwise, 20 degrees, fast, slow). Use these terms to form robot commands; "move body forward slowly", "extend left arm", "flex right elbow 30 degrees", "rotate wrist clockwise", "close hand".

We might also introduce keywords for temporal coordination and sensory conditions such as "while", "then", "on contact", etc. This would allow expressions such as, "extend both arms while opening both hands", which is useful as a method of delivering an object, or, "on contact close hands quickly", which is a method for grasping a rotating object.

This vocabulary essentially defines a programming language for controlling simple robot actions. On the input side, the voice recognition and natural language understanding tasks appear almost ideal candidates for automation; the vocabulary is limited and sentence meanings correspond to robot actions. The program generation task may be more difficult, depending upon the underlying robot action primitives and the complexity of the control structure the user desires. Michalowski [2] has addressed this problem in the design of robot arms and wheelchairs for assisting the severely handicapped. He has produced programmable, operational robots, but without reactive control.

Controlling detailed manipulations by voice command is likely to be quite tedious, and too inefficient for use in retrieval scenarios. A solution is to raise the level of discourse, allowing the astronaut to interact with EVAR in terms of a vocabulary of physical behaviors. For example, the astronaut might say, "use the back-in procedure", or "use the clapper procedure", which causes the robot to begin execution of a predefined multistep plan. The behaviors function as contexts which provide specific command options to the astronaut in addition to causing robot action. So, for example, the "back-in" procedure directs EVAR to rotate 180 degrees, apply gentle thrust towards the astronaut, and terminate when it senses impact. Commands for controlling EVAR speed, and for initiating spin cancelling become available in this context. (Behaviors can be used as the basis for segmenting EVAR's sensing, processing, and resource requirements. See section 7.)

Figure 6-1: A set of grappling tools



a) butterfly net

b) lifesaver

c) rod and reel

d) clappers

e) Brupiro grappler

f) momentum leech

### 6.5. Return

Planning for the return trip involves acquisition and trajectory calculations as before, but in this case it is reasonable to assume EVAR knows its velocity and position relative to the space station.

The act of moving towards the space station with the object in tow is a different proposition. In this situation, *the EVAR/object total mass and center of mass are both unknown*, and can be significantly different from EVAR values alone. This suggests an adaptive control solution, where the control parameter is units of thrust, and the measure is units of deviation from the desired (or expected) direction of travel. (Measurement of thrust, calibrated in Newtons, is not required.)

If exact numeric solutions are desired (for example, to know how much fuel is going to be available or if the task can *in theory* be solved) it may be possible to run a simple experiment with EVAR, using MMU gyros to measure acceleration against a known application of force. The proper instrumentation for this task is currently not present on the MMU; it lacks calibrated accelerometers for measuring translation and rotation rates, and a force sensor for measuring the moment actually being applied.

## 7. Scenario analysis

This scenario brings out a number of technical problems in machine vision, dexterous manipulation, man-machine interaction and robot programming.

Concerning machine vision, the environment is both well known and restricted. However, the image understanding problems are non-trivial. A representative problem is; extract an astronaut from a scene with the sun in the background (using video or laser range data), and autonomously identify his leg, vs. his head. Track this object, and develop a predictive model for its location. Note that the astronaut will be tumbling, and moving his limbs. This problem clearly requires research and application of the technologies of optic flow, model based vision, and shape detection.

A second obvious problem concerns navigation in 3 dimensions with avoidance of moving obstacles. No such vision/planning systems exist to date; the closest examples are in autonomous land vehicles (although land motion is arguably harder).

The manual grappling task brings out issues in dexterous manipulation. Here, there is a need for rapid planning and execution of motion (to grasp a tumbling wrench while a predictive motion model is in effect), with a potential need for compliant response and coordinated action (use of two hands, vs. one).

Concerning man-machine interaction, the discussion of grappling with an astronaut brought out a need for real time construction of small programs via speech input, or for an extended EVAR command vocabulary.

Finally, the uncontrolled nature of the scenario as a whole indicates a need for a different approach to robot programming; EVAR cannot view retrieval as execution of a preset plan. To illustrate this point, consider a few things which can "go wrong" during retrieval:

1. The object may be an astronaut, tangled in a girder, rotating at high speed. As a whole, that system will be outside the tolerance of the available grappling tools, even though the astronaut alone is within EVAR abilities.
2. Both EVA astronauts will be lost, EVAR will bring one to the other, and end up functioning as an assistant during grappling instead of the primary active agent.
3. EVAR will encounter mechanical problems. Not all sensing systems will be operational. Communications will temporarily fail. Tools will malfunction (for example, the momentum

leech won't disengage, or the clappers will not deploy completely). Beacons will not work.

4. Initial grappling will fail, and make the situation worse. (EVAR will experience a collision and inherit significant momentum.)
5. EVAR will lose orientation during spin cancelling, and will have to reacquire the Space Station. Something unknown will go wrong and it will have to be manually directed (teleoperated) back to the Space Station.
6. There will be critical time pressures during grappling.
7. The object will obscure EVAR sensors after grappling, or, it will get tangled in the tethering process.
8. The pod bay doors will not open.

The salient feature of these examples is that unexpected complications will stretch preparations for each of acquisition, rendezvous, grappling, return and transfer. Substantial improvisation may be required. This argues for a view of EVAR as toolbox that provides a range of applicable behaviors, as opposed to a single purpose, preprogrammed device.

## 8. Technical Approach

Advanced Decision Systems is at the beginning of a two-year project to develop research software capable of guiding EVAR through the retrieval process. In order to build a functional system in that time frame (or in any reasonable future period), we believe that the technical problems discussed above cannot be tackled head on. Our approach relies on the following key ideas;

- We simplify vision processing by use of user assisted scene interpretation (after Lawton [1]).
- We avoid manipulation tasks through use of grappling tools.
- We support user interaction by programming EVAR in terms of a vocabulary of physical behaviors.
- We provide reactive response by embedding these behaviors in an architecture which examines all possible actions each time step.

The goal in user assisted scene interpretation is to map recognition and modelling problems into tracking. (For example, the user identifies the astronaut's leg in the image, selects a generalized cylinder from the model library and provides an initial fit to the rotating object. The system takes over the tracking task from this point.)

The last two points above address a major issue of current robotic control; the need to have plans but also react to external events as they occur. A system architecture for reactive plan execution is shown in figure 8-1.

The main feature of this architecture is the abstraction called the *current program*, which is a constantly modified data structure that maintains all the behaviors, tasks, and situation triggers on mind of EVAR at any given time.

Computationally, each behavior is a strong context for specifying action. Examples are to "maintain an object fix", to "maintain distance", "maintain orientation", or at a larger scale, a "rendezvous mode" which can be hierarchically decomposed into simpler contexts. Each behavior requires limited world knowledge, supports specialized sensor processing routines, and has triggers which respond to events natural in that context (e.g., impact during grappling). For the purposes of arbitration, each behavior can also identify the resources it requires.

The use of strong contexts allows a unique approach to plan representation; after Schoppers [3], each behavior encodes *all* paths from the current situation to the goal. This provides a great deal of reactivity.

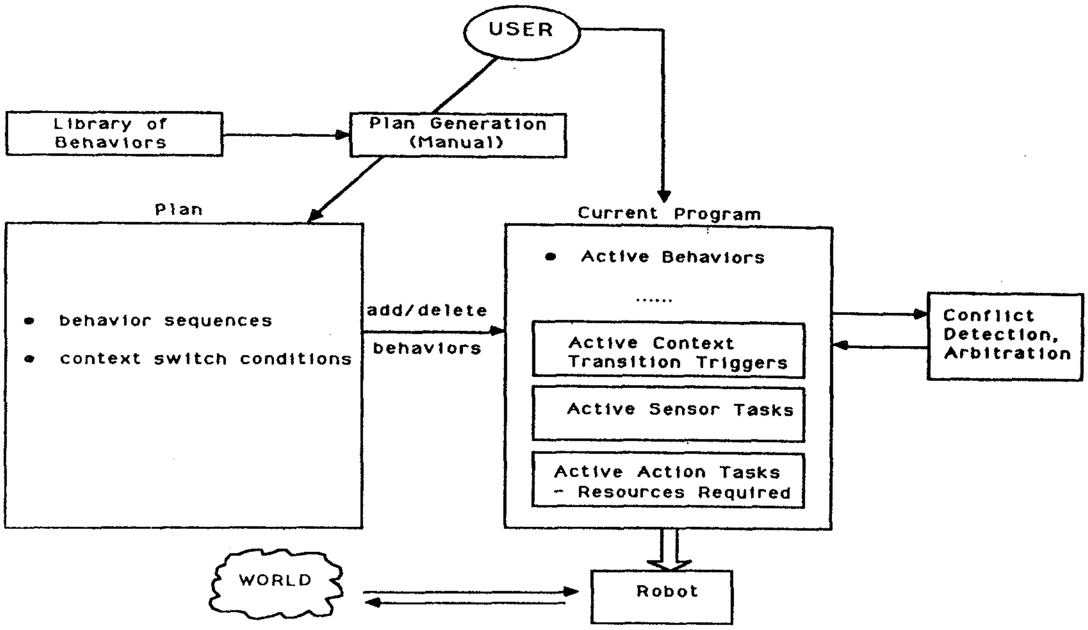


**Figure 8-1:** An Architecture for Reactive Plan Execution

In particular, the robot is not dependent on previous actions working as desired; the step relevant to the current situation is always applied. Longer plans, such as the high level EVAR retrieval sequence, are expressed as sequences of contexts together with their transition criteria (e.g., rendezvous transitions to grappling when the standoff maneuver is in process).

The basic decision loop of this architecture is as follows:

1. determine if any context switch criteria have been met, if so, activate and deactivate the appropriate behaviors,
2. process all behaviors in the current program one cycle (this involves both diagnosis of the current situation and selection of the appropriate action)

In this view, a plan defines appropriate decision contexts, and is treated as a set of suggestions, to be taken as the immediate situation allows.

# References

1. Lawton, Daryl. Vision System for Human and Robot Teams. Advanced Decision Systems, 1987.

2. Michalowski, S.J. Progress Towards a Robotic Aid for the Severely Disabled. Proceeding of the Sixth CISM-IFToMN Symposium on Theory and Practice of Robots and Manipulators, 1986.

3. Schoppers, M.J. Universal plans for unpredictable environments. Proc 10th IJCAI, 1987.

4. Shapiro, D.G. Architectures for Semi-Autonomous Planning. Phase I Final Report TR 3197-01, Advanced Decision Systems, 1988.

# Computed Torque Control of a Free-Flying Cooperating-Arm Robot

Ross Koningstein Marc Ullman
Robert H. Cannon Jr.
Stanford University Aerospace Robotics Laboratory*

January 30, 1989

## Abstract

**This paper presents a unified approach to solving free-floating space robot manipulator endpoint control problems using a control formulation based on an extension of computed torque. Once the desired endpoint accelerations have been specified, the kinematic equations are used with momentum conservation equations to solve for the joint accelerations in any of the robot's possible configurations: fixed base or free-flying with open/closed chain grasp. The joint accelerations can then be used to calculate the arm control torques and internal forces using a recursive order n algorithm. Initial experimental verification of these techniques has been performed using our laboratory model of a two-armed space robot. This fully autonomous spacecraft system experiences the drag-free, zero-g characteristics of space in two dimensions through the use of an air cushion support system. Results of these initial experiments are included which validate the correctness of the proposed methodology. The final section addresses the further problem of control in the large where not only the manipulator tip positions but the entire system consisting of base and arms must be controlled. The availablity of a physical testbed has brought many benefits to this work—particularly a keener insight into the subtleties of the problem at hand.**

## 1 Introduction

To achieve fast, precise control of a physical system, accurate dynamical modelling is required. Dynamical modelling quickly becomes complex and cumbersome for human derivation as controlled systems become more and more complex. This section will formalize the process of computed torque control specification for robotic manipulator dynamical systems, introducing terms easily generated by algorithmic means and suitable for computer implementation. The control technique will also present extensions and formalisms for dealing with free-flying and closed chain rigid body manipulator systems, all of which share the characteristic of being easily machine generated. The basic premise for this technique is derived from the computed torque control technique.[1]. This technique uses kinematics for determining joint acceleration inverse dynamics for obtaining the corresponding joint torques. Specification of desired controls in operational or cartesian space[2] requires that the inverse and derivative of the system's Jacobian **J** be used. The Jacobian is expressed by

$$\mathbf{v}^{\text{endpoint}} = \mathbf{J}\dot{q}$$

where **v** is a vector of the speeds of the manipulator endpoints, measured in some coordinate system and $\dot{q}$ are the derivatives of the joint angles. Research by Alexander[3] into the control of free-flying robots first showed that the Jacobian was non-square. Subsequently, Umetani and Yoshida[4] demonstrated that the system Jacobian could be augmented by momentum equations to enable solving for joint accelerations. Independent investigation has led to the formalization of the structure of the Jacobian Matrix, using Kane's [5] notational convention, and augmenting a system's Jacobian to include both momentum relations and kinematic constraints implied by closed chains. The procedure presented here for Jacobian generation makes it possible to solve for actuator joint torques without determining reduced order equations of motion. Instead, it is possible to solve for these torques directly with a simple recursive order n procedure.

*Work performed under NASA contract NCC-2-333

### 1.1 Concepts used in Analysis

This theory for serial chain manipulators is derived using Kane's dynamical analysis techniques. The analysis that follows assumes that the velocities **v** of points and angular velocities $\omega$ of bodies in the system under consideration can be expressed in a Newtonian reference frame as follows:

$$\begin{aligned} \mathbf{v}^i &= \sum_{s=1}^{p} \mathbf{v}_s^i u_s \\ \omega^i &= \sum_{s=1}^{p} \omega_s^i u_s \end{aligned}$$

where the generalized speeds $u_{1..n}$ are linear combinations of the derivatives of the generalized coordinates $\dot{q}_{1..n}$. The partial angular velocities of bodies, and partial velocities of points, as defined by Kane[5], can be shown to be:

$$\begin{aligned} \mathbf{v}_r &= \frac{\partial}{\partial u_r}\mathbf{v} \\ \omega_r &= \frac{\partial}{\partial u_r}\omega \end{aligned}$$

## 2 Jacobian Structure

### 2.1 Desired Accelerations

First, a method will be demonstrated which formulates the system Jacobian using partial velocities. The desired endpoint accelerations will then be expressed using these partial velocities and their derivatives, which is the basis for the computed torque method. The Jacobian, expressed using generalized speeds[1] , is used as follows:

$$\mathbf{v}^{\text{endpoint}} = \mathbf{J}u$$

The endpoint acceleration can then be expressed as:

$$\mathbf{a}^{\text{endpoint}} = \mathbf{J}\dot{u} + \dot{\mathbf{J}}u$$

and the joint accelerations can be solved for by rearranging these equations:

$$\dot{u} = \mathbf{J}^{-1}(\mathbf{a}^{\text{endpoint}} - \dot{\mathbf{J}}\ u)$$

The Jacobian matrix's components are dependent upon the partial velocities and partial angular velocities of the endpoint of the manipulator(s) in the system. An endpoint velocity can be expressed in terms of its partials as:

$$\mathbf{v}^{\text{endpoint}} = \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} u_r$$

and therefore 3D endpoint velocity can be expressed in terms of speeds along some established inertial x,y and z directions, for example, along unit vectors which we define as **x**, **y** and **z**:

$$\begin{aligned} \mathbf{v}^{\text{endpoint}} \cdot \hat{\mathbf{x}} &= \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{x}}\ u_r \\ \mathbf{v}^{\text{endpoint}} \cdot \hat{\mathbf{y}} &= \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{y}}\ u_r \\ \mathbf{v}^{\text{endpoint}} \cdot \hat{\mathbf{z}} &= \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{z}}\ u_r \end{aligned}$$

the elements of the Jacobian due to an endpoint's velocity is therefore:

$$\begin{aligned} J_{1r} &= \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{x}} \\ J_{2r} &= \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{y}} \\ J_{3r} &= \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{z}} \\ &\vdots \end{aligned}$$

As shown above, desired endpoint accelerations can be expressed in terms of the Jacobian, its derivative, and the generalized speeds and their derivatives. The derivatives of the elements of the Jacobian can also be determined from the partial velocities:

$$\begin{aligned} \dot{J}_{1r} &= \dot{\mathbf{v}}_r^{\text{endpoint}} \cdot \hat{\mathbf{x}} \\ \dot{J}_{2r} &= \dot{\mathbf{v}}_r^{\text{endpoint}} \cdot \hat{\mathbf{y}} \\ \dot{J}_{3r} &= \dot{\mathbf{v}}_r^{\text{endpoint}} \cdot \hat{\mathbf{z}} \\ &\vdots \end{aligned}$$

where the derivatives, taken in a Newtonian reference frame, of the partial velocities are

$$\dot{\mathbf{v}}_r^{\text{endpoint}} \triangleq \frac{\mathrm{d}^{\mathrm{N}}}{\mathrm{dt}}\mathbf{v}_r^{\text{endpoint}}$$

which can be calculated very easily given the angular velocity of the body that the partial velocity vectors

[1] If one chooses $u \triangleq \dot{q}$ then this is the standard Jacobian. If not, it becomes a more generalized Jacobian. The theory is valid for either case.

are based in. This completes the formal description of the Jacobian elements for desired accelerations. Note that desired angular accelerations are treated in an identical manner, allowing body angular acceleration specification.

## 2.2 Momentum Conservation

In any free-flying system of bodies, the linear and angular momenta vary according to the external forces on the system. On a free-flying robot, these are the system thrusters. If assume that these thruster settings are known a priori, we are able to predict the rate of change of the system momenta. The Jacobian can be augmented with linear and angular momenta equations to include these system states in the calculation of the desired generalized accelerations. Inclusion of these relations can make a Jacobian full rank, and suitable for application of the computed torque method.

First, the linear momentum, then the angular momentum of the system will be examined. The linear momentum $\mathbf{L}^i$ of a body $i$ in the system is

$$\begin{aligned}\mathbf{L}^i &= m^i\mathbf{v}^{i*}\\ &= m^i\sum_{s=1}^{n}\mathbf{v}_s^{i*}u_s\\ &= \sum_{s=1}^{n}\mathbf{L}_s^i u_s\end{aligned}$$

where the *partial linear momentum* of body $i$ is defined by

$$\mathbf{L}_s^i \triangleq m^i\mathbf{v}_s^{i*}$$

The linear momentum $\mathbf{L}$ of a system of $\nu$ bodies is the sum of the linear momenta of each body $i$ in the system:

$$\begin{aligned}\mathbf{L} &= \sum_{i=1}^{\nu}\mathbf{L}^i\\ &= \sum_{i=1}^{\nu}m^i\mathbf{v}^{i*}\\ &= \sum_{i=1}^{\nu}m^i\sum_{s=1}^{n}\mathbf{v}_s^{i*}u_s\\ &= \sum_{i=1}^{\nu}\sum_{s=1}^{n}m^i\mathbf{v}_s^{i*}u_s\\ &= \sum_{i=1}^{\nu}\sum_{s=1}^{n}\mathbf{L}_s^i u_s\\ &= \sum_{s=1}^{n}\mathbf{L}_s u_s\end{aligned}$$

where the *partial linear momentum* of the system of $\nu$ bodies is defined by

$$\mathbf{L}_s \triangleq \sum_{i=1}^{\nu}m^i\mathbf{v}_s^{i*}$$

The *partial linear momenta* of the system can be formulated using the mass and center of mass partial velocity of each body in the system. The process of building an augmented Jacobian using these vector quantities is similar to the process used for the partial velocities discussed in the previous section, and will be discussed after the angular momentum terms are examined.

The angular momentum $\mathbf{H}^i$ of each body $i$, about its center of mass is:

$$\begin{aligned}\mathbf{H}^i &= \mathbf{I}^{i/i*}\omega^i\\ &= \mathbf{I}^{i/i*}\sum_{s=1}^{n}\omega_s^i u_s\\ &= \sum_{s=1}^{n}\mathbf{I}^{i/i*}\omega_s^i u_s\\ &= \sum_{s=1}^{n}\mathbf{H}_s^i u_s\end{aligned}$$

where the *partial angular momentum* $\mathbf{H}_s^i$ of each body is defined as

$$\mathbf{H}_s^i \triangleq \mathbf{I}^{i/i*}\omega_s^i$$

The central angular momentum $\mathbf{H}$ of the system of $\nu$ bodies about the system's center of mass point, is:

$$\begin{aligned}\mathbf{H} &= \sum_{i=1}^{\nu}\mathbf{H}^i + \sum_{i=1}^{\nu}(\mathbf{r}^{i*}-\mathbf{r}^{cm})\times m^i\mathbf{v}^{i*}\\ &= \sum_{i=1}^{\nu}(\mathbf{I}^{i/i*}\omega^i + (\mathbf{r}^{i*}-\mathbf{r}^{cm})\times m^i\mathbf{v}^{i*})\\ &= \sum_{i=1}^{\nu}(\sum_{s=1}^{n}\mathbf{I}^{i/i*}\omega_s^i u_s + \sum_{s=1}^{n}(\mathbf{r}^{i*}-\mathbf{r}^{cm})\times m^i\mathbf{v}_s^{i*}u_s)\\ &= \sum_{i=1}^{\nu}\sum_{s=1}^{n}(\mathbf{H}_s^i u_s + (\mathbf{r}^{i*}-\mathbf{r}^{cm})\times\mathbf{L}_s^i u_s)\\ &= \sum_{s=1}^{n}\mathbf{H}_s u_s\end{aligned}$$

where the *partial angular momentum* $\mathbf{H}_s$ of the system is defined as

$$\mathbf{H}_s \triangleq \sum_{i=1}^{\nu}(\mathbf{H}^i_s + (\mathbf{r}^{i*} - \mathbf{r}^{cm}) \times \mathbf{L}^i_s)$$

A set of Jacobian augmentation equations can be set up which describe the relation between the momenta and the generalized speeds.

$$\begin{aligned} \mathbf{L} &= \mathbf{J}^L u \\ \mathbf{H} &= \mathbf{J}^H u \end{aligned}$$

The elements of the Jacobian due to the linear and angular momenta are therefore:

$$\begin{aligned} J^L_{1r} &= \mathbf{L}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

and

$$\begin{aligned} J^H_{1r} &= \mathbf{H}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

The *partial momenta* can be formulated automatically using the partial velocities in the system.

Expected momentum rates (due to external forces and torques) can be expressed in terms of these Jacobian augmentation equations and their derivatives along with the generalized speeds and their derivatives.

$$\begin{aligned} \dot{\mathbf{L}} &= \mathbf{J}^L \dot{u} + \dot{\mathbf{J}}^L u \\ \dot{\mathbf{H}} &= \mathbf{J}^H \dot{u} + \dot{\mathbf{J}}^H u \end{aligned}$$

The derivatives of the elements of the augmented Jacobian can be determined from the partial momenta:

$$\begin{aligned} \dot{J}^L_{1r} &= \dot{\mathbf{L}}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

and

$$\begin{aligned} \dot{J}^H_{1r} &= \dot{\mathbf{H}}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

where the derivatives, taken in a Newtonian reference frame, of the partial momenta are:

$$\begin{aligned} \dot{\mathbf{L}}_r &\triangleq \frac{\mathrm{d}^{\mathrm{N}}}{\mathrm{dt}}\mathbf{L}_r \\ \dot{\mathbf{H}}_r &\triangleq \frac{\mathrm{d}^{\mathrm{N}}}{\mathrm{dt}}\mathbf{H}_r \end{aligned}$$

and the rate of change of the momenta are given by:

$$\begin{aligned} \dot{\mathbf{L}} &= \sum \mathbf{F}^{\mathrm{ext}} \\ \dot{\mathbf{H}} &= \sum \mathbf{T}^{\mathrm{ext}} + \sum (\mathbf{r}^{ext} - \mathbf{r}^*) \times \mathbf{F}^{\mathrm{ext}} \end{aligned}$$

This completes the formal description of the Jacobian elements for momentum conservation.

## 2.3 Closed Chains

In a dynamical system with nonholonomic constraints, the generalized speeds $u_{1..n}$ are not independent, rather, one (or more) are dependent on the rest. In the system considered, a manipulator system, this condition can arise when two ends of a chain touch and are held together, either by a pin joint, or rigidly. The case of a velocity constraint on the a manipulator, a nonholonomic constraint situation, will be analyzed, and the constraint equations will be expressed in terms of quantities used in the kinematics derivations.

The constraint of endpoint closure is described by:

$$\mathbf{v}^{\mathrm{endpoint}_1} = \mathbf{v}^{\mathrm{endpoint}_2}$$

expanding this into partial velocities,

$$\sum_{r=1}^{n} \mathbf{v}_r^{\mathrm{endpoint}_1} u_r = \sum_{r=1}^{n} \mathbf{v}_r^{\mathrm{endpoint}_2} u_r$$

defining a constraint velocity[2]:

$$\begin{aligned} \mathbf{C} &\triangleq \mathbf{v}^{\mathrm{endpoint}_1} - \mathbf{v}^{\mathrm{endpoint}_2} \\ &= 0 \end{aligned}$$

and the *constraint partial velocities*[3] evaluate to:

$$\mathbf{C}_r = \mathbf{v}_r^{\mathrm{endpoint}_1} - \mathbf{v}_r^{\mathrm{endpoint}_2}$$

[2] The concept of a constraint velocity is not dependent upon having a free-floating base and hence works for all instances of closed kinematic chains

[3] Although the constraint velocity is zero, the individual *constraint partial velocities* are non-zero.

It is evident that by dot multiplication with inertial basis vectors, as was done with endpoint velocity, this vector equation can be reduced to scalar equations for incorporation into the system Jacobian.

$$0 = \mathbf{J}^C u$$

where the elements of these Jacobian augmentation equations are:

$$\begin{aligned} J^C_{1r} &= \mathbf{C}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

These constraint partial velocities can be formulated automatically using the partial velocities of the endpoints of the manipulator which are touching.

Differentiating the constraint augmentation equations automatically expresses the acceleration constraints:

$$0 = \mathbf{J}^C \dot{u} + \dot{\mathbf{J}}^C u$$

The derivatives of the constraint augmentation equations can also be determined from the partial velocity derivatives:

$$\begin{aligned} \dot{J}1r &= \dot{\mathbf{C}}_r \cdot \mathbf{x} \\ &\vdots \end{aligned}$$

where the derivatives, taken in a Newtonian reference frame, of the constraint partial velocities are

$$\begin{aligned} \dot{\mathbf{C}}_r &\triangleq \frac{\mathrm{d}}{\mathrm{dt}} \mathbf{C}_r \\ &= \dot{\mathbf{v}}_r^{\mathrm{endpoint}_1} - \dot{\mathbf{v}}_r^{\mathrm{endpoint}_2} \end{aligned}$$

This completes the formal description of the Jacobian elements for closed chain constraints. Note that angular velocity constraints can be treated in an identical manner.

## 3 Joint Acceleration Solution

The full system Jacobian $\mathbf{J}^S$ can now be constructed using the following components: A regular Jacobian which relates the linear and angular velocities of the manipulator endpoint(s) to the the system's generalized speeds. Next augmentation equations describing the rates of change of system momenta are added. Finally, augmentation equations which ensure that the chain closure constraint is met are added. This process results in a full rank Jacobian that looks like:

$$\mathbf{J}^S = \begin{bmatrix} \mathbf{J} \\ \mathbf{J}^L \\ \mathbf{J}^H \\ \mathbf{J}^C \end{bmatrix}$$

A corresponding set of control objectives can be formulated:

$$\mathbf{a}^S = \begin{bmatrix} \mathbf{a}^{\mathrm{endpoint}} \\ \sum \mathbf{F}^{\mathrm{ext}} \\ \sum \mathbf{T}^{\mathrm{ext}} + \sum (\mathbf{r}^{ext} - \mathbf{r}^*) \times \mathbf{F}^{\mathrm{ext}} \\ 0 \end{bmatrix}$$

Relating these two quantities is the equation:

$$\mathbf{a}^S = \mathbf{J}^S u$$

from which we can solve for the derivatives of the generalized speeds corresponding to this set of control objectives:

$$\dot{u} = \mathbf{J}^{S^{-1}} (-\dot{\mathbf{J}}^S u + \mathbf{a}^S)$$

The resulting derivatives of the generalized speeds can then be used in an inverse dynamics routine to obtain corresponding joint control torques.

## 4 Order n Inverse Dynamics

In this section a simple and straightforward algorithm to solve the inverse dynamics equation for the control torques along a serial chain with revolute joints will be presented. Traditional computed torque control schemes have used the following equation to compute the joint torques:

$$\mathbf{M}\ddot{q} = \mathbf{V}(q, \dot{q}) + \mathbf{T}$$

This method requires $\mathcal{O}(n^2)$ computations, and requires that the mass matrix and non-linear terms of the system $\mathcal{S}$ be computed, then desired joint accelerations and known joint rates be used to generate a vector from which the control torques are easily derived. We will present an alternate method of computing these joint torques in $\mathcal{O}(n)$ computations. This method is based on the Newton-Euler method of formulating robot equations of motion, but instead of generating equations symbolically, we will generate numerical values for accelerations, joint forces and torques, and actuator torques as we traverse the robot's chain manipulator.

As we recurse down the rigid body chain, endpoint accelerations are used to determine the accelerations of all the joints and each of the center of mass points of the $\nu$ bodies in the system. We can use the link recursion relation that the acceleration at the start of a link is related to the acceleration at the end of a link as follows:

$$\begin{aligned}\mathbf{a}^{\text{start}} &= \mathbf{a}^{\text{end}} \\ &\quad -\boldsymbol{\alpha}^{\text{link}} \times \mathbf{r}^{\text{start to end}} \\ &\quad -\boldsymbol{\omega}^{\text{link}} \times \boldsymbol{\omega}^{\text{link}} \times \mathbf{r}^{\text{start to end}}\end{aligned}$$

where the following components are derived as follows:

$$\boldsymbol{\alpha}^{\text{link i}} = \boldsymbol{\alpha}^{\text{link i}-1} + \ddot{q}_i \cdot \lambda^{\mathbf{i}}$$

The axis direction $\lambda^{\mathbf{i}}$ is a positive rotation, in a right handed sense, along $q_i$. The forces and moments are propagated back from the end of each chain. We assume the force and moment at the end of the chain is a known value, typically zero. If the chain is closed, then a desired 'squeeze' force can be assumed as a starting internal force at the link end, and conceptually cutting the closed chain, into two.

We take moments about the joint at the start of the link, and consider only the components along the joint's axis $\lambda^{\mathbf{i}}$. The moments due to the center of mass acceleration and the link's angular acceleration are easily evaluated given its mass properties. The joint motor torque will be the only unknown in the equation

$$\begin{aligned}\mathbf{T}^{i} \cdot \lambda^{\mathbf{i}} &= -(\mathbf{T}^{\text{link end}} \\ &\quad +\mathbf{r}^{\text{start to end}} \times \mathbf{F}^{\text{end}} \\ &\quad -\mathbf{r}^{\text{start to }*} \times m^{i}\mathbf{a}^{\text{i}*}) \cdot \lambda^{\mathbf{i}}\end{aligned}$$

Now take moments about the link start point, which are the moments applied to the end of the next link in. Likewise, the sum of the forces will yield the forces applied by this link to the end of the next link in. The focus of reference can now be shifted to the next link in, where this process can be repeated until all of the control torques have been determined.

The process of solving for the joint control torques or forces is fairly straightforward, and if the robot has two or more arms, the solution for the control values for the various arms can be done in parallel.

# 5 Implementation

The Jacobian formulation method introduced here has been used to generate the joint acceleration specification matrix equation necessary in order to solve the computed torque control problem for the general 3D case of a free-flying robot with kinematic chain manipulators. The $\mathcal{O}(n)$ inverse dynamics solution has also been derived for this general 3D case. A specialized and partially optimized derivation for 2D has been done to allow testing on our experimental free-flying robot model.

The dynamical system under study, a dual arm satellite manipulator model, is essentially a serial chain of rigid bodies, and undergoes only minor changes (in terms of structure) when it grasps an object: it either becomes a longer chain, or it becomes a closed chain. If the equations of motion of a chain system have a certain form, then the addition of extra links to the system should result in a small change in the computation of the equations of motion - and not necessitate the rederivation of the system's equations of motion from scratch. The possibility of generating equations of motion and control algorithmically is desirable, since this task is then no longer a manual procedure. For our 2D robot testbed, the algorithms, given the joint accelerations, to compute the control torques are $\mathcal{O}(n)$.

Continuing work in the analysis of robot dynamics by Rosenthal[6], Rodriguez[7] and others have shown that robot dynamics for simulation can be solved in $\mathcal{O}(n)$ computations. In the spirit of this process, we have presented an algorithm for control which is also $\mathcal{O}(n)$.

# 6 Experimental Hardware

We have built a laboratory model of a dual-armed space robot which experiences in two-dimensions the drag-free, zero-g characteristics of space. These characteristics are achieved through the use of air cushion technology which allows our vehicle to float on a $9'x12'$ granite surface plate with a drag-to-weight ratio of about $10^{-4}$ and gravity induced accelerations below $10^{-5}g$—a very good approximation to the actual conditions of space. The robot is a fully self contained spacecraft containing

- an onboard gas subsystem (used both for flotation and for propulsion via thrusters)
- a complete electrical power system with plug-in rechargeable batteries packs[4] and power condi-

[4] The battery packs can also be recharged while on board the vehicle through the use of an umbilical power cord

tioning, distribution, and monitoring circuitry

- a full complement of sensors and signal conditioning electronics
- a high speed microprocessor based computer system with a floating point coprocessor
- a complete set of digital and analog data acquisition and control interfaces
- a fiber optic based data/communications link to a network of off-board computers

The robot measures $50cm$ in diameter and stands $65cm$ high with a total mass of just under $50kg$. In order to simplify maintenance operations as well as to facilitate future design modifications the robot was designed as a series of independent modules. These modules take the form of layers (see figure 1) which each perform a distinct task. The layers can be easily separated[5] when necessary for servicing or repair.

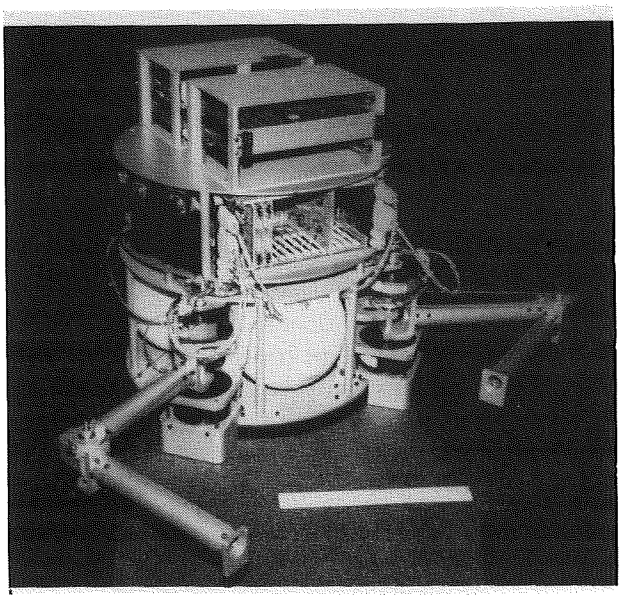


Figure 1: Stanford University Aerospace Robotics Laboratory Dual-Arm Space Robot

Figure 2 shows the nomenclature used for modeling the dynamics and characterizing the mass properties of the robot. The base body has three degrees of freedom $(x, y, \theta)$ and sports eight gas jet thrusters mounted as four ninety-degree pairs sitting at the corners of a square inscribing its outer circumference.

A pair of two-link planer arms aligned with a set of ninety-degree separated rays are attached to the base.

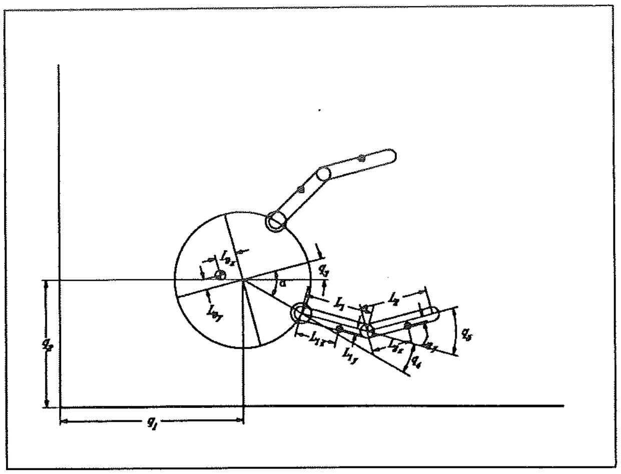


Figure 2: Free body diagram of space robot indicating nomenclature used for dynamic modelling.

These manipulator arms are driven by a coaxial set of limited angle DC torque motors—the shoulder joint being driven directly while the elbow joint is driven though a cable from the elbow motor which rides on the shoulder link. Both joints are instrumented with RVDTs for sensing joint angles. Analog differentiators provide corresponding rate signals in hardware. The manipulators are equipped with pneumatically actuated grippers which possess a single degree of freedom along the z-axis. Objects can be grasped by lowering the gripper plungers into cup-like grasp points mounted on the objects.

The onboard computer system runs the VxWorks[6] real time operating system. This operating system allows us to develop code on our Sun Workstations which can then be downloaded to the target processor via a fiber optic Ethernet link. Since the real time OS contains a complete implementation of TCP/IP and NFS our target processor can access files and data on our host server. We have configured our system with a set of subnets so that we can communicate between on and off board processors without incurring delays due to traffic on our workstation LAN.

We will ultimately be adding an on-board vision system in order to allow us to perform endpoint control. This addition will enabling us to capture and manipulate free-floating targets.

## 7 Experimental Results

We have implemented the control methodology described above on our space robot system and it works!

[5] The main layers can be separated without requiring the use of any tools.

[6] *VxWorks*$^{TM}$ is a product of Wind River Systems, Emeryville, CA

# 8 Large Motion Control

In order for free flying space robots to be effective instruments in the space automation arsenal they must be capable of autonomously executing large motions which involve the coordinated motion of their base body and their manipulators. It is for this reason that the task of gross motion control of a space robot poses a set of interesting and unique challenges which differ from the typical satellite positioning/attitude control problem or the uncontrolled free-floating base situation presented above. In most satellite control problems we are interested in achieving two principal goals: 1) keeping the satellite as a whole on its proper orbit path, and 2) keeping various sensors and/or communications devices pointed in desired directions. These objectives amount to requirements for holding the center of mass of the satellite on track while servoing the attitude of the main body so as to keep the pointing actuators within their allowable ranges of motion. As noted above, the linear and angular momentum principles tell us that the total linear and total angular momenta are unaffected by the internal actuators so this problem divides nicely into three distinct parts: 1) controlling the position of the system center of mass, 2) controlling the attitude of the main body, and 3) controlling the orientations of the respective sensors. Clearly part 1 is independent of parts 2 and 3; however part 3 acts as a disturbance to part 2 and visa versa.

By way of contrast, in the space robot gross motion control problem we are interested in controlling the base body position and orientation so as to place or maintain the manipulator arm tip position(s) in a desired workspace. Since we are interested in the actual positions of the manipulators (as opposed to the orientation of sensors in the case of a satellite) we must control both the base position and orientation rather than just the system center of mass position. In particular, if we are operating in a densely populated workplace (e.g. in the middle of space station construction) we must know the exact extents of our base body and all of its appendages. There are, of course, certain circumstances were we might be executing a large motion slew (one which requires base motion in order to complete) away from any potential obstacles. In this case we may not be concerned with the manipulator tip positions or the precise base position and thus can control the position of the system center of mass. Therefore, a number of different control situation may arise and enumerated below:

- The robot is in position to perform some task; however, since their is no way for it to anchor itself to the environment it is working in[7] we must perform station keeping of the base position and orientation to keep the manipulators within the required workspace.
- The robot is executing a large motion slew along some prescribed trajectory with a large corridor of unobstructed space surrounding it. In this case, we can control position of the system center of mass without concerning ourselves with the actual location or orientation of the base and the manipulators.
- The robot is attempting to intercept a free floating object such as a satellite and must execute a trajectory which will rendezvous with the target in such a way as to match both its position and velocity at the intercept point. In planning and performing such a trajectory we must assure that the base position and orientation allow the manipulators sufficient freedom of reach so that they can successfully grapple the target without running into the limits of their workspace.

Clearly it is this last case which is the most challenging and therefore the most interesting. In order to successfully capture a free floating target we must simultaneously control our manipulator tip positions as well as the robot base position and orientation. Since the corresponding states are coupled with each other it becomes necessary to view the system as whole rather than as decoupled parts. Simply generating an intercept trajectory which is realizable given the limited actuator authority available, the ever present dynamic constraints imposed by a free floating robot[8], and any temporal constraints which might exist(e.g. the object might float out of reach if we do not get to it soon enough) presents a formidable problem. Various trajectory generation, validation, and control algorithms which address these issues will be the focus of a future paper.

# References

[1] John J. Craig. *Introduction to Robotics Mechanics and Control.* Addison-Wesley, Reading, MA, 1986.

[2] Oussama Khatib. Dynamic control of manipulators in operational space. In *Proceedings of the*

[7] If there was some way for the robot to anchor itself to its environment, we would not have the "free-flying" robot control problem at all.

[8] Imagine driving a car where the only way to slow down was to put it in reverse.

*Sixth CISM-IFToMM Congress on Theory of Machines and Mechanisms*, pages 1128–1131, New Delhi, India, December 1983.

[3] Harold L. Alexander. *Experiments in Control of Satellite Manipulators.* PhD thesis, Stanford University, Department of Electrical Engineering, Stanford, CA 94305, December 1987.

[4] Y. Umetani and K. Yoshida. Continuous path control of space manipulators mounted on omv. *ACTA Astronautica*, 15(12):981–986, 87.

[5] Thomas R. Kane and David A. Levinson. *Dynamics: Theory and Application.* McGraw-Hill Series in Mechanical Engineering. McGraw-Hill Book Company, New York, NY, 1985.

[6] Dan E. Rosenthal. Order N Formulation for Equations of Motion of Multibody Systems. In G. Man and R. Laskin, editors, *Proceedings of the Workshop on Multibody Simulation*, pages 1122–1150, Pasadena, CA, April 1988. NASA Jet Propulsion Laboratory. JPL D-5190, Volume III.

[7] G. Rodriguez and K. Kreutz. Recursive mass matrix factorization and inversion. Technical report, NASA Jet Propulsion Laboratory, Pasadena, CA, March 1988. JPL Publication 88–11.

# NEXT GENERATION SPACE ROBOT

Tsutomu IWATA, Mitsushige ODA, Ryoichi IMAI

Tsukuba Space Center
National Space Development Agency of Japan
2-1-1, Sengen, Tsukuba-City, 305, Japan

## Abstract

This paper outlines our recent research effort on the next generation space robots. The goals of this research are to develop the fundamental technologies and to acquire the design parameters of the next generation space robot. Visual sensing and perception, dexterous manipulation, man-machine interface and AI techniques such as task planning are identified as the key technologies. In addition, the design study of the ground testbed model and the simulator to evaluate the performance of the integrated technologies are now under way.

## 1. Introduction

The advent of the Space Station and the space platforms in late 1990s will increase the necessity for the in-orbit servicing activities such as assembly, inspection, equipment exchange, resupply, repair and maintenance. To meet these various in-orbit servicing demands with much safety and efficiency, the development and the application of the intelligent and flexible space robots are required.

The National Space Development Agency of Japan ( NASDA ) started the research and development of the Remote Manipulator System attached to the Space Station Japanese Experiment Module ( JEM ) in 1985. We have also initiated the study on the advanced space robots, -Next Generation Space Robot-, based on the technologies developed in the JEM Remote Manipulator System project. The phase 1 research was started in 1988 and is scheduled to end in 1991. This research encompasses mission analysis, identification of key technologies, system architecture, and performance demonstration by the ground testbed model.

## 2.Generation of the Space Robot

We classify the space robots in three generations according to the degree of autonomy as given below :

First generation : programmed control and/or proximately operated manual control robots such as Space Shuttle Remote Manipulator.

Second generation : teleoperation with some some autonomy ( semi-autonomous ) robots.

Third generation : highly intelligent and autonomous robots.

JEM Remote Manipulator System which is a 10-meter-long, 6 degree-of-freedom and master-slave/programmed control manipulator is evidently a first generation space robot. Our research efforts are concentrated on the second generation space robots. The characteristics of each generation of the space robot are summarized in Table-1.

Table-1. CHARACTERISTICS OF EACH GENERATION OF THE SPACE ROBOT

| CHARACTERISTICS | 1st GENERATION | 2nd GENERATION | 3rd GENERATION |
|---|---|---|---|
| OBJECT | KNOWN | KNOWN | UNKNOWN |
| SERVICING AREA | IN AND AROUND THE VEHICLE | ·EARTH ORBIT<br>·SPACE LABORATORY | ·DEEP SPACE<br>·LUNAR AND MARS SURFACE |
| MISSION | ·ORU EXCHANGE<br>·ASSEMBLY (STANDARD AND COOPERATIVE PARTS)<br>·BERTHING<br>·VEHICLE TOW | ·AUTOMATED SPACE EXPERIMENT<br>·ASSEMBLY (FLEXIBLE, FRAGILE PARTS)<br>·PLATFORM MAINTENANCE<br>RESUPPLY<br>INSPECTION<br>REPAIR<br>·RETRIEVAL | ·CONSTRUCTION<br>·FULL EVA<br>·EXPLORATION<br>·MANUFACTURING |
| MAN-MACHINE I/F | ·MASTER-SLAVE CONTROL<br>·VISUAL IMAGE<br>·FORCE FEEDBACK | ·TASK LEVEL LANGUAGE CONTROL<br>·TELEPRESENCE | ·NATURAL LANGUAGE INTERFACE |
| MANIPULATION | ·SINGLE ARM<br>·STANDARD END EFFECTOR<br>·GRIPPERS | ·COOPERATIVE ARMS<br>·INTERCHANGEABLE DEXTEROUS HANDS<br>·FLEXIBLE ARM | ·ROBOTS COOPERATION<br>·NON-CONTACT MANIPULATION (APPLICATION OF THE ELECTORO-MAGNETIC FORCE) |
| SENSING & PERCEPTION | ·VISION<br>·FORCE/TORQUE<br>·PROXIMITY/TOUCH | ·VISUAL RECOGNITION (DATA BASED MATCHING)<br>·SENSOR FUSION<br>·TEXTURE | ·ENVIRONMENT RECOGNITION |
| SYSTEM EXAMPLE | JEMRMS | OSV | LUNAR ROVER |

## 3.Mission of the Second Generation Space Robot

Three different types of the second generation space robots are considered to analyze the mission of the future space activities around 2000.

Fixed based robot : This type of robot conducts routine services and rather simple, repetitive space experiments mainly to save the human workload in space.

Truss mobile robot : This type of robot moves along the truss structure of the large spacecraft like MSC ( Mobile Servicing Center : Canada ) and conducts the external activities in place or in support of manned EVA.

Free Flying robot : This type of robot has the capability of orbital maneuvering / changing , and conducts the in-orbit services to the unmanned platforms.

The possible missions and tasks of the above systems are summarized in Table-2. Fig.-1 shows one concept of the free flying robots temporarily named OSV (Orbital Servicing Vehicle ).

Table-2. MISSION OF EACH TYPE OF THE SPACE ROBOT

| MISSION | TASK | FIXED BASED | TRUSS MOBILE | FREE FLYING |
|---|---|---|---|---|
| SPACE EXPERIMENT | MONITOR, SAMPLE AND EQUIPMENT EXCHANGE, EQUIPMENT HANDLING AND OPERATION | ○ | – | – |
| MAINTENANCE | VISUAL, ELECTRICAL AND MECHANICAL INSPECTION, ORU AND MODULE EXCHANGE | ○ | ○ | ○ |
| RESUPPLY | PROPELLANT, FLUID AND GASES TRANSFER | – | ○ | ○ |
| ASSEMBLY | TRUSS CONSTRUCTION AND EXTENSION, MODULE LOADING, CABLE MATE/DEMATE | – | ○ | ○ |
| REPAIR | FAILURE UNIT EXCHANGE, THERMAL BLANKET REPLACEMENT | – | ○ | ○ |
| BERTHING | PLATFORM, HOPE AND LOGISTICS VEHICLE BERTHING AND RELEASING | – | ○ | – |
| CAPTURE | SPACE DEBRIS AND FAILURE SATELLITE RETRIEVAL | – | ○ | ○ |
| SPACE TOWING | RENDEZVOUS DOCKING WITH PLATFORMS AND LOGISTICS VEHICLE, ORBITAL TRANSFER | – | – | ○ |

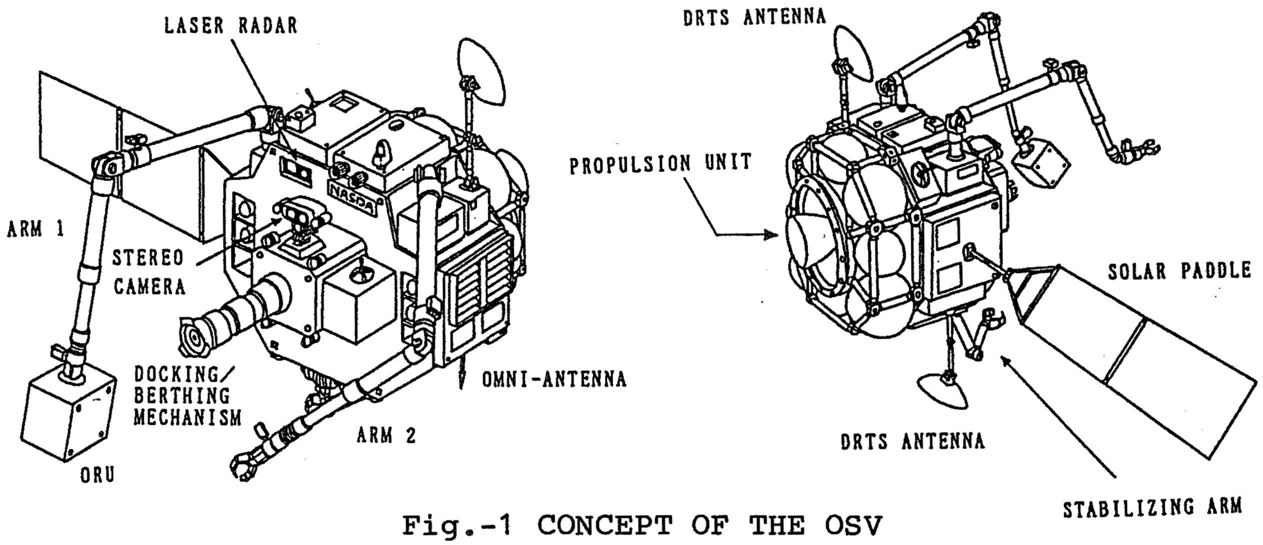


Fig.-1 CONCEPT OF THE OSV

### 4.Key Technologies

Based on the mission analysis, key technologies needed for the second generation space robots are identified as shown in Fig.-2. Teleoperation combined with some autonomous control function, which is called supervisory control or telerobotics, is the primary concern to relief of the human workload and to compensate for the time delay in the remote control loop of the second generation space robot. Dexterous manipulation is also essential to increase the applicability of the robots to the various space activities.

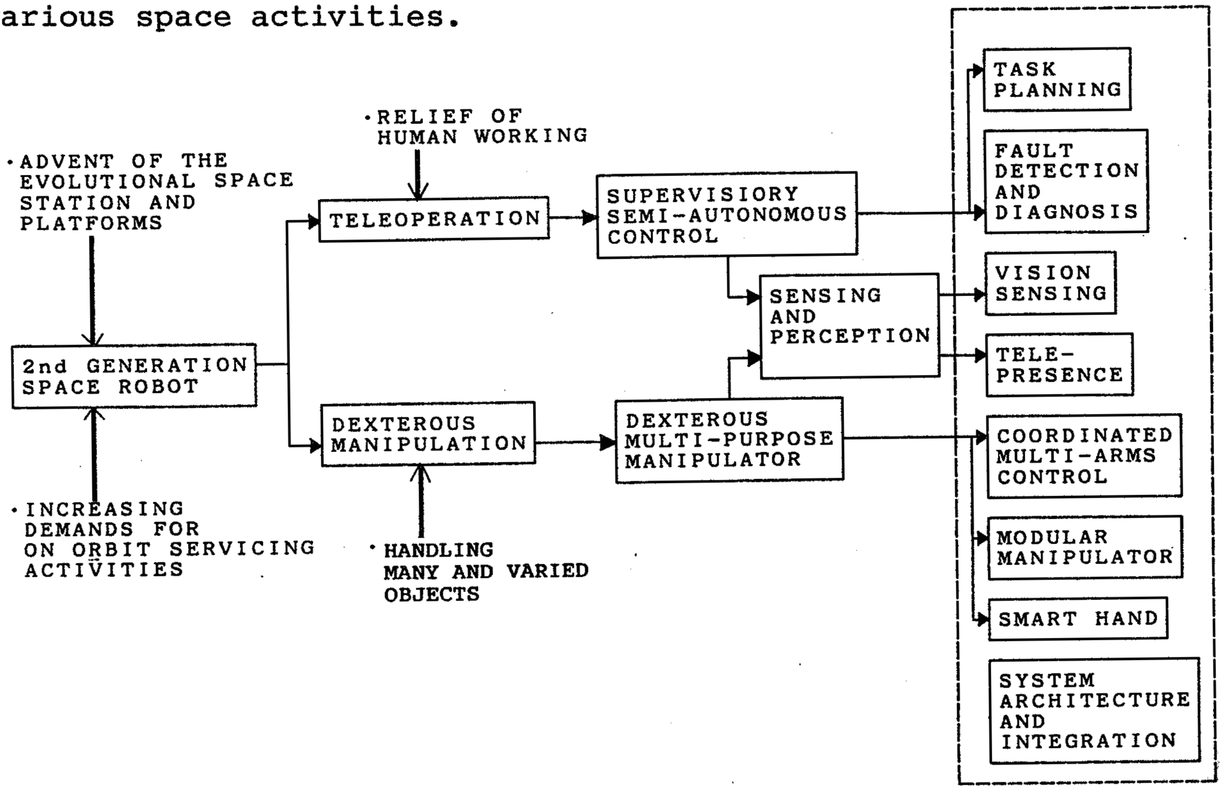

·ADVENT OF THE EVOLUTIONAL SPACE STATION AND PLATFORMS
2nd GENERATION SPACE ROBOT
·INCREASING DEMANDS FOR ON ORBIT SERVICING ACTIVITIES
·RELIEF OF HUMAN WORKING
TELEOPERATION
DEXTEROUS MANIPULATION
· HANDLING MANY AND VARIED OBJECTS
SUPERVISIORY SEMI-AUTONOMOUS CONTROL
SENSING AND PERCEPTION
DEXTEROUS MULTI-PURPOSE MANIPULATOR
TASK PLANNING
FAULT DETECTION AND DIAGNOSIS
VISION SENSING
TELE-PRESENCE
COORDINATED MULTI-ARMS CONTROL
MODULAR MANIPULATOR
SMART HAND
SYSTEM ARCHITECTURE AND INTEGRATION

Fig.-2 KEY TECHNOLOGIES OF THE SECOND GENERATION SPACE ROBOT

The goals of our phase 1 research on those key technologies are as follows.

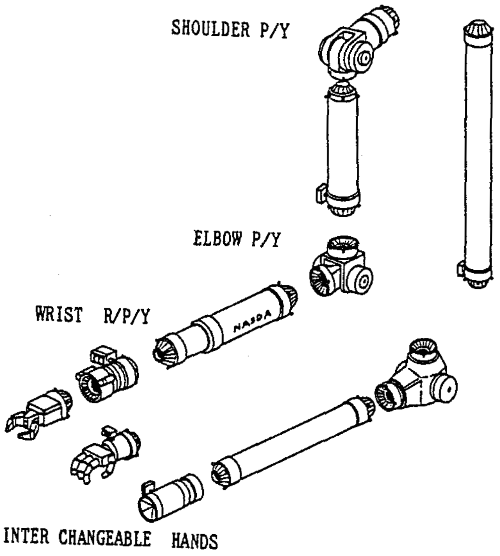
Task planning : Task planning covers path planning which generates the sequence of the robot motion and the obstacle avoidance using task rules, geometric data base and sensor data. The most important point is the robustness of the planning system against the uncertainties of the real world.

Vision sensing : Acquisition, tracking and recognition of the marked or known objects by the vision sensor under the conditions of the simulated orbital lighting.

Telepresence : The objectives of this work are to develop and evaluate the advanced man-machine interface technologies such as the application of the 3-D vision and predicted simulation vision.

Dexterous manipulation : This work covers the development and the evaluation of the coordinated multi-arms control, modular reconfigurable manipulator and the smart hands. Fig.-3 shows the concept of the modular manipulator. The manipulator system is reconfigurable by exchanging standard sized arm links, actuator units and hands.

In addition, the evaluation of the system architecture and integration with the ground testbed model is the major concern in this phase.



EXAMPLE FEATURES OF THE MANIPULATOR CONFIGURATION FOR THE OSV

| | |
|---|---|
| ARM LENGTH | 4-5m/arm |
| DEGREE OF FREEDOM | 8 (1:trnaslational<br>7:rotational) |
| TIP SPEED | 200mm/sec ( max.) |
| TIP FORCE | 100N ( max.) |
| TORQUE | 50Nm ( max.) |
| ACCURACY | less than 2mm |
| OBJECT MASS | 1 ton |
| HAND | interchangeable |
| COMPLIANCE | passive & active (software) control |
| ARM MASS | 250kg/arm |
| POWER | 500W/arm ( max.) |

Fig.-3 CONCEPT OF THE MODULAR MANIPULATOR

## 5.Testbed model and the simulator

The objectives of the ground testbed model and the simulator are to evaluate the performance of the key technologies and the applicability of the semi-autonomous robots to the near-future space activities. The system configuration and the block diagrams of the testbed model and the simulator employed in the phase 1 research are shown in Fig.-4 and Fig.-5 respectively.

The simulator consists of the testbed model, experimental work bench ,man-machine interface system and work stations for the task planning and graphic simulations. The experimental work bench is easily reconfigurable and applicable to the various in-orbit servicing simulations. The testbed model which consists of manipulators , visual sensors and local control unit is mounted on the mobile platform of the motion simulator. In the experiment of the free flying target capturing, this system can simulate the relative 6 degree-of-freedom motion between the target and the testbed model by the coordinated motion between the gimbal system of the target and the mobile and rotational platform.

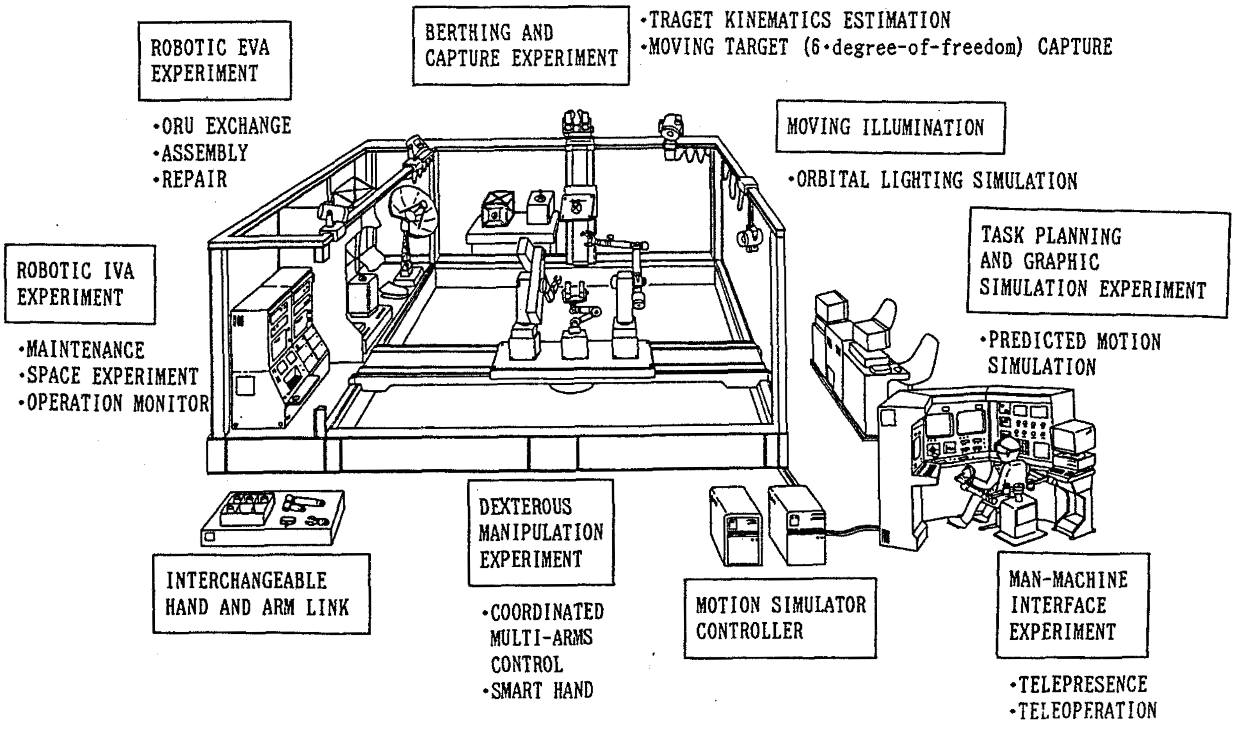

ROBOTIC EVA EXPERIMENT
•ORU EXCHANGE
•ASSEMBLY
•REPAIR
BERTHING AND CAPTURE EXPERIMENT
•TRAGET KINEMATICS ESTIMATION
•MOVING TARGET (6•degree-of-freedom) CAPTURE
MOVING ILLUMINATION
•ORBITAL LIGHTING SIMULATION
ROBOTIC IVA EXPERIMENT
•MAINTENANCE
•SPACE EXPERIMENT
•OPERATION MONITOR
TASK PLANNING AND GRAPHIC SIMULATION EXPERIMENT
•PREDICTED MOTION SIMULATION
INTERCHANGEABLE HAND AND ARM LINK
DEXTEROUS MANIPULATION EXPERIMENT
•COORDINATED MULTI-ARMS CONTROL
•SMART HAND
MOTION SIMULATOR CONTROLLER
MAN-MACHINE INTERFACE EXPERIMENT
•TELEPRESENCE
•TELEOPFRATION

Fig.-4 SYSTEM CONFIGURATION OF THE TESTBED MODEL AND THE SIMULATOR

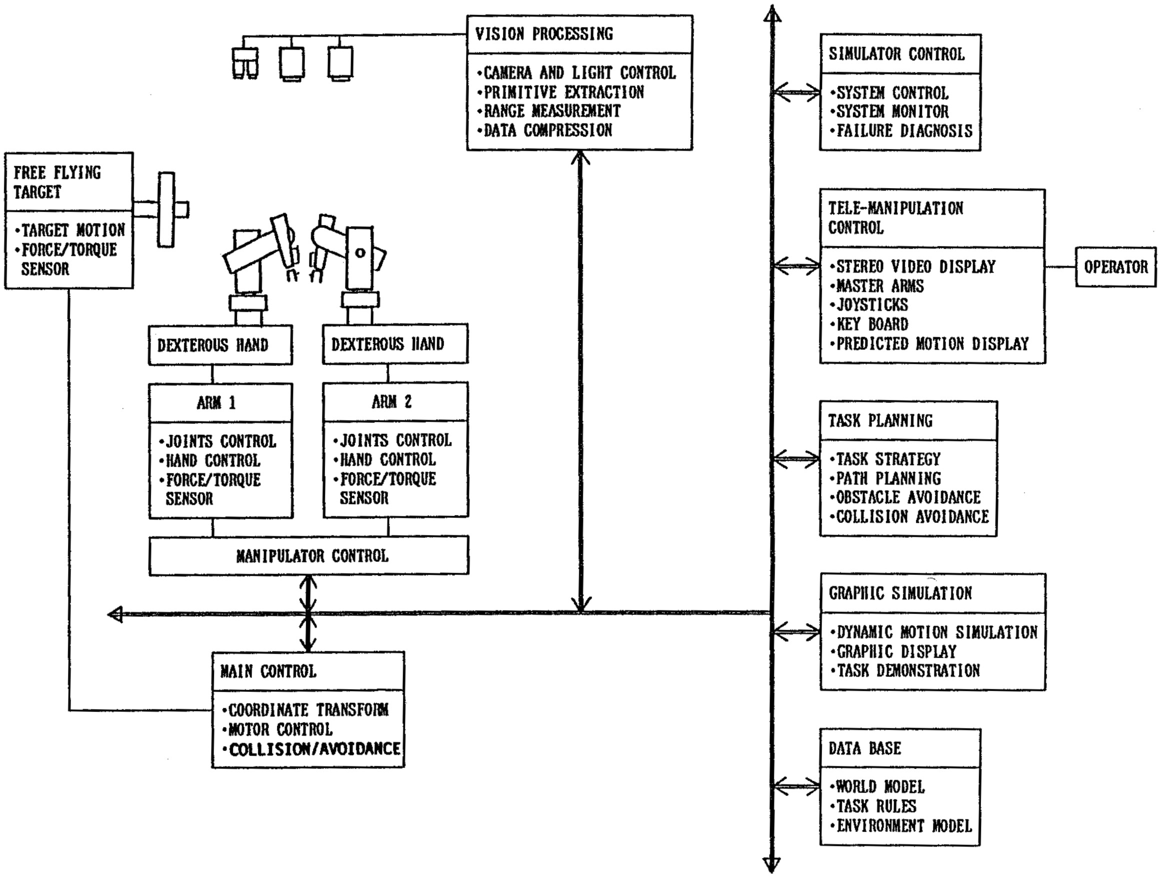


Fig.-5 BLOCK DIAGRAMS OF THE TESTBED MODEL AND THE SIMULATOR

## 6. Research and Development Scenario

Fig.-6 shows the research and development schedule of the second generation space robots. The research and development is divided in two phases. The major objectives of the phase 1 research are given below.

- obtain the system concept of the second generation space robot
- analyze the in-orbit robotic servicing tasks
- research and development on the semi-autonomous and the dexterous manipulation technologies
- evaluate the performance of the integrated technologies by the ground testbed model and the simulator

The feasibility study on the specific robot systems and the subsystem development will be conducted in the phase 2 research planned to start in 1992. In addition, the in-orbit robotics experiment with the Space Technology Experiment Platform ( STEP ) and the JEM is planned around 1997. The performance and the applicability of the integrated robotic technologies , a sequence of the task procedures,and the teleoperation/semi-autonomous control scheme will be verified in the in-orbit experiment.

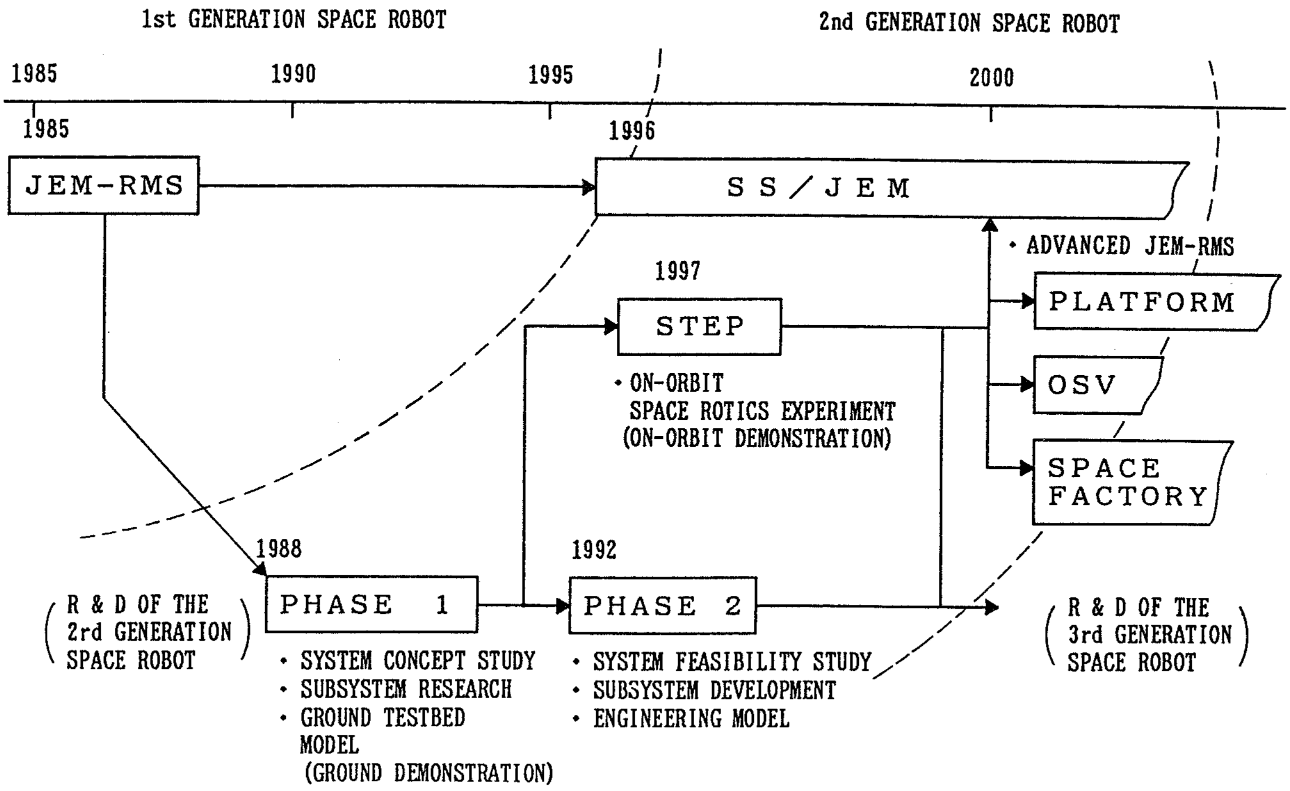


Fig.-6 RESEARCH AND DEVELOPMENT SCENARIO

## 7.References

1. " Study Report on Robot-Satellite" ( NASDA Contract Report ), Institute for Future Technology, March 1984 ( in Japanese ).

2." Automation & Robotics for the National Space Program", California Space Institute, University of California, February 1985.

3. Montemerlo M D ; "NASA's Automation and Robotics Technology Development Program", IEEE Conference on Automation & Robotics , 1986.

4. Blais Th, Lacombe J L, Wetzel P ; "Utilization of Robotics and Teleoperation for Future In-Orbit Operations", Proc. First European In-Orbit Operations Technology Symposium ( ESA SP-272 ), November 1987.

5. Pronk C N A, Ersu E, Lippary A L, Elfving A ; "Definition of the EUROSIM Simulator Subsystem" , Proc. First European In-Orbit Operations Technology Symposium ( ESA SP-272 ), November 1987.

6. Myers J K ; "System Integration of a Telerobotic Demonstration System ( TDS ) Testbed", First Annual Workshop on Space Operations Automation & Robotics ( NASA CP-2491 ), 1987.

7. Ikeuchi M, Oda M ;" Space Robotics in Japan", AIAA/NASA First Symposium on Space Robotics & Automation, November 1988.

# MANIPULATOR COORDINATION

# COORDINATION IN A HIERARCHICAL MULTI-ACTUATOR CONTROLLER

A. Meystel

Department of Electrical and Computer Engineering
Drexel University
Philadelphia, PA 19104

## Abstract

A hierarchical multi-actuator controller is represented as a multi-resolutional information (knowledge) system utilizing a number of intelligent modules with decision making capabilities. The laws of multi-resolutional information (knowledge) organization and processing are presumed to be satisfied including the rules of dealing with redundant knowledge. A general case is considered in which a process to be controlled by a multiplicity of actuators is a distributed one and the condition of distribution can be formulated analytically. Operation of a lumped multi-actuator process is a particular case which has a broad practical application.

***Key Words:*** *Coordination, Decision Making, Decoupling, Generalization, Interpretation, Knowledge, Multiresolutional, Negotiations, Representation, Sensor Integration.*

## 1. Introduction

This paper was motivated by a specifics of the coordination processes in the systems with decoupled subsystems (e.g. using feedback linearization, and decoupling, or FLDT as in [1]). It is tempting to consider the decoupled subsystems as independent decision makers (though with an intentionally distorted view of the world created by FLDT). Nevertheless one can expect that the process of negotiations among the "independent" decision makers can be simulated in an on-line intelligent controller. The controller is based upon the structure of "intelligent module" first presented in [2,3]. General familiarization with the paradigm of multiresolutional control systems can be done by reading [4].

The progress in technology in recent years can be characterized by the increase of interest to the problem of coordination. Our research was affected primarily by papers [5-7], and conceptually can be considered a further extension of [8] to the domain of multiresolutional control and simulation of the process of negotiating independent controllers.

Two types of coordination are analyzed in this paper: inter-level , and within-level coordination ILC and WLC. ILC is performed by distribution of information and decision making activities among the levels of resolution in such a way as to minimize the cost-functional (in our examples we consider two cost functionals: time of computation and time of operation). ILC does not necessarily require a subsystem of coordination: a set of rules can properly handle all ILC activities. WLC is dealing with multiple decision making agents at a level whose decisions affect each other and eventually affect the overall performance of the system. Thus, the amount of situation-related coordination activities is large even in comparatively simple cases, and a separate coordination subsystem of WLC is required.

WLC is performed by transforming the input information (knowledge) sets in such a way as to decouple the planning/control activities of the parallel agents, and make their decision making

C-4

processes independent of each other at each moment of time. An *intentionality map* is being developed and maintained by the subsystem of WLC for each decision making agent. (This map is becoming an important component in the subsystem of learning). Coordination based on intentionality map is illustrated for examples of parallel processing. A computer architecture of a nested hierarchical controller is introduced for a hierarchical multi-actuator system.

## 2. Structure of the system

The overall structure of the system under consideration is shown in Figure 1. The Knowledge Based Organizer, and the Dispatcher both are submitting information to the General Coordinator which provides joint consistent operation of the set of N Coordinators. Each of them coordinates activities of a set of m subsystems consisting of controllers, actuators, and sensors. Each elementary closed-loop subsystem of controller-actuator-sensor equips a particular elementary process which can be characterized by a relative local independence. Sensors provide information for closing the loop of the subsystem. Simultaneously all m systems of sensors merge their information within the system of Sensor Integration related to the particular Coordinator, and provide the main feedback for this Coordinator.

The procedures of the trajectory generation are based upon the concept of open loop controller. The desirable trajectory is planned based upon the principles of optimum (e.g. minimum time) control, and then it is inverted in order to find the input which is required to provide the optimum motion. Since the optimum trajectory is computed based upon definite assumptions the real values of physical parameters lead to deviations of real trajectory from the prescribed plan. Thus, the feedback control applied in this system is dedicated to only compensate the deviations from the prescribed plan.

The concept of minimum time control for a system with FLDT leads to a particular structure of a program: all compensations are supposed to be done by corrections in switching times. Thus, a number of problems typical for "tailoring" the appropriate trajectories is not necessary here.

All Sensor Integrators submit their information to the General Sensor Integrator which serves as a feedback for the General Coordinator. It transforms all multiplicity of sensor information into the language understandable for the General Coordinator. This type of structure can be applied within a variety of systems including multilink manipulators, autonomous robots, and complicated systems of material processing similar to the one described in [9] which employs the system shown in Figure 1.

Each of the controllers has a structure demonstrated in Figure 2,a. Information from sensor S is being processed in a multiresolutional fashion as described in [2-4] interacting with the domain and context knowledge within the multiresolutional hierarchy of K, generating plans and controls within the similar p/c subsystem. The simplified representation of this system is shown in Figure 2,b. It is essential for understanding the processes of control in a multiresolutional system that the system shown in Figure 2,a can be considered as a system with three parallel loops as shown in Figure 3. These parallel loops are working (and designed) as if they were independent control loops. Coordination is provided by maintaining consistency of using generalization rules within the hierarchies P, K, and P/C. Resolution levels of these systems (tessellata) are communicating with each other.

Figure 1 can be redrawn in a simplified form using notation Figure 2,b. It is shown in Figure 4. From this illustration it is clear that coordinator is the real controller of the overall process. The horizontal lines show coupling which takes place between knowledge bases, perceptual systems, planning/control systems, sensors, actuators, and especially within the process where coupling is usually playing very important role, seriously affecting the design of controllers as well as motion control programs. Using principles demonstrated in [1] for a multilink manipulator, we apply method of generalized transformation for developing a system of linearization and decoupling.

The decoupled system is shown in Figure 5. It incorporates a new subsystem: a Decoupler which contains a multiplicity of look-up tables computed off-line in order to provide pseudo-independent actuation of the machine. Interestingly enough, the process is being decomposed in a multiplicity of pseudo-independent fictitious processes that can be controlled by 'independent" decision-makers.

## 3. Process of Coordination

In such a system each of the actuators is working in its own state space and has a)its own pseudo-goal, as well as b) its own system of constraints. Both position of the goal and configuration of constraints must be constantly recomputed under supervision of Decoupler and Coordinator. Assume the initial and the goal location are shown by the circles (see Figure 6). Configuration of the obstacles is shown by the bold line. In the beginning the subsystem of P/C computes the path to be traversed. Then the FLDT driven controller computes the system of jerks, accelerations, and speeds to be followed in order to provide minimum time operation (also bold lines on time diagrams). The algorithm of coordination works in the following steps.

Step 1. Coordinator builds the set of intentionality maps for all subsystems (parallel operation is presumed).
Step 2. Intentionality maps are distributed among the subsystems.
Step 2. Planning/control subsystem in each of the controllers performs the set of planning/control procedures (top-down in each tessellatum of the whole set of multiresolutional intentionality map).
Step 4. All actuators perform their first step of the motion.
Step 5. Sensors submit information to the subsystems of perception, subsystem of knowledge organization updates map, map is submitted to Coordinator.
Step 6. Together with FLDT system Coordinator transforms these maps into the updated Intentionality maps (parallel operation is presumed).
Step 7. Go to Step 1.

As soon as the motion started, Coordinator recomputes the configuration of the constraint boundary taking in account the information about motion of all actuators of the system. It is done by creating the "Intentionality Map" for each of the actuators. Planning/control subsystem replans the path for the new configuration of constraints. Then Decoupler recomputes the switching times, and the motion continues in a little bit different direction. This process is repeating each discrete of time accepted in the system. As a result the trajectory of real motion has a configuration which under no circumstances could be computed in the beginning unless the process of negotiations would not be provided in a comprehensive way. The problem is not yet solved concerning the degree of optimality in the set of trajectories obtained in this way.

When the set of coordinators has a General Coordinator above them the system operates as follows.

Step 1. Process the set of changes in the sensor information delivered from each of the particular sensors into a set of increments of the actuator trajectories which serve as descriptions of the motion of the all set of actuators (parallel operation is presumed).
Step 2. Generalize the set of increments of the actuator trajectories into an increment of the vector-trajectory which serves as a description of the overall process development.
Step 3. Submit to the General Coordinator the set of all increments of the vector-trajectory for the process (parallel operation is presumed).
Step 4. Develop the set of Coordinator Intentionality maps (parallel operation is presumed).
Step 5. Submit the set of Coordinator Intentionality maps to all Coordinators (parallel operation is presumed).

The system of coordination described in this paper is flexible since it allows for constant updating of the information in all subsystems during the process of MRKP with no interruptions, or delays.

## References

1. A. Meystel, A. Guez, "Optimum Positioning of a Servomechanism", Proc. of 21-st IEEE Conference on Decision and Control, Vol. 3, Orlando, Fl., 1982

2. A. Meystel, "Theoretical Foundations of Planning and Navigation for Autonomous Robots", *International Journal of Intelligent Systems*, Vol. II, No. 2, Summer 1987, p.p. 73-128

3. A. Meystel, "Intelligent Control in Robotics", *Journal of Robotic Systems*, 1988

4. A.Meystel,"Techniques and Potential Capabilities of Multiresolutional Information (Knowledge) Processing" (in these Proceedings)

5. J. Fortuni-Amat, J. Talavage, "A solution Procedure for the Hierarchical Coordination of Constrained Optimizing System", Int. J. Systems Sci., Vol. 12, No. 2, 1981, p.p. 133-146

6. E. Durfee, V. Lesser, D. Corkill, "Coherent Cooperation Among Communicating Problem Solvers", IEEE Transactions on Computers, Vol. C-36, No.11, 1987, p.p. 1275-1291

7. Y. Bestaoui, "Adaptive Hierarchical Control for Robotic Manipulators", Robotics, Vol. 4, 1988, p.p. 145-155

8. V. Hayward, S. Hayati, "KALI: An Environment for the Programming and Control of Cooperative Manipulators", Proc. of the 1988 ACC, Vol. 1, Atlanta, GA, 1988, p.p.473-478

9. D. Apelian, A. Meystel, "Knowledge Based Control of Material Processing" (to be published in Proceedings of the 118-th TMS Annual Meeting, February 27-March 2, 1989, Las Vegas, Nevada; Volume on Symposium "Metallurgical Processes for the Year 2000 and Beyond", Publ. by The Minerals, Metals, and Materials Society, 1989).

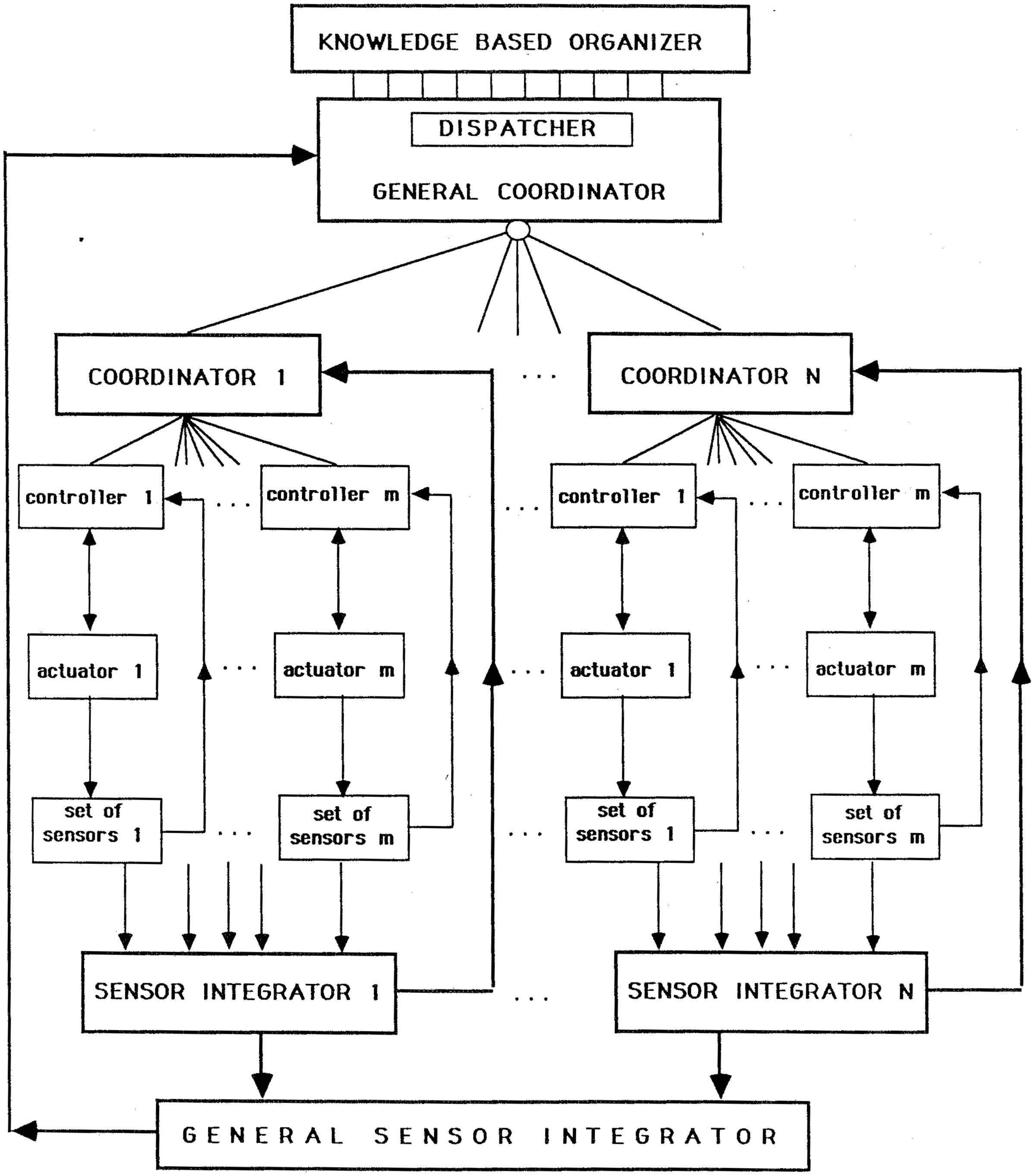


Figure 1. The structure of Multiactuator multiresolutional control system with two levels of coordination.

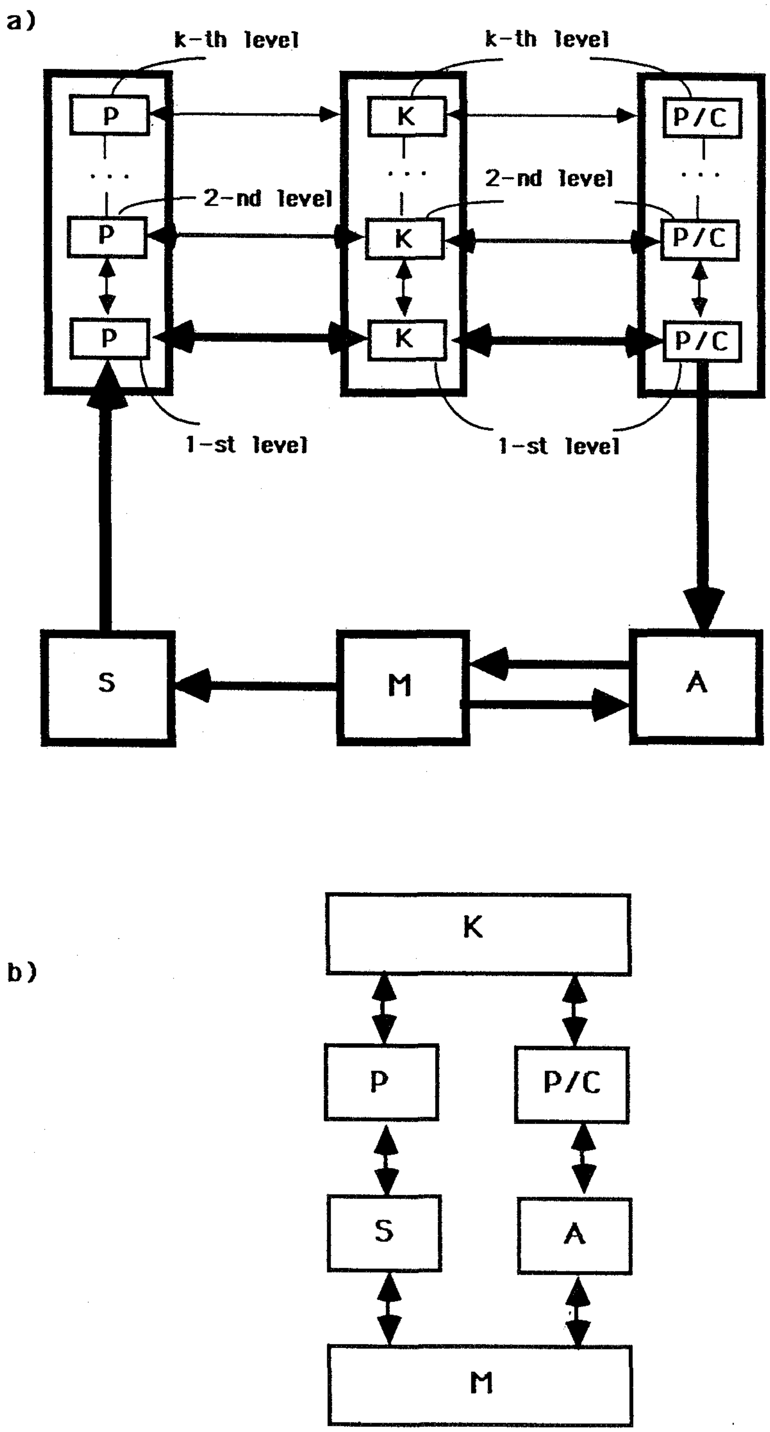


Figure 2. Structure of a single controller

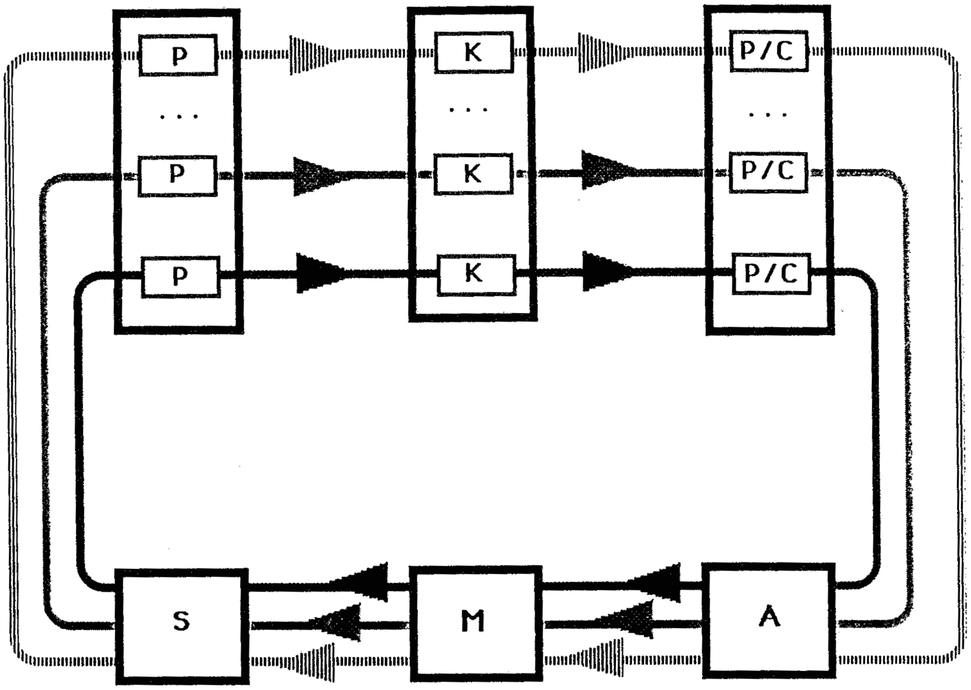


Figure 3. Parallel control loops in a multiresolutional controller

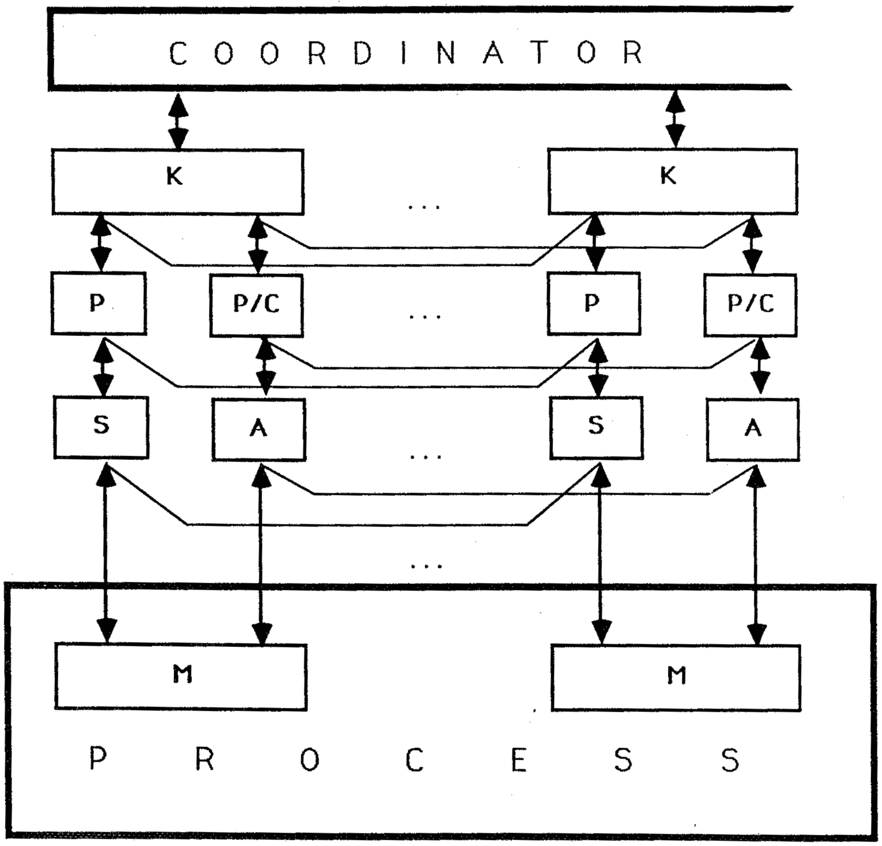


Figure 4. Coupled control structure of the overall process.

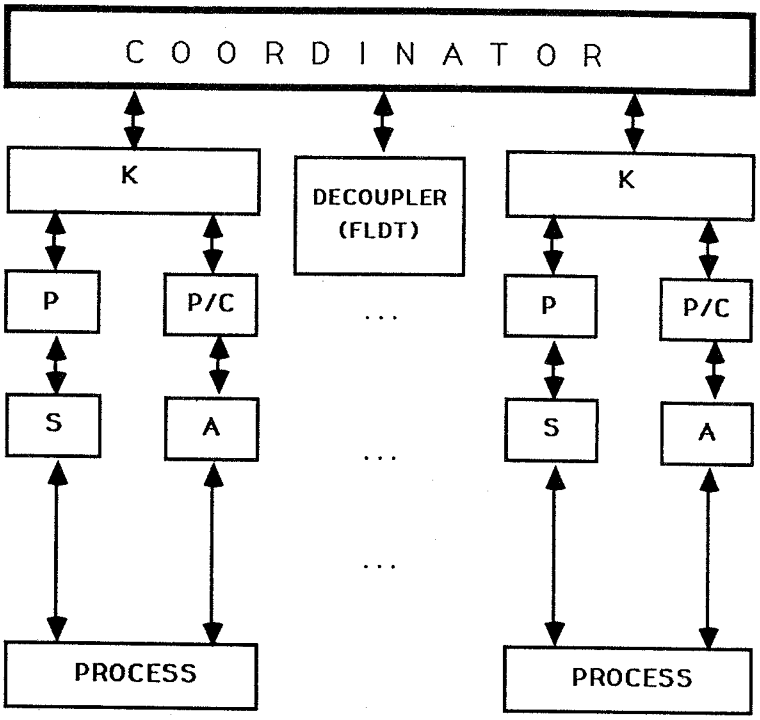


Figure 5. Decoupled control structure

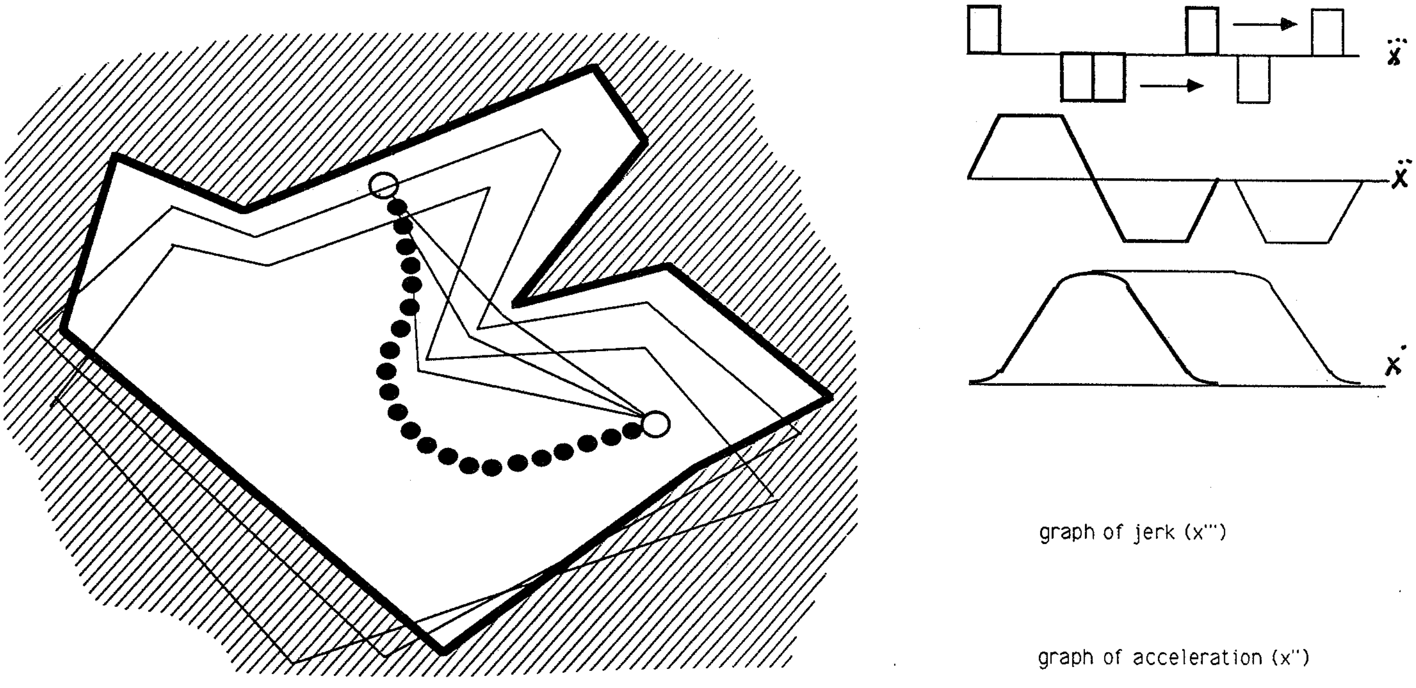


Figure 6. Motion shown on an intentionality map

[bold-planned control,
thin lines-postponed deceleration]

# DISTRIBUTED COMMUNICATIONS AND CONTROL NETWORK FOR ROBOTIC MINING

William H. Schiffbauer
Pittsburgh Research Center
U.S. Bureau of Mines
Pittsburgh, PA 15236

## ABSTRACT

The application of robotics to coal mining machines is one approach the U.S. Bureau of Mines is pursuing to increase productivity while providing enhanced safety for the coal miner. Toward that end, a network composed of microcontrollers, computers, expert systems, real-time operating systems, and a variety of program languages are being integrated by the Bureau that will act as the backbone for intelligent machine operation.

Actual mining machines, including a few customized ones, have been given tele-robotic semi-autonomous capabilities by applying the described network. Control devices, intelligent sensors and computers onboard these machines are showing promise of achieving improved mining productivity and safety benefits. Current research using these machines involves navigation, multiple machine interaction, machine diagnostics, mineral detection, and graphical machine representation. Guidance sensors and systems employed include: sonar, laser rangers, gyroscopes, magnetometers, clinometers, and accelerometers.

This paper provides information on the Bureau's network of hardware/ software and its implementation on mining machines. Anticipated coal production operations using the network are discussed. A parallelism is also drawn between the direction of present day underground coal mining research to how the lunar soil (regolith) may be mined. A conceptual lunar mining operation that employs a distributed communication and control network is detailed.

## 1. INTRODUCTION

Research is being conducted by the Bureau of Mines to make mining safer for the worker by minimizing the hazards to which he is exposed. Additionally, the Bureau seeks to increase worker efficiency by providing machines with enhanced capabilities employing the latest robotic technologies.

Presently, the work is concentrating on the coal mine face area (coal extraction point). A typical face area is shown in figure 1. It includes a continuous miner which extracts the coal, a roof-bolter which provides a means of securing the roof where coal has been extracted, and a shuttle car which transports the coal from the face area to a conveyor belt. The conveyor belt transports the coal to the surface where it is usually processed into a cleaner product. In general, the face machinery is powered from a central point using ac power. The workers in such mine face areas are exposed to numerous safety hazards including being crushed by roof falls; being injured by mine equipment, explosions, coal dust inhalation, loss of hearing, and electrocution. The inefficiencies of such a face area are usually caused by equipment failures, slow transport of coal from the continuous miner to the conveyor belt (shuttle car delays), and maintenance of the equipment.

An obvious approach to improve the safety of the workers is to remove them from the face and into a secure area. Remote-controlled machines are now available that at least get the person off the machine and away from some roof falls, but the worker is still exposed to hazardous situations by being in the face area. Increasing the efficiency of the coal production process can be addressed in a number of ways. The application of robotics, computer diagnostics, and machine redesigns will help.

The present focus of one Bureau research team, in response to the needs of safety and increased productivity, is to develop an autonomous coal extraction machine. The work involves numerous research areas [1-5] that are being investigated as logically and functionally distinct entities.

This paper describes the network that facilitates the interchanging of information between entities and also provides control of some of those entities. The architecture of the distributed network's hardware and software as applied to a few applications is described, as well as the machinery on which it is installed. Also, a hypothetical concept for lunar liquid oxygen (LOX) production is described and the network which could be used to control it is also discussed.

## 2. THE RESEARCH ENVIRONMENT

The Bureau has a mining equipment test facility (METF) at its Pittsburgh Research Center (PRC). A large building houses a few full-sized mining machines along with a large block of simulated coal (coalcrete). Figure 2 shows the inside of the building and the general layout of the research area. Shown in the center of the picture is a Joy 16CM[1] continuous mining machine which serves as the testbed for our experiments. The coalcrete is used for dynamic testing of mining machines under computer control.

## 3. THE JOY 16CM MINING MACHINE

The Joy 16CM is a continuous coal mining machine designed to operate underground. It is 16 ft wide, 4 ft high, 40 ft long, and weighs 50 tons. A control devices combination of electric and hydraulic systems run the various machine. The Bureau added a control computer that parallels the human control functions to the machine. Additionally, sensors were placed on all the machine appendages and environmental systems.

## 4. THE MINING MACHINE COMPUTER SYSTEM

A Bureau-designed onboard computer provides access to all sensors and provides complete control of the Joy 16 CM machine. Additionally application computers, both onboard and off-board the machine, provide for navigation, diagnostics, production planning and other activities. The backbone of this system is a real-time control network called Bitbus [6]. The Bureau's implementation of Bitbus, known as BOM/NET, consists of a collection of microcontroller boards that are configured for both stand-alone operation and as gateways to other computers.

[1] Use of manufacturers names is for identification only and does not imply endorsement by the Bureau of Mines.

## 5. GENERAL NETWORK DESCRIPTION

Figure 3 shows the general layout of the network hardware as it is presently being used. The network was built to provide a method of tying together a variety of systems and subsystems so that they can communicate and interact with one another. Each system connected to the network can generally be classified as a sensor type, a control type, or a planner type. Additionally, each module is designed to be as intelligent and as self-reliant as possible. This, in effect, minimizes the amount of data that must be passed between modules, facilitating the highest level of real-time performance.

## 6. THE NETWORK OPERATING SYSTEM

A very efficient and compact operating system is used to manage all the resources of BOM/NET. It is called iDCX 51 [6] which is a real-time, multitasking operating system. Both iDCX 51 and the data link protocol are embedded in the firmware of every Bitbus card used in this network. iDCX 51 permits each node to have up to 8 concurrently-running, prioritized programs or tasks, and all can be interrupt driven.

## 7. THE NETWORK HARDWARE

Referring to figure 3, node 1 provides complete control and monitoring of the Joy 16CM mining machine. Built-in functions provides control of all of the machine appendages and control devices, as well as collection of sensor data. Node 2 is a microcontroller-based navigation system [2] that employs a gyroscope, an electronic compass, and clinometers. Node 3 is configured as a gateway to a Sun 3/60 computer. The Sun 3/60 computer is a workstation that is being used to develop a laser-based navigation system [1]. Node 4 is configured as a gateway to a Sun 3/160 computer. This workstation serves as a manual command center, a developmental platform for control programs and a graphical display device. Node 5 is configured as a gateway to a Symbolics 3670 computer that provides hydraulic system presentation using color graphics. Node 6 is configured as a gateway to a speaker-dependent voice recognition system [7]. Node 7 is a voice synthesizer [8] that provides messages for many system functions. Node 8 acts as a gateway to a specially constructed vehicle called a Locomotion Emulator (LE). The LE is a 3-wheeled, all-steer, all-drive mobile vehicle (see figure 4) which is software configurable and can emulate the motion and computer configured command set of most wheeled or tracked vehicles. Node (F) acts as the communication manager of the network.

## 8. RESEARCH SPINOFFS

Demonstration of the inherent capabilities of BOM/NET led to a request by another Bureau group for a customized remote control package for a new highwall mining system (HMS). The system was required for monitoring and remote control of a thin-seam continuous miner (TSCM) and a 76-m long multiple-unit continuous haulage (MUCH) conveying system. The MUCH system consists of 12 vehicles, each vehicle has autotracking abilities and features, integral chain conveyor, and four-wheel steering with two-wheel drive. An artist's drawing of the HMS is shown in figure 5. The HMS is controlled from a protected human-engineered operator station (HEOS) that remains outside of the coal seam being mined. The HMS features a laser-based guidance system, dual machine-mounted color TV cameras with dual HEOS-mounted video monitors, a complete remote control set for all system functions, bar graph sensor display boards, and a complete suite of TSCM-mounted sensors for machine position and diagnostics. Further details of the HMS can be found in the publications of reference [9].

## 9. A CONCEPTUAL DESIGN FOR A LUNAR MINING OPERATION

### Overview

Experience in mining and mining methods has been accumulated for many different geological environments, terrains, deposit types, rock properties, depths, climates, and other factors that affect mining operations on earth. Additionally, experience is being gained in applying automation and robotics to machinery used for mining. Application of this experience to what has been learned about the lunar surface should enable us to generate designs for harvesting lunar resources that will be required to support lunar-based, lower earth orbit, and other space missions.

The design of a lunar mining operation will be based on a diverse list of fundamental considerations including at least the following items: (1) the product; (2) the volume of production; (3) the environment; (4) the machine/design; (5) maintenance/repair; (6) power requirements/power source; (7) human requirements; (8) processes involved; and (9) production coordination. All of the above considerations warrant intense evaluation and study well beyond the scope of this paper. Therefore, the major emphasis provided here will be the coordination of the mining operation via a communications and control network. The remaining items will be briefly discussed to form the whole concept.

### The Product

Analyses of lunar rock and soil from six Apollo missions have identified mineral properties that can be used to supply many of the resources necessary to support space activities. The results have shown that the rocks and soil are rich in oxygen, silicon, and valuable metals such as iron and titanium [10]. Oxygen is the product of choice because of its use in fueling spacecraft. Oxygen makes up 6/7 of the mass of propellant utilized by cryogenic rockets [11]. Estimates vary, but on the average 1 pound of solid oxygen can be extracted from 135 pounds of lunar soil. The most predominant minerals in the lunar soil, containing oxygen are, olivine, pyroxene, and ilmenite. The reduction of these minerals to produce LOX will probably require a combination of processes like carbothermal extraction, electrostatic or magnetic separation, electrolysis, and liquification. Hydrogen, the second major ingredient of rocket fuel, could also be obtained from lunar dust, but it is very rare. Estimates show 50 to 200 ppm in the lunar regolith. Hydrogen recovery could be performed in conjunction with the production of oxygen using a thermal release process [12].

### The Volume of Production of Oxygen

The amount of oxygen ($O_2$) that can be produced will be based on many factors, most of which are unknown at this point. However, we do know that to produce 40 pounds of $O_2$ per hour, it requires the processing of 5,400 pounds per hour of lunar soil (57.5 $ft^3$). Production volume will, of course, be dictated by demand, and supply will be limited by the efficiency of the production process.

## The Environment

Information, supplied in one report [13] shows the lunar surface temperatures vary from +137° C in sunlight to -169° C at night or in a shadow, and variations can be as rapid as 174° C/min. The lunar day, as well as the lunar night, lasts for two weeks. The atmosphere on the moon consists primarily of solar wind gases such as hydrogen, helium, and neon, and is at a pressure of 10 (E-12) torr. Compared to the surface conditions, the subsurface is more stable due to the lunar regolith being such an excellent insulator. It maintains an unvarying thermal gradient. At 6 ft deep, the temperature is -17° C, at 8 ft it is -16° C, and at 100 ft it is estimated to be a constant 2° C.

## Production Machines/Design

Lunar equipment design for soil extraction, transportation, and processing will require input from many disciplines. Tribiological considerations will be major due to an atmospheric pressure of 10 (E-12) torr. At that pressure only solid lubricants can be used for bearings and frictional surfaces. Problems like vacuum adhesion [14] will require unique solutions. Less energy and forces will be required to excavate soil and rock, due to 1/6 the gravity of earth. Countering that, however, may be the requirement of increased tractive forces necessary to counterbalance excavation by using energy-robbing anchoring mechanisms. Stresses in equipment due to light/shadow effects while a machine makes a turn in the sunlight will require special attention. As an example, a straight aluminum rod, half in sunlight and half in its own shadow, if a meter long on the underside, will be approximately 1.0375 meters long on the top, resulting in an appreciable warp. Perhaps a sun umbrella could minimize the problem. Night-time mining would also be an option.

The machinery designed for lunar mining should include the highest levels of automation and robotics that will require the least amount of human intervention. It should be multifunctional, and include enough redundancy to minimize system failures. A "smart wheel" concept, which has also been referred to as a Standardized Mobility Unit (SMU) described in reference [15] seems appropriate. An operation would consist of a number of SMU's. Each would be identical in size with standardized wheels. Each wheel would feature independent steering and suspension, and each could be engaged or disengaged, i.e., driving or free-wheeling. Coupling methods for each SMU would provide for mechanical, electrical, and communication conveyance, and each could be operated locally, remotely, and autonomously. Standardized attachment points on the SMU would permit attaching devices such as manipulator arms, tools, bins, conveyors, or crew transportation pods. Power could be onboard, supplied by fuel cells, or could come from some central power distribution point through other SMU's.

### Maintenance/Repair

By design, the machinery used to produce LOX should be simple to maintain and highly reliable. Use of modular construction techniques and providing a high degree of component redundancy will facilitate both easy maintenance and could provide for continued operation even after multiple failures. As a result, the inventory of spare parts would also be minimal.

### Power Requirements and Power Source

Full-scale oxygen production of the sort being described, will have a huge power requirement. A conservative estimate would be 100 kW. Power of this magnitude can best be supplied by a nuclear source at least in the beginning phases of production. Later, solar or maybe fuel cells using lunar hydrogen and oxygen could be developed to supply power.

### Human Requirements

If a lunar mining operation becomes a reality, it will have to be supported by at least intermittent human supervision. A major concern of human occupation of the moon is the exposure to harmful radiation. The 5 rem/y exposure limit for earth-based radiation workers might be an appropriate standard for radiation protection. Reference [16] shows that adherence to this standard would make it necessary to bury a lunar habitat beneath several meters of lunar regolith and limit human activity on the lunar surface to "regular working hours", inside an enclosed vehicle. Occasional solar flares producing solar energetic particle events would also limit surface activity to within range of protective habitats. Radiation is only one human concern; numerous others will have to be addressed.

## 10. LOX PRODUCTION FACILITY

Using the cited references, combined with general mining concepts and application of a few unique ideas, a LOX production facility was conceived (figure 6). Emphasis was placed on keeping the human in a protected habitat, in a self-contained module placed in a lava tube. All mining and LOX processing would be totally automated and the human would act only as an observer with the power to prevent failures, correct improper operations, and initiate modifications in unprogrammed situations. Navigation in the pit would be based on a laser ranging technique using the corners of the pit as points of reference. The regolith extractor (RE) would be based on a couple of SMU's. The RE would follow a grid/cell production pattern as shown. The harvested regolith would be transferred by a multiple section conveyor train (MSCT) to the LOX processing plant. Refuse material would be transferred back to the pit by another MSCT. The two MSCT's would be composed of conveyor sections attached to and powered by the SMU's. Power for the RE and the MSCT's would be centrally supplied from the LOX processing plant, but backup power onboard each SMU would be provided.

## 11. PRODUCTION COORDINATION VIA A DISTRIBUTED ARCHITECTURE

The distributed architecture necessary for coordinating and controlling a LOX operation would be extensive. The system used would also have to withstand unusual environmental hazards such as temperature extremes and radiation. However, the system would not have to meet the standards required for providing life-support systems. Reliability of systems would be provided by parallel redundancy, cross-strapping,[2] and built in predictive failure techniques using expert systems. A multiple cabled network for interconnecting the SMU's would prevent a failure from interrupting communications between the other SMU's. The network would provide slave switches that would switch nodes between cables or isolate nodes out of the system. Network monitors and control computers would provide integrity through status polling, diagnostics, and control of operations. NASA has at least one patent application for such a network [17]. Each SMU in the network would contain a cluster of computers dedicated to specific tasks. The tasks would include at least a vehicle controller, position locator, obstacle avoidance, route planner, diagnostics, peripheral controller, and a communications handler. Vehicle control would be determined by onboard systems and by commands provided by an off-board production planner. The position locator could use onboard navigation sensors or could obtain its position from an off-board device. Expert systems embedded in hardware could perform continuous diagnostics and flag failures or impending failures to supervisory computers. A peripheral controller would permit control of devices attached externally to the SMU. The communications handler would provide for onboard, network or radio intercommunications.

The regolith processing plant would perform a variety of separation and oxygen extraction cycles. Control and operation would be autonomous and the same computer network used on the RE and MSCT would be used. Coordination of LOX production would primarily be provided from a central point where the human operators would oversee the process and provide input as needed.

## 12. ANALYSIS OF THE LUNAR MINING OPERATION

The described lunar mining operation was developed using the limited amount of lunar data that are available, and a small set of design considerations. Before a lunar mining operation becomes a reality, much more data must be obtained about the lunar surface. It is still unknown whether the Surveyor and Apollo sites are representative of the remainder of the lunar surface. The SMU's and peripheral devices described are elegant concepts but proof of performance must be verified in a simulated lunar environment. Many potential processes to reduce the lunar regolith to liquid oxygen have been conceived and some have been tested, but a method to identify design failings of full-scale production of oxygen employing these processes has not been established. Supervision of the production operation will undoubtably employ a distributed computer network architecture. The selection of devices used in a lunar mining operation should be based on a quantitative analysis of alternatives and priorities. Members of NASA's Automation and Technology Branch have written a computer program [18] that can provide an analysis of alternatives based on certain design criteria.

[2] A construction technique that consists of connecting redundant components in such a manner that a single component failure will not cause a module failure.

## 13. CONCLUSIONS

The Bureau of Mines has integrated a distributed processing network, that enables a diverse collection of computers and intelligent sensor systems to intercommunicate and interact over a common data path using a simple protocol. The installation of the network on a mining machine used for research has accelerated the generation of intelligent navigation and control algorithms. The installation of the network on a mining machine used for production has demonstrated its real-time control capability. Each of the cited applications have shown that the distributed processing and control network (BOM/NET) has increased the reliability and the functionality of the system to which it has been attached.

Real-time performance and intercomputer communications are a must in any type of robotic system. Robotics will no doubt command a large role in a lunar mining operation and, therefore, some type of distributed communications and control system will be employed.

## 15. REFERENCES

[1] Anderson, D. L. Position and Heading Determination of a Continuous Mining Machine Using an Angular Position Sensing System. Proceedings of the Ninth WVU International Coal Mine Electrotechnology Conference, Morgantown, WV, 1988.

[2] Sammarco, John J. Mining Machine Orientation Using Inertial, Magnetic, and Gravitational Sensors. IEEE Industry Applications Society, 23rd Annual Meeting, Pittsburgh, PA, Oct 2-7, 1988. IEEE Catalog No. 88CH2565-0.

[3] Mitchell, J. A Diagnostic Maintenance Expert System for the Hydraulic Subsystem of a Continuous Miner. Proceedings of the Ninth WVU International Coal Mine Electrotechnology Conference, Morgantown, WV, 1988.

[4] Berzonsky, B. E. A Knowledge-based Electrical Diagnostic System for Mining Machine Maintenance. Proceedings of the Ninth WVU International Coal Mine Electrotechnology Conference, Morgantown, WV, 1988.

[5] Schnakenberg, Jr., George H. U.S. Bureau of Mines Coal Mining Automation Research. Proceedings of 3rd Canadian Symposium on Mining Automation, Montreal, Canada, September 1988, pp. 145-157.

[6] Intel Corp., 5200 NE Elam Young Parkway, Hillsboro, Oregon 97234. iDCX 51 Distributed Control Executive Guide 460367-001.

[7] Kurtzweil Applied Intelligence Inc., 411 Waverley Oaks Rd., Walthamn, MA 02154. Kurtzweil Voice System.

[8] Speech Plus Inc., 640 Clyde Ct., P.O. Box 7461, Mt. View, CA 94039, Prose 2020.

[9] Kwitowski, August. Computer-Based Remote Control of A Highwall Mining System. Presented at The AusIMM ILLwarra Branch, 21st Century Higher Production Coal Mining Systems-Their Implications, Wollongong, NSW, April 1988.
Mayercheck, W. D. and R. J. Evans. Coal Extraction, Transport, and Logistics Technology for Underground Mining. U.S. Bureau of Mines IC 9181, 1988, 81 pp.

[10] Duke, Michael B., Wendell W. Mendell, and Barney B. Roberts. Strategies for a Permanent Lunar Base. Proceedings of Lunar Bases and Space Activities of the 21st Century. W. W. Mendell, Editor. National Academy of Sciences, Washington, DC., October 29-31, 1984.

[11] Rosenberg, Sanders D. A Lunar-Based Propulsion System. Proceedings of Lunar Bases and Space Activities of the 21st Century. W. W. Mendell, Editor. National Academy of Sciences, Washington, DC., October 29-31, 1984.

[12] Meek, Thomas T. Microwave Processing of Lunar Materials: Potential Applications. Proceedings of Lunar Bases and Space Activities of the 21st Century. W. W. Mendell, Editor. National Academy of Sciences, Washington, DC., October 29-31, 1984.

[13] Watson, Patricia Mendosa. Mining on the Moon. Proceedings of the Third International Conference on Innovative Mining Systems, University of Missouri, Rolla, November 2-4, 1987, pp 202-206.

[14] Ryan, J. A., and J. J. Grossman. Comments on Lunar Surface Adhesion. Proceedings of the Seventh Annual Working Group Extraterrestrial Resources, NASA SP-229, June 17-18, 1969, pp. 113-115.

[15] Harrison, F. Wallace, Nancy Sliwa, Don Soloway, and Karin Cornils. Automation and Robotics Considerations. Lunar Base Study Report. NASA Langley Research Center.

[16] Silberberg R., C. H. Tsao, J. H. Adams, Jr., and J. R. Letqw. Radiation Doses and LET Distributions of Cosmic Rays. Rad. Res., 1984, pp. 98, 209-226.

[17] NASA Resident Office Technology Utilization NPO-16949.

[18] Bard, Jonathan F. Evaluating Space Station Applications of Automation and Robotics. IEEE Trans., Eng. Managmt, vol. EM-33, No. 2, May 1986, pp. 102-111.

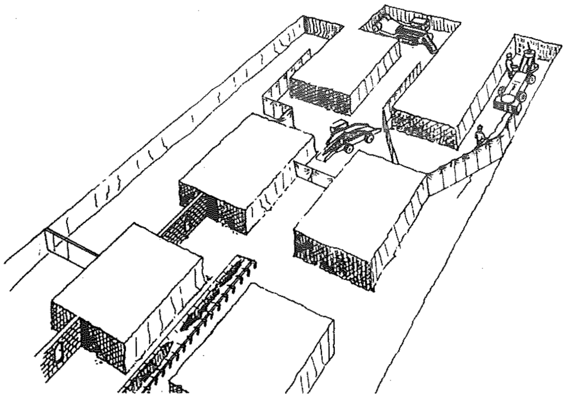


Figure 1. - Room and pillar face area.

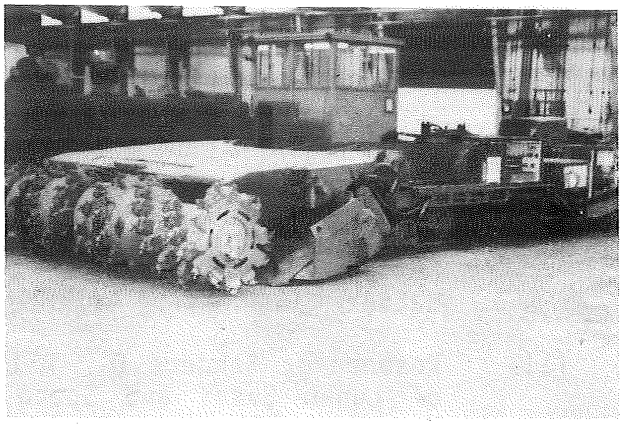


Figure 2. - Joy 16CM research area.

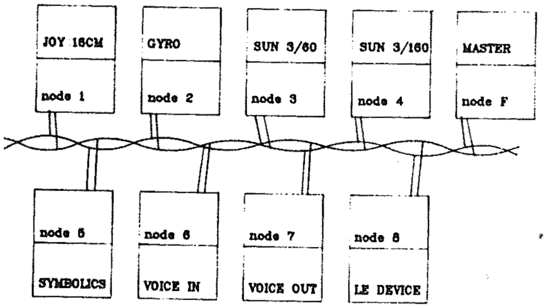


Figure 3. - Network layout.

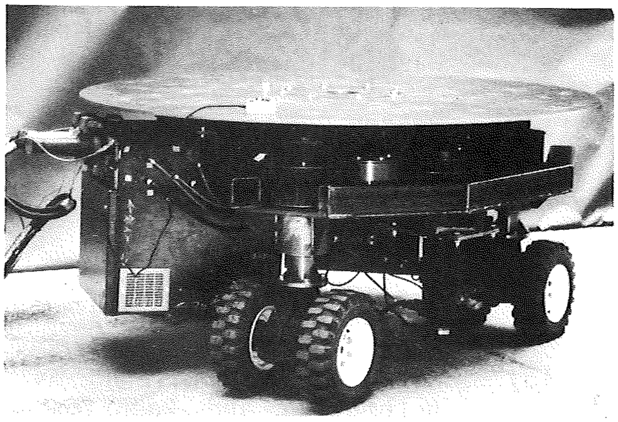


Figure 4. - Locomotion Emulator.

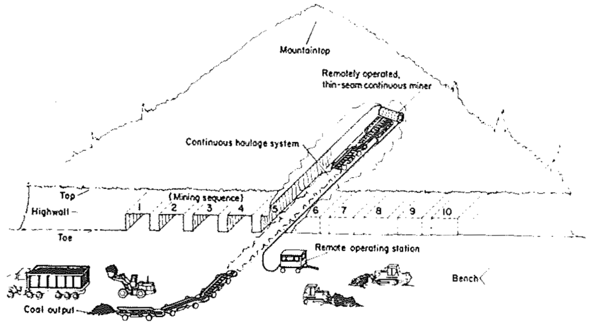


Figure 5. - Highwall mining system.

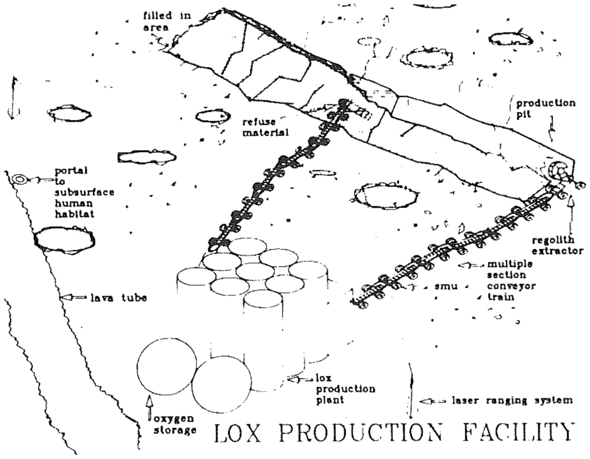


Figure 6. - Lox production facility.

# COMPUTER SIMULATION AND DESIGN OF A THREE DEGREE-OF-FREEDOM SHOULDER MODULE

D. Marco
Dept. of Mech. Engineering
Naval Postgraduate School
Monterey Ca. 93943

L. Torfason
Dept. of Mech. Engineering
Univ. of New Brunswick
Fredericton, N.B.

D. Tesar
Carol Cockrell Curran Chair in Engineering
Dept. of Mech. Engineering
Univ. of Texas at Austin
Austin, Tx.

## ABSTRACT

This paper presents an in-depth kinematic analysis of a three degree-of-freedom fully-parallel robotic shoulder module. The major goal of the analysis is to determine appropriate link dimensions which will provide a maximized workspace along with desirable input to output velocity and torque amplification. First-order kinematic influence coefficients which describe the output velocity properties in terms of actuator motions provide a means to determine suitable geometric dimensions for the device.

Through the use of computer simulation, optimal or near optimal link dimensions based on predetermined design criteria are provided for two different structural designs of the mechanism. The first uses three rotational inputs to control the output motion. The second design involves the use of four inputs, actuating any three inputs for a given position of the output link. Alternative actuator placements are examined to determine the most effective approach to control the output motion.

## 1. INTRODUCTION

Recent attention has been given to an alternative to serial robotic architecture. Designs based on closed loop kinematic chains or parallel architecture are currently under investigation. The general idea of parallel design is that the output body is controlled by a combination of mutually supportive constraints. Specifically, the structural links and input actuators all act together, directly influencing the motion of the output body. This method provides improved structural rigidity, precision positioning, and favorable load distribution [1].

The shoulder module shown in Figure 1 is a purely spherical device using three inputs. It has been conceived to precisely control three rotational degrees-of-freedom. Since the device kinematically mimics a ball-and-socket joint, no translational freedoms are present. The design has been examined and analyzed in [2,3]. Included in these references are the derivation of the first and second-order influence coefficients needed to describe the velocity and acceleration characteristics of the system, along with some preliminary optimization work.

The shoulder may be used as a component for a six degree-of-freedom force feedback joystick controller [4]. The joystick structure consists of the shoulder at the base, connected to a chain of three revolute joints with the operator gripper located at the end of the chain. Since force-feedback controllers require actuators to apply reflective forces on the operator, it is of utmost importance to design the controller to be as light as possible. This will minimize operator effort and fatigue. Locating the shoulder at the base will enable three of the actuators to be fixed to ground. Therefore, not representing any mass to the moving system. The shoulder may also serve as the base component for a combination serial/parallel manipulator [5]. In this case, locating the three shoulder drives on the ground will greatly reduce link deflections as well as inertial effects associated with purely serial architectures.

The primary objective of this work is to obtain suitable mechanism dimensions for the shoulder module based on predetermined design criteria. Link dimensions which provide uniform torque and velocity transmission from input to output are sought. Since the velocity and torque are inversely proportional at constant power, it is desired to obtain an equal balance between the two. Achieving this balance will ensure ease of actuator speed control as well as reduction of the actuator torque requirements. In most robotic systems,the region of the workspace in which the manipulator has adequate performance characteristics is considerably smaller than the overall workspace. Therefore, maximization of favorable operating regions is also desired.

Determination of the most effective actuation strategy is clearly of importance.Currently, the shoulder is conceptualized to be actuated by rotational inputs on the base as shown in Figure 1.

Section 2 briefly outlines the geometric and kinematic description of the device, with the available dimensions for design optimization. Section 3 outlines the design methodology used to establish the performance criterion. Sections 4 and 5 present the results of optimization for two different structural designs. Described in the sixth section are alternative actuation schemes which determine the best input strategies. The final section presents results and concluding remarks.

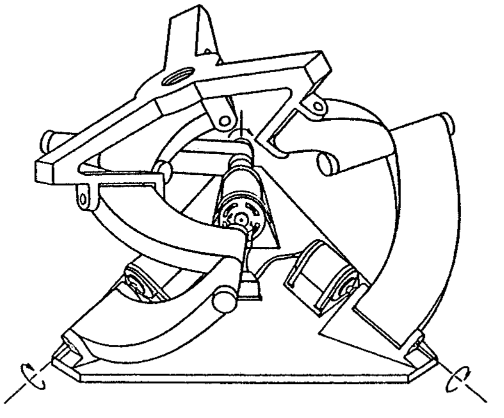


**Figure 1** The Shoulder Module

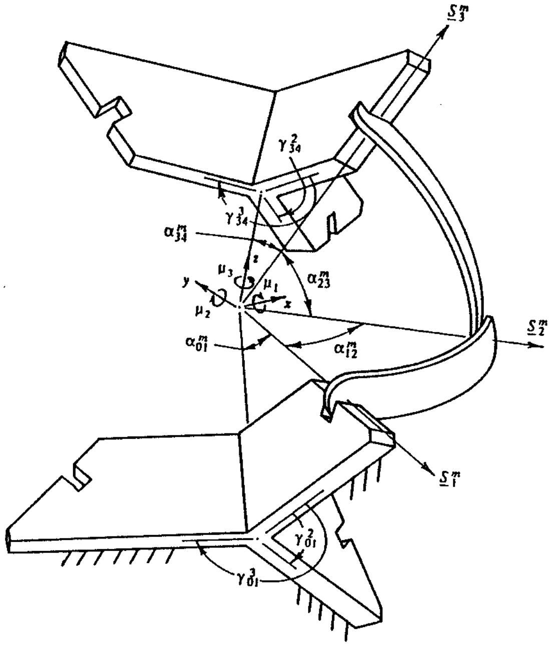


**Figure 2** Shoulder Geometric Description

## 2. SHOULDER MODULE GEOMETRIC AND KINEMATIC DESCRIPTION

The design consists of two tetrahedrons connected by three spherical binary link pairs or "dyads". Only one dyad is shown in Figure 2 for clarity. The upper tetrahedron is the moving or output link which possesses the three rotational degrees of freedom. The lower tetrahedron is the base or ground link. Each dyad chain is made up of two spherical links and three revolute joints driven by a separate actuator located on the base joint.

In order to produce purely spherical motion, all nine joint axes must intersect at a common point. For each dyad, the revolute joint axes directions are denoted by $\underline{S}_n^m$. The superscript, $m$ , corresponds to the subchain or dyad being described, while the subscript, $n$ , denotes a particular joint in the subchain. There are three subchains and three joints per chain.

To uniquely define the orientation of the output link, successive rotations about the respective x, y, z body fixed axes of the output link are performed. The origin of both the global and moving (body fixed) coordinate systems are located at the intersection of the nine joint axes. Initially, both the global and moving coordinate systems are coincident with each other . Then the output link is first rotated about the x-axis to define rotation angle $\mu_1$. This is followed by a rotation $\mu_2$ about the resulting y-axis, and finally a rotation $\mu_3$ about the resulting z-axis. At the workspace center, $\mu_1 = \mu_2 = \mu_3 = 0$. This orientation will be refered to as the reference position.

The twist angles of the dyad links $\alpha_{12}^m$ and $\alpha_{23}^m$ are the fixed angles between joints $\underline{S}_1^m$ and $\underline{S}_2^m$, and $\underline{S}_2^m$ and $\underline{S}_3^m$ respectively. Along with the twist angles are the apex angles. These angles describe the geometry of the upper (output link) and the lower (ground link) tetrahedrons given by $\alpha_{34}^m$ and $\alpha_{01}^m$. The second set of angles needed to describe the links are the edge displacement angles, $\gamma_{34}^m$ and $\gamma_{01}^m$. These angles define the relative angular position of the edge planes of the links as shown.

In general there are sixteen parameters required to completely describe the geometry of the shoulder, six twist, six apex, and four non-zero edge displacement angles. They make up the complete set of variables available for optimization. If symmetry of the device is exploited, the sixteen required parameters reduce to only four. In this case, all dyads are identical and each tetrahedron is symmetrical (i.e. each edge plane is spaced 120° apart).

The relationship between the input actuator speeds and the output ternary velocity is [2]:

$$\underline{\dot{u}} = [G_{\phi}^{u}]\underline{\dot{\phi}} \tag{1}$$

where $\underline{\dot{\phi}}$ is the vector of input actuator speeds which is the time derivative of each input angular displacement, and $\underline{\dot{u}}$ is the output link angular velocity expressed in a global reference. $[G_{\phi}^{u}]$ is the influence coefficient matrix or Jacobian where the superscript, $u$, refers to the dependent variable, while the subscript, $\phi$, denotes the independent variable [6,7]. The Jacobian is defined by

$$[G_{\phi}^{u}] = \frac{\partial \underline{\dot{u}}}{\partial \underline{\dot{\phi}}} = \begin{bmatrix} \frac{\partial \underline{\dot{u}}}{\partial \dot{\phi}_1} & \frac{\partial \underline{\dot{u}}}{\partial \dot{\phi}_2} & \frac{\partial \underline{\dot{u}}}{\partial \dot{\phi}_3} \end{bmatrix} \tag{2}$$

where the $i^{th}$ component of Equation (2) is

$$[G_{\phi_i}^{u}] = \begin{bmatrix} \partial \dot{u}_x / \partial \dot{\phi}_i \\ \partial \dot{u}_y / \partial \dot{\phi}_i \\ \partial \dot{u}_z / \partial \dot{\phi}_i \end{bmatrix} \tag{3}$$

The x, y, and z subscripts in Equation (3) refer to the x, y, and z components of $\underline{\dot{u}}$ expressed in a global reference.

The Jacobian, $[G_{\phi}^{u}]$, can be used to obtain the input torque requirement, $\underline{T}_{\phi}$, for a given applied load state, $\underline{T}^{u}$, using a virtual work formulation. $\underline{T}_{\phi}$ is a three component vector of the input actuator torques along the $\underline{S}_1^1$, $\underline{S}_1^2$, and $\underline{S}_1^3$ axes. These are the torques required to counteract the applied loads on the output link to maintain the system in static equilibrium. For static equilibrium, the virtual work, $\delta W$, must be zero, therefore

$$\delta W = \underline{T}_{\phi} \cdot \delta\underline{\phi} + \underline{T}^{u} \cdot \delta\underline{u} = 0 \tag{4}$$

where the virtual output displacements, $\delta\underline{u}$, must conform to the constraints of the system. From Equation (1)

$$\delta\underline{u} = [G_{\phi}^{u}]\delta\underline{\phi} \tag{5}$$

since,

$$\begin{aligned} \delta\underline{u} &= \underline{\dot{u}}\,\delta t \\ \delta\underline{\phi} &= \underline{\dot{\phi}}\,\delta t \end{aligned} \tag{6,7}$$

Subtituting Equation (5) into Equation (4), the following result is obtained

$$\underline{T}_{\phi} = -[G^{u}_{\phi}]^{T}\underline{T}^{u} = -\underline{T}^{L}_{\phi} \tag{8}$$

where $\underline{T}^{L}_{\phi}$ is defined as the effective load at the input $\underline{\phi}$ due to loads applied to the output link.

## 3. SHOULDER DESIGN METHODOLOGY

In order to establish the required design method based on the input and output velocity relationship, the Rayleigh quotient is used . For a symmetric matrix [A], the quotient is stated as

$$R\,(\underline{x}) = \frac{x^{T}[A]x}{\underline{x}^{T}\underline{x}} \tag{9}$$

The quotient $R\,(\underline{x})$ is minimized by the eigenvector, $\underline{x}_{\min}$, corresponding to the minimum eigenvalue, $\lambda_{\min}$, of [A]. Similarily, $R\,(\underline{x})$ is maximized by the eigenvector, $\underline{x}_{\max}$, associated with the maximum eigenvalue, $\lambda_{\max}$, of [A]. This places and upper and lower bound on $R\,(\underline{x})$ by

$$\frac{\underline{x}^{T}_{\max}[A]\underline{x}_{\max}}{\underline{x}^{T}_{\max}\ \underline{x}_{\max}} = \frac{\underline{x}^{T}_{\max}\ \lambda_{\min}\ \underline{x}_{\max}}{\underline{x}^{T}_{\max}\ \underline{x}_{\max}} = \lambda_{\max} \tag{10}$$

$$\frac{\underline{x}^{T}_{\min}[A]\underline{x}_{\min}}{\underline{x}^{T}_{\min}\ \underline{x}_{\min}} = \frac{\underline{x}^{T}_{\min}\ \lambda_{\min}\ \underline{x}_{\min}}{\underline{x}^{T}_{\min}\ \underline{x}_{\min}} = \lambda_{\min} \tag{11}$$

Therefore, for any vector $\underline{x}$

$$\lambda_{\min} \leq \frac{x^{T}[A]x}{\underline{x}^{T}\underline{x}} \leq \lambda_{\max} \tag{12}$$

Recalling Equation (1) it is desired to obtain an upper and lower bound for the output velocity, $\dot{\underline{u}}$, and the input speeds, $\dot{\underline{\phi}}$. . In order to achieve this, the ratio of the output to the input velocity magnitude is expessed as

$$\frac{\dot{\underline{u}}}{\dot{\underline{\phi}}} = \left[\frac{\dot{\underline{u}}^{T}\dot{\underline{u}}}{\dot{\underline{\phi}}^{T}\dot{\underline{\phi}}}\right]^{\frac{1}{2}} \tag{13}$$

and substituting Equation (1) into Equation (13) yields

$$\frac{\dot{\underline{u}}}{\dot{\underline{\phi}}} = \left[\frac{\dot{\underline{\phi}}^{T}[G^{u}_{\phi}]^{T}[G^{u}_{\phi}]\dot{\underline{\phi}}}{\dot{\underline{\phi}}^{T}\dot{\underline{\phi}}}\right]^{\frac{1}{2}} \tag{14}$$

Equation (12) may now be written as

$$(\Lambda_{\min})^{\frac{1}{2}} \leq \left[\frac{\dot{\underline{\phi}}^{T}[G^{u}_{\phi}]^{T}[G^{u}_{\phi}]\dot{\underline{\phi}}}{\dot{\underline{\phi}}^{T}\dot{\underline{\phi}}}\right]^{\frac{1}{2}} \leq (\Lambda_{\max})^{\frac{1}{2}} \tag{15}$$

where $\Lambda_{\max}$ and $\Lambda_{\min}$ are the maximum and minimun eigenvalues of $[G^{u}_{\phi}]^{T}[G^{u}_{\phi}]$. Since the performance of $\dot{\underline{u}}$ relative to $\dot{\underline{\phi}}$ is of interest, Equation (1) is back substituted into Equation (15) to obtain

$$|\dot{\underline{\phi}}|(\Lambda_{\min})^{\frac{1}{2}} \le |\dot{\underline{u}}| \le (\Lambda_{\max})^{\frac{1}{2}} |\dot{\underline{\phi}}| \tag{16}$$

Defining

$$\lambda_{\max} = (\Lambda_{\max})^{\frac{1}{2}}$$
$$\lambda_{\min} = (\Lambda_{\min})^{\frac{1}{2}} \tag{17,18}$$

Equation (16) becomes

$$|\dot{\underline{\phi}}|\,\lambda_{\min} \le |\dot{\underline{u}}| \le \lambda_{\max}\,|\dot{\underline{\phi}}| \tag{19}$$

This formulation may also be used to obtain the extremes of the torque values. Recalling Equation (8), and applying the same procedure as before with $[G^u_\phi][G^u_\phi]^T$, the magnitude of the input and output torques are related by

$$|\underline{T}^u|\,\lambda_{\min} \le |\underline{T}_\phi| \le \lambda_{\max}\,|\underline{T}^u| \tag{20}$$

Equations (19) and (20) now provide a means to analyze the torque and velocity characteristics of the shoulder. These characteristics are described by the relative values of the maximum and minimum eigenvalues. As seen from Equation (19), large eigenvalues indicate a large velocity amplification from the input actuators to the output link. On the other hand, small eigenvalues indicate a small amplification. In contrast to the above relationship, the torque amplification is inversely proportional to the velocity amplification. This is analogous to mechanical advantage. Refering to Equation (20), large eigenvalues indicate a low torque transmission from the input to the output, and small eigenvalues signify high torque transmission.

From a practical point of view, the designer would like to have high torque transmission from input to output, for this increases payload and reduces actuator size. This leads to the assumption that small system eigenvalues are desired. However,this will sacrifice the speed of response of the device, since small eigenvalues signify a small velocity amplification. This also implies a small range of motion. Due to these characteristics, a give and take situation exists between the load capacity and operating speed. In order to have balance between the two, eigenvalues of one are desired. This is an ideal case and since the shoulder mechanism is very non-linear, it can not be achieved throughout the entire workspace.

A second justification for unity eigenvalues concerns the relative separation of the maximum and minimum eigenvalues. Both in general will not be relatively large or relatively small. The larger the separation between $\lambda_{\min}$ and $\lambda_{\max}$, the larger the separation between the upper and lower bounds of the velocity and torque relationships. Refering to Equation (19), the magnitude of the output velocity will be at its upper limit if the input velocity vector lies along $\underline{x}_{\max}$ of $[G^u_\phi]^T[G^u_\phi]$, and at its lower limit with this vector along $\underline{x}_{\min}$. Similar characteristics are observed by the applied torque vector in Equation (20). When this vector lies along $\underline{x}_{\max}$ of $[G^u_\phi][G^u_\phi]^T$, the required input torque is at a maximum. If the output torque vector lies along $\underline{x}_{\min}$, the opposite action takes place and high torque transmission is present from the actuators to the output. This leads to a second basic design criteria: It is desired that the variance between the maximum and minimum eigenvalues be kept to a minimum.

## 4. DESIGN OF THE THREE DYAD SHOULDER

The maximum and minimum eigenvalues are evaluated for given output link positions throughout the workspace. In doing this, an insight into the desirable mechanism dimensions are obtained. The Jacobian is a function of the fixed geometric parameters along with the position of the output link. This suggests that by varying the fixed dimensions, (i.e. twist, apex, and edge displacement angles), the input to output relationship can be tuned to provide suitable performance throughout the workspace. However, since the Jacobian is also a function of the output position, the input output relationship is not constant and highly non-linear, and only certain regions of the workspace will be suitable for operation.

In order to compactly represent the locations and magnitudes of the maximum and minimum eigenvalues, contour plots are generated to determine the regions of desirable operation. The eigenvalues are evaluated over an operating range of $\mu_1$ , $\mu_2$ , $\mu_3$ , each from -90° to +90°. Since the three independent parameters must be represented on paper, two of the variables, $\mu_1$ and $\mu_2$ are evaluated over the -90° to +90° range with $\mu_3$ held constant . $\mu_3$ is then incremented from -90° to +90° to give an overall view of the operating range. Once the plots for each constant value of $\mu_3$ are generated, they are overlayed, and the contour lines common to all of the plots (i.e. throughout the range of $\mu_3$ ) are traced out .

After an extensive trial-and-error search for suitable link dimensions, including non-symmetric designs, near optimal values were isolated. These values determined that the design should be symmetrical. Contour plots for two representative geometries are shown in Figures 3 and 4. Comparing the plots for each case it is observed that no region of unity eigenvalues exist. This condition is approched at the workspace center (reference position), but is never attained. It is also evident that the variance between the maximum and minimum eigenvalues increases outward from the workspace center.

The eigenvalue behavior for a geometry with apex angles of 45° and twist angles of 90° is shown in Figure 3. This design provides a large, smooth region of maximum eigenvalues along with a small gradient present in the minimum eigenvalue plot. However, there are two areas of high maximum eigenvalue gradient in locations $\mu_1$ = -36° with $\mu_2$ = 45° and -45°. In these regions, the dyad links are becoming fully extended approaching a stationary configuration. In an attempt to remove the full extension, thereby increasing the mechanical advantage, the dyad twist angles were increased from 90° to 100° and then to 120°. This adjustment did improve the behavior in the regions described, but an overall degradation of performance was observed.

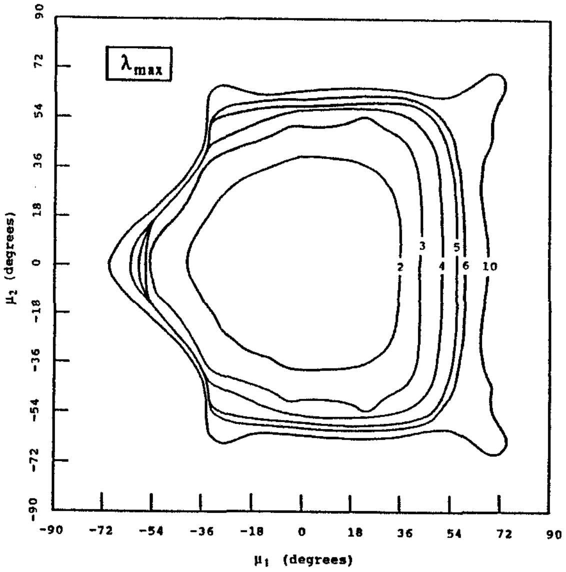


**Figure 3** Maximum and Minimum Eigenvalues

Figure 4 depicts the eigenvalue behavior for apex angles of 50° and twist angles of 90°. This design has been selected as having the best overall performance characteristics. A very large, smooth area of maximum as well as minimum eigenvalues is present. The only detraction from this case is again some evidence of full extension of the links. For this set of dimensions, the extension occurs in the areas of $\mu_1$ = -50° with $\mu_2$ = 45° and -45°. This is not as severe as in the previous case, and since these regions occupy a small percentage of the workspace, they may be avoided during operation.

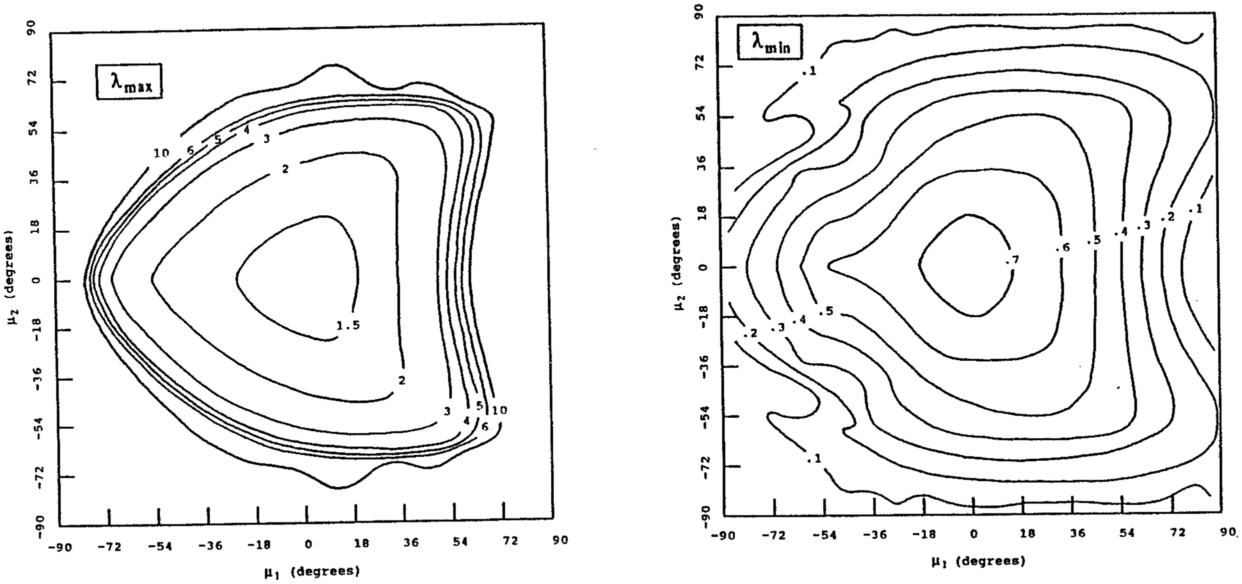


**Figure 4** Maximum and Minimum Eigenvalues

## 5. DESIGN OF THE FOUR DYAD SHOULDER

The results of the previous section have shown the performance of the shoulder is dependent on the fixed geometric parameters. Once these values have been selected, the performance characteristics for a particular set of dimensions are invariant. It would be desirable to be able to change the mechanism dimensions during operation to adapt to the non-linear behavior of the system. Since this is not practical, a design utilizing four inputs has investigated. This design is identical to the three dyad shoulder except an additional actuated dyad has been included connecting the base link to the output link. Since the fourth dyad is geometrically identical to the other three, and its revolute joint axes also intersect at a common point, the spherical output motion will not differ from the three dyad design.

The four dyad design is shown in Figure 5. Each leg of the base and output link is spaced at 90° apart to maintain symmetry. All four dyads will not be actuated simultaneously. Since three inputs are required to position and control the output , only three of the inputs will be active at any one time. The three of four inputs which best fulfill the design criteria will be actuated to control the output motion. The fourth dyad actuator will remain inactive.

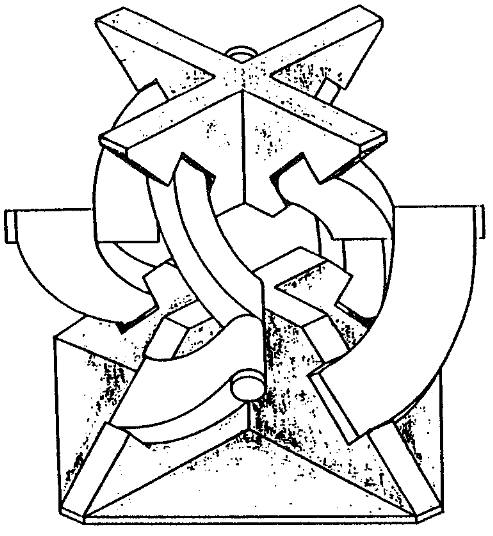


**Figure 5** The Four Dyad Shoulder

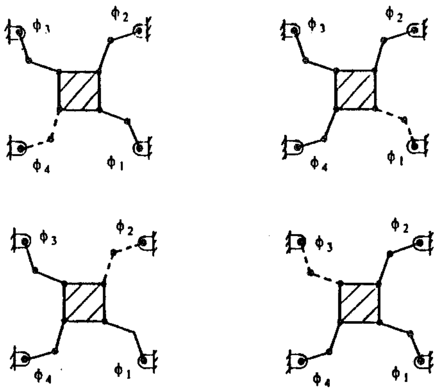


**Figure 6** Actuation Strategy

For ease of graphical representation, the planar equivalent of the device, shown in Figure 6, will be used to describe the selection of the best combination of three inputs to actuate the system. Since only three out of four can be used at any one time, the combination yielding a maximum eigenvalue nearest unity was selected. This choice is one of many, and, minimum eigenvalues could have been used. However, this makes the selection more difficult because their characteristics are more variable.

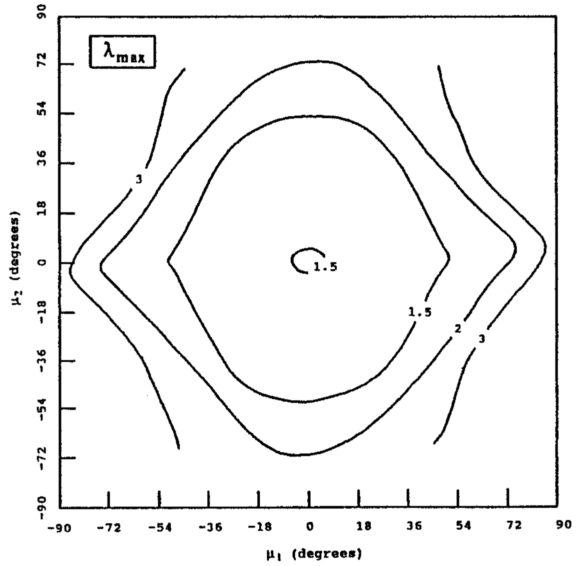


**Figure 7** Maximum and Minimum Eigenvalues

The eigenvalue behavior with apex angles of 45° and twist angles of 90° is shown in Figure 7. A region of very good maximum eigenvalues exist with a smooth gradient outward from the center of the workspace. The minimum eigenvalue results were not as promising, but are relatively smooth. Strong symmetry is evident about the workspace center. This is expected due to the ability to alternate between four geometric arrangements during operation.

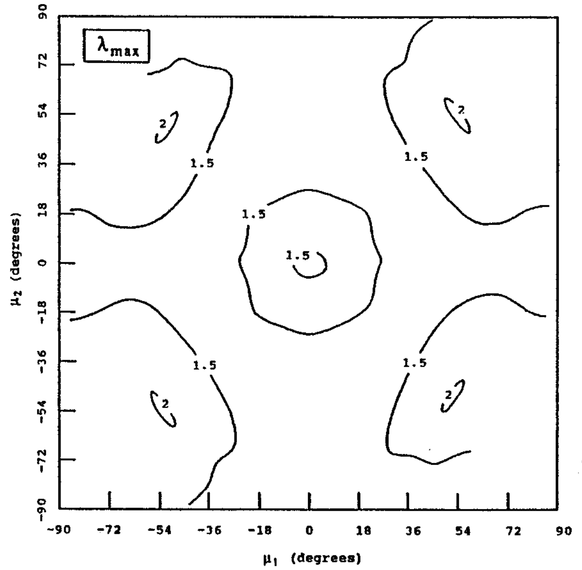


**Figure 8** Maximum and Minimum Eigenvalues

Exceptional results are obtained for apex angles of 50° and twist angles of 90° shown in Figure 8. A near constant value of 1.5 is present throughout the workspace. The corresponding minimum eigenvalues also behave

ORIGINAL PAGE IS
OF POOR QUALITY

remarkably well. Like the previous case, an increasing variance of the maximum and minimum eigenvalues is observed near the edges of the workspace. However, for this case the variance is relatively small. Therefore, the set of optimal mechanism dimensions selected for the three dyad shoulder has also been selected for the four dyad design.

## 6. ALTERNATIVE ACTUATION LOCATIONS

The method of transfer of generalized coordinates [6,7], has been applied to obtain the kinematic model relating the various inputs to the output parameters. The derivation of the kinematic model for actuating the device from any three of the joint axes is outlined in Appendix A. Various input locations were investigated, but superior results to base actuation were not found.

## 7. CONCLUSIONS

The analysis, simulation and geometric design of the shoulder has been presented. With the first order kinematic influence coefficients defined, a useful design tool was developed. The magnitudes of the system eigenvalues were shown to provide upper and lower bounds on the input to output velocity and torque characteristics. This prompted the search for mechanism dimensions which would provide system eigenvalues near one throughout the workspace.

Illustrative examples of maximum and minimum eigenvalue contour plots have been presented. These plots provided a good graphical representation of the system performance for various sets of mechanism dimensions. The following results have been determined for the three dyad shoulder.

Use symmetric geometries; edge displacement angles of 0°,
120°, 240° for both input and output links.

Apex angles of 50° for both links and dyad twist angles of 90°
for all three dyad chains

The workspace should be limited to output orientation
angles of $\mu_1$, $\mu_2 = \pm 60^\circ$; $\mu_3 = \pm 90^\circ$.

An analysis of a four dyad shoulder was also presented. This alternative structural design proved to be better than the three dyad system. The most suitable apex and dyad twist angles were found to be identical to those of the optimal three dyad shoulder. Any three of the four available dyad chains were actuated at any one time, thus enabling the three best performing dyads to control the output for a particular position. Regions of maximum eigenvalues near one were observed throughout the workspace.

Various input locations were investigated, but superior results to base actuation were not found.

## REFERENCES

1 Hunt, K. H., "Structural Kinematics of In-Parallel-Actuated Robot Arms", *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 105, 1983, pp. 705-712.

2 Cox, D. J., and Tesar, D., "The Dynamic Modeling and Command Signal Formulation for Parallel Multi-Parameter Robotic Devices", Masters Thesis, Univ. of Florida, Gainsville, FL, 1981.

3 Torfason, L., "Design of the Florida Shoulder", Internal Report, Univ. of Florida, Gainsville, FL, 1983.

4 Bryfogle, M., and Tesar, D., "The Design of a Universal Spatial Seven Degree-of-Freedom Manual Controller for Teleoperator Systems", Masters Thesis, Univ. of Florida, Gainsville, FL, 1981.

5 Sklar, M., and Tesar, D., "Dynamic Analysis of Hybrid Serial Manipulator Systems Containing Parallel Modules", *ASME Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 110, 1988, pp. 109-115.

6 Freeman, R. A., and Tesar, D., "Dynamic Modeling of Serial and Parallel Mechanisms/Robotic Systems, Part 1-Methodology", *Trends and Developments in Mechanisms, Machines, and Robotics*, DE-Vol. 15-3, 1988, pp. 7-27.

7 Freeman, R. A., and Tesar, D., "Dynamic Modeling of Serial and Parallel Mechanisms/Robotic Systems, Part 2-Applications", ibid.

8 Thomas, M., and Tesar, D., "Dynamic Modeling of Serial Manipulator Arms", *ASME Journal of Dynamic Systems, Measurement and Control*", Vol. 104, No. 3, 1982, pp. 218-228.

## APPENDIX A TRANSFER OF GENERALIZED COORDINATES

The kinematic relationship between the output link velocity, $\dot{\underline{u}}$, and any set of generalized coordinates, $\dot{\underline{q}}$, is given by

$$\dot{\underline{u}} = [G^u_q]\dot{\underline{q}} \tag{A1}$$

The first step to obtain the influence coefficients is to model each of the three dyad chains as a separate three degree-of-freedom serial manipulator. Each manipulator consists of an upper and lower dyad link and the output link. The output link is the common output to the three manipulators. It can be shown that the Jacobian for each dyad chain, $m$, is given by [8]

$$[{}_mG^u_\phi] = [\ \underline{S}^m_1 \quad \underline{S}^m_2 \quad \underline{S}^m_3\ ] \tag{A2}$$

The velocity relationship for each chain will be

$$\begin{aligned} \dot{\underline{u}} &= [{}_1G^u_\phi]{}_1\dot{\underline{\phi}} \\ \dot{\underline{u}} &= [{}_2G^u_\phi]{}_2\dot{\underline{\phi}} \\ \dot{\underline{u}} &= [{}_3G^u_\phi]{}_3\dot{\underline{\phi}} \end{aligned} \tag{A3}$$

$$\begin{aligned} {}_1\dot{\underline{\phi}} &= (\ {}_1\dot{\phi}_1\ ,\ {}_1\dot{\phi}_2\ ,\ {}_1\dot{\phi}_3\ ) \\ {}_2\dot{\underline{\phi}} &= (\ {}_2\dot{\phi}_1\ ,\ {}_2\dot{\phi}_2\ ,\ {}_2\dot{\phi}_3\ ) \\ {}_3\dot{\underline{\phi}} &= (\ {}_3\dot{\phi}_1\ ,\ {}_3\dot{\phi}_2\ ,\ {}_3\dot{\phi}_3\ ) \end{aligned} \tag{A4}$$

In order to obtain the link output velocity, $\dot{\underline{u}}$, in terms of any three of the nine input velocities, Equations (A3) must be solved for ${}_m\dot{\underline{\phi}}$ by

$${}^m\dot{\underline{\phi}} = [{}_mG^u_\phi]^{-1}\dot{\underline{u}} = [{}^mG^\phi_u]\dot{\underline{u}} \tag{A5}$$

The independent variable $\dot{\underline{u}}$ is common to all three equations. One may select any three particular rows from the three inverse Jacobians in Equations (A5). This will form a fourth, composite Jacobian which may be generalized to accommodate any three inputs by the following

$$\dot{\underline{q}} = \begin{Bmatrix} {}^m\dot{\phi}_n \\ {}^m\dot{\phi}_n \\ {}^m\dot{\phi}_n \end{Bmatrix} = \begin{bmatrix} [{}^mG^\phi_u]_{n;} \\ [{}^mG^\phi_u]_{n;} \\ [{}^mG^\phi_u]_{n;} \end{bmatrix}\dot{\underline{u}} \qquad \begin{aligned} m &= 1,2,3 \\ n &= 1,2,3 \end{aligned} \tag{A6}$$

where $m$ refers to any leg and $n$ to any input. The symbol $n;$ refers to the $n^{th}$ row of the matrix. To obtain the desired input to output relationship, Equation (A6) must be inverted as follows

$$\dot{\underline{u}} = [G^q_u]^{-1}\dot{\underline{q}} = [G^u_q]\dot{\underline{q}} \tag{A7}$$

# A COLLISION AVOIDANCE SYSTEM FOR A SPACEPLANE MANIPULATOR ARM

A. SCIOMACHEN, P.G. MAGNANI

Tecnospazio S.p.A.
Via Mercantesse,3 - 20021 Baranzate di Bollate
MILANO (ITALY)

## Abstract

**The aim of this paper is to report on part of the activity performed by Tecnospazio in the area of collision avoidance related to the Hermes spaceplane. In particular, a collision avoidance system which has been defined, developed and implemented in this project is presented.**

## 1. Introduction

The collision avoidance capability is a necessary feature for good safety performances of both automatic and teleoperated space robotic systems [1,2].

In **Figure 1** the Hermes spaceplane currently under development at ESA is represented [3,4]. The spaceplane, similarly to the US-STS, carries a manipulator arm presently designed with six degrees of freedom, two at the shoulder, one at the elbow and three at the EE, respectively. The arm is about 10 m. in lenght, presents a non negligible flexibility and can operate both automatically, through programmed sequences, and under direct human teleoperation.

It must anyway be noted that during the teleoperation the astronaut has not full visibility of the arm/obstacle relative position, while during automatic mode a protective "software mask" is advisable in order to prevent any system failure. It is therefore essential to have an anticollision capability flexible enough to operate off-line (during task planning), on-line (during task execution) and to allow the operator easy environment modifications.

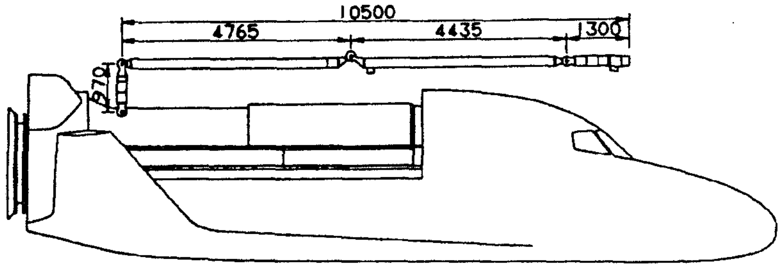


Fig. 1 - HERMES Spaceplane

The proposed collision avoidance software approach is presently under development/preliminary implementation in the industrial world. Nevertheless,the techniques and the concepts that are considered are already widely used in specific applications, e.g. CAD/CAM, multiarm robots. Therefore the approach used, even if advanced in its conceptions, has been kept as practical as possible, avoiding the use of techniques not yet established.

## 2. Mission Requirements for a Collision Avoidance System

In this scenario, a collision avoidance system is a software tool whose task is to detect a status of collision which may occur between the Hermes spaceplane, the objects hardcoupled with Hermes, the manipulator arm and a payload attached to the manipulator arm.

A status of collision must be verified through the execution of the related algorithms both under real conditions and under simulation. In the first case HERA is moving and the system is in operating mode; otherwise, no HERA movement is involved.

The on-line system is active both in automatic mode and in operator controlled mode/single joint mode. Automatic mode means execution and verification of preprogrammed trajectories under automatic control. In this case the system asks for the operator support any time a collision is detected. In the other mode, operator controlled movements are executed and verified; the operator controls directly the arm through a suitable device, for instance a joystick. The single joint mode is activated in case of failures of the control system in order to retract HERA to its stowage position moving one joint at a time and in operations in which a fully stretched arm is required.

The HERA programming mode takes place off-line and is executed on ground, by an operator. If a collision is detected, the operator must modify the current trajectory using the information provided by the system.

The inputs to the system are a set of joint coordinates representing different configurations of the arm during its movement along a planned trajectory, and the stopping distance value, that is the distance required to stop the arm. Starting from these information, the system verifies whether a collision is possible or not. The following four types of collision have to be verified:

- link-obstacle
- link-link
- payload-obstacle
- payload-link.

Due to the geometric characteristics of the objects likely to be involved in a collision, each of these types of check is performed sequentially by a separated algorithm.

Observing **Figures 1 and 2**, representing respectively the Hermes and the HERA referring configuration, some considerations about the workspace can be derived. In particular, no collision is possible between the first link and any obstacle, between contiguous links and between the payload and the last three links. No restrictions exist as far as a possible collision between the payload and any obstacle is concerned: the payload, in fact, can collide with an obstacle in any position reachable by the arm.

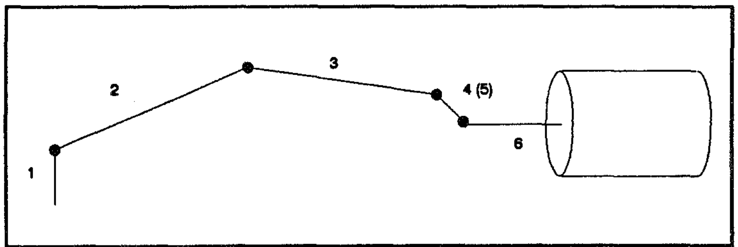


Fig. 2 - HERA Referring Configuration

The collision status (yes/not), together with other suitable information for a path planning and a validation support, are provided by the system in order to allow a timely and efficient intervention. The outputs are sent both to an user-interface module as video messages and audio warnings and to the manipulator controller. In particular, when a collision is foreseen, the following information are given:

- a collision warning
- the objects involved in the detected collision, i.e. arm, obstacle, payload
- the colliding link, the face of the obstacle and the intersection point, in case of collision of the arm with an obstacle
- the two involved links and the collision point, in case of collision of the arm with itself
- the face of the obstacle intersecting the payload, in case of collision of the payload with an obstacle
- the colliding link, in case of collision of the arm with the payload.

A single operator interface must manage all the warnings and the information needed. The above information are not presented directly by the collision avoidance system but are returned to the calling program that must send them to the operator interface of the overall system. It has to be observed that, apart the collision detection, all the operations required in the recovery phase, e.g. switching between operating mode, are not in charge of the collision avoidance software. The collision avoidance routines send information to the system that performes the necessary actions.

It is required that the collision avoidance system be effective keeping at the minimum the false warnings. A reasonable upper bound of the resolution of the system is the stopping distance: this means that the sum of the margins used to take into account discretizations, uncertainties, tolerances and approximations in the geometrical models cannot be greater than the maximum stopping distance.

The resource availability for the collision avoidance system is very limited: the overall system must run, at most, at a rate of 18 kflops. This value includes the time required to check the conformity between the outer world and the model contained in the database. Moreover, the memory available on board is about 40 kbytes.

## 3. The Proposed Collision Avoidance Method

The proposed collision avoidance software system [5,6] computes the intersection between the solids representing the arm, the payload and the objects. Only the approximated geometrical models of any kind of objects, increased by the stopping distance value, are considered. So two objects are defined to be in collision if and only if their relative distance is less than the stopping distance.

The computation of the stopping distance, which is a function of the mass and velocity of the arm and of the payload, and of the brake feature, is performed by an external module.

Information about the arm position and the outer world are given through geometric models and the related data are contained in specific databases. The obstacle database contains the geometrical description of all the fixed objects, i.e. Hermes parts and other obstacles, if any. The basic element is a convex polyhedron: non convex polyhedra can be modelled as the union of a finite number of convex polyhedra. For each object contained in the database the equations of the faces and the coordinates of the edges, defined with respect to the global reference frame, are stored.

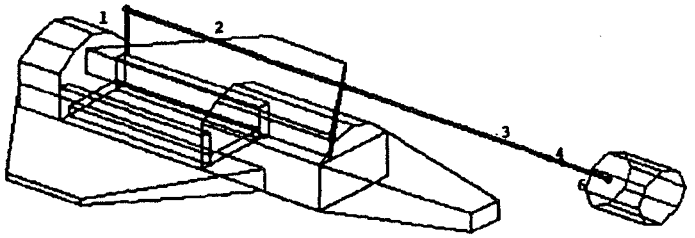


Fig. 3 - The HERMES Model

The Hermes model, depicted in **Figure** 3, is composed of 8 convex polyhedra, 54 faces and 76 vertices. In particular, it is composed of:

| | | |
|---|---|---|
| • tail | : | a prism with 7 base sides; |
| • cargo bay | : | 3 prisms with 4 base sides each; |
| • left wing | : | a generic convex polyhedron; |
| • right wing | : | a generic convex polyhedron; |
| • cockpit | : | a prism with 7 base sides; |
| • forward fuselage | : | a pyramid with 4 base sides. |

As the shape of the links is about cylindrical, the geometry of a link is defined giving the length of the axis and the radius. The payload is modelled as a cylinder.

Starting from the data contained in the world model database, the collision avoidance tasks are assigned both to a geometric method and to a forbidden values method. A possible combined implementation of the two methods can be used; in this approach, the forbidden values method checks the collision between the first three links and any obstacle, while the geometric method is used for all the other types of collision.

As it has been already said, in the collision detection phase four main modules are used in the collision avoidance system, namely:

- link-obstacle module
- link-link module
- payload-obstacle module
- payload-link module.

Before the execution of the geometric method can start, an initialization procedure is required in order to compute the coordinates of the geometric baricenter of the obstacles contained in the database and the radia of the spheres circumscribing the obstacles. As it is herebelow described, these data are required in the preliminary collision tests.

A basic step of any collision avoidance module is the direct kinematic computation. From joint coordinates the cartesian positions of the two extremes of each link are obtained using the Denavitt-Hartemberg matrices [7].

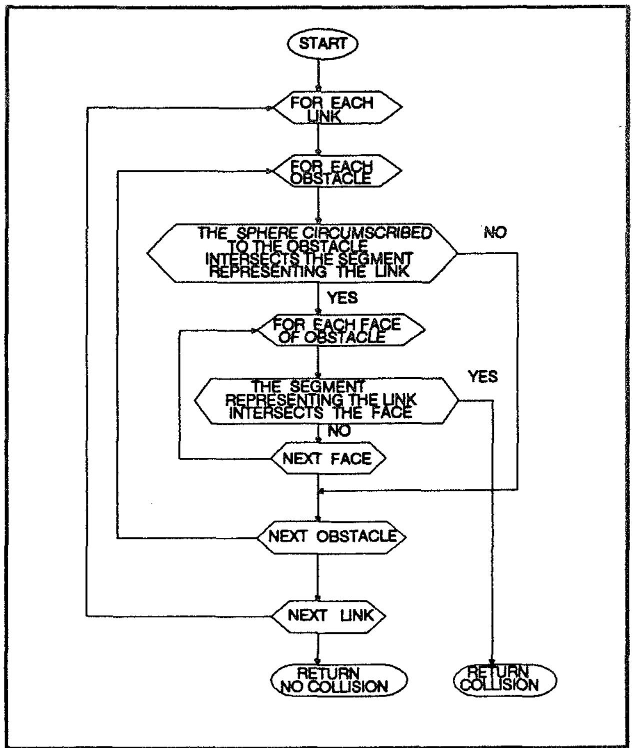

START
FOR EACH LINK
FOR EACH OBSTACLE
THE SPHERE CIRCUMSCRIBED TO THE OBSTACLE INTERSECTS THE SEGMENT REPRESENTING THE LINK
NO
YES
FOR EACH FACE OF OBSTACLE
THE SEGMENT REPRESENTING THE LINK INTERSECTS THE FACE
YES
NO
NEXT FACE
NEXT OBSTACLE
NEXT LINK
RETURN NO COLLISION
RETURN COLLISION

Fig. 5 - Link Obstacle Module Flow-Chart

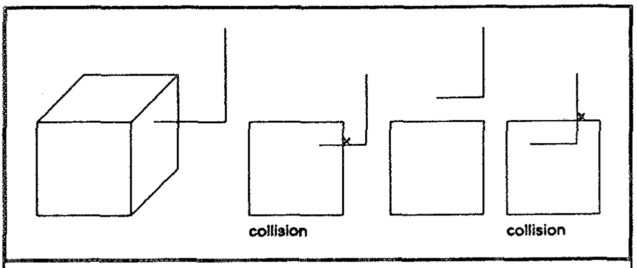


Fig. 4 - Link Obstacle Collision Situation

## Link-Obstacle Module

The links involved in the link-obstacle collision module are 2, 3, 4 and 6 (see Figs. 1 and 2). Link 1, by definition, cannot collide and link 5 has length = 0 and hence it is not considered.

This module finds the intersections of the segment representing a link with the obstacles, that are enlarged by a value equal to the maximum radius of the link. This is the so called 'obstacle growing' technique. The possibility of a collision between the link under consideration and any obstacle is initially verified intersecting the sphere circumscribing the obstacle with the segment representing the link. If no intersection is found the distance between the two objects is such that no collision is possible. Otherwise, a further check is necessary and for each face of the obstacle the intersection between the straight line including the link and the plane containing the face of the obstacle is verified. If an intersection point is found and if this point belongs to the link, then it is to be verified whether the point lies on the obstacle face or not. So it is possible to say that there is a collision if and only if there is one point belonging both to the link and to one face of the polyhedron representing the obstacle.

It has to be noted that the case in which the segment lies on the plane including the face does not represent a collision situation. Moreover, the case of parallelism between the segment and the plane is not considered; in fact, in such a situation a possible collision is found checking for the other links the intersection between the segment and another face, adjacent to the present one. **Figure 4** depicts the situations in which a collision is detected and those in which there is not a collision.

The whole link-obstacle collision module is summarized in the flow-chart of **Figure 5**.

## Link-Link Module

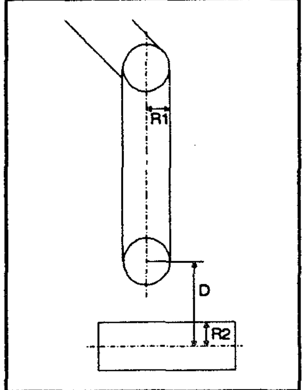


Fig. 6 - Link-Link Collision Test

In the link-link collision module a link is modelled as a cylinder with two hemispheres at its ends. In this module the possibility of a collision between the arm and itself is verified, checking whether the distance between the two segments representing the links is less than the sum of the radia of the links. An example of collision-free situation is given in **Figure 6**.

Referring to Figure 2, only the following couples of links must be verified: 1-3, 1-4, 1-6, 2-4, 2-6 and 3-6.

Starting from the coordinates of the extremes of the links, the minimum distance between two segments in a 3D space is computed [8] through the derivatives, with respect to the two links equations parameters, of the distance equation and solving the resulting system. If its determinant is not zero, the points at minimum distance are computed. If both the two identified points belong to the segments representing the links then their coordinates and their relative distance are computed. If at least one of the points does not belong to the link, eight different cases, i.e. the extremes of one link against a point

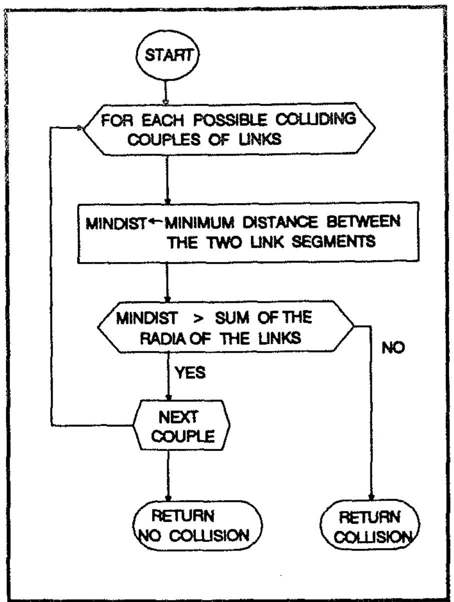

START
FOR EACH POSSIBLE COLLIDING COUPLES OF LINKS
MINDIST←MINIMUM DISTANCE BETWEEN THE TWO LINK SEGMENTS
MINDIST > SUM OF THE RADIA OF THE LINKS
NO
YES
NEXT COUPLE
RETURN NO COLLISION
RETURN COLLISION

Fig. 7 - Link Link Module Flow-Chart

internal to the other link and the extremes of one link against the extremes of the other link, are to be examined. The eight distances so computed are then stored. Among all the above distances the minimum one and the corresponding points are extracted. Otherwise, i.e. if the determinant is zero, the links are parallell and their minimum distance and the relevant points are computed. A check is performed in order to verify whether the two axes are coincident or not. In the first case, the minimum distance is equal to zero if the two segments are overlapped and to the distance between the two nearest extremes, otherwise. In the second case, the minimum distance is equal to the distance between one extreme and the normal projection of the extreme of the other link. This is obtained by computing the intersection between the first link axis and the plane normal to the first link and containing the extreme of the second link.

The whole link-link collision module is summarized in the flow-chart of **Figure 7**.

## Payload-Obstacle Module

In the payload-obstacle collision module the possibility of a collision between the payload and any obstacle is verified. For this reason, a preliminary test is performed in order to exclude those obstacles that have no possibilities of colliding with the payload. This test is performed checking whether the position of the payload is such that a collision with the obstacle under consideration can occur.

In this module both the payload and the obstacle are inscribed in a sphere. At the beginning the distance between the payload and the obstacle is computed and compared with the sum of the radia of the two spheres. If it is larger then there is no collision. If the spheres intersect each other the possibility of a collision between the payload and any face of the currently examined obstacle has to be verified computing the distance from the extremes of the axis of the payload to the plane containing the examined face. Some different cases can arise. Referring to **Figure 8a**, if $D_1 > R$ and $D_2 > R$ and the two extremes are not on the opposite side of the plane then there is no collision between the payload and the face of the obstacle. If such a condition is satisfied the next face is verified. Otherwise, it has to be checked whether the projections of the two extremes of the payload on the plane including the face of the obstacle are internal or external to the face. Referring to **Figure 8b**, if $D_1 < R$ and $P_1$ belongs to the face, or if $D_2 < R$ and $P_2$ belongs to the face, or if the extremes of the

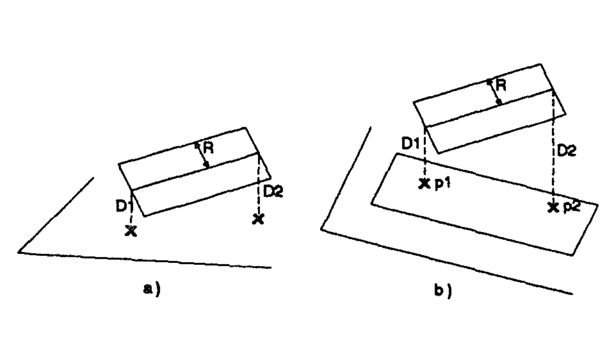


Fig. 8 - Payload Obstacle Collision Test

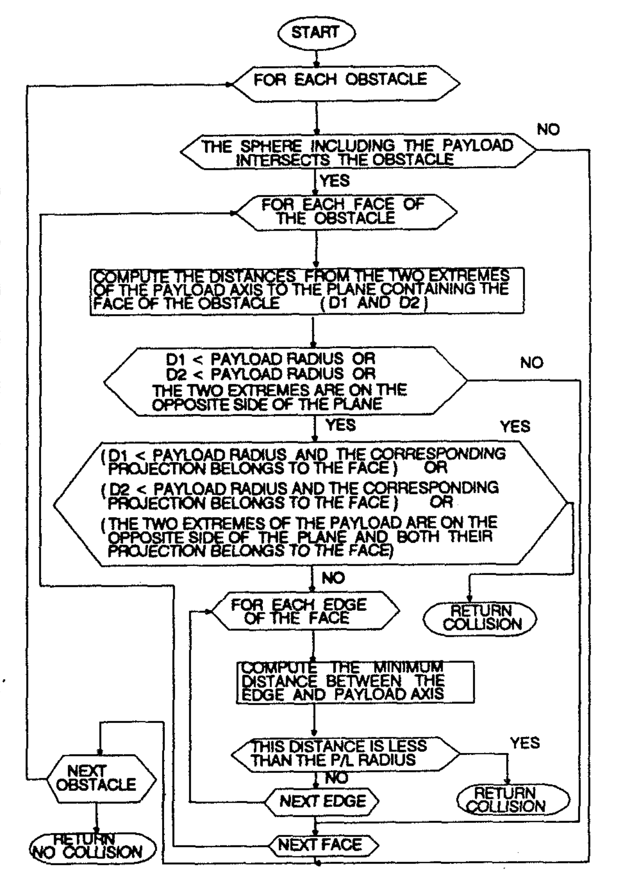

START
FOR EACH OBSTACLE
NO
THE SPHERE INCLUDING THE PAYLOAD INTERSECTS THE OBSTACLE
YES
FOR EACH FACE OF THE OBSTACLE
COMPUTE THE DISTANCES FROM THE TWO EXTREMES OF THE PAYLOAD AXIS TO THE PLANE CONTAINING THE FACE OF THE OBSTACLE (D1 AND D2)
D1 < PAYLOAD RADIUS OR D2 < PAYLOAD RADIUS OR THE TWO EXTREMES ARE ON THE OPPOSITE SIDE OF THE PLANE
NO
YES
YES
(D1 < PAYLOAD RADIUS AND THE CORRESPONDING PROJECTION BELONGS TO THE FACE) OR (D2 < PAYLOAD RADIUS AND THE CORRESPONDING PROJECTION BELONGS TO THE FACE) OR (THE TWO EXTREMES OF THE PAYLOAD ARE ON THE OPPOSITE SIDE OF THE PLANE AND BOTH THEIR PROJECTION BELONGS TO THE FACE)
NO
FOR EACH EDGE OF THE FACE
RETURN COLLISION
COMPUTE THE MINIMUM DISTANCE BETWEEN THE EDGE AND PAYLOAD AXIS
THIS DISTANCE IS LESS THAN THE P/L RADIUS
YES
NO
NEXT OBSTACLE
NEXT EDGE
RETURN COLLISION
RETURN NO COLLISION
NEXT FACE

Fig. 9 - Payload Obstacle Module Flow-Chart

payload are on the opposite side of the plane and both their projections belong to the face then there is a collision. If none of these three conditions is verified a further check is required in order to establish whether a collision situation between each edge of the current face and of the examined solid is presented. For this reason the minimum distance between the two involved segments is computed; if it is less than the radius of the payload then there is a collision.

The whole payload-obstacle collision module is summarized in the flow-chart of **Figure 9.**

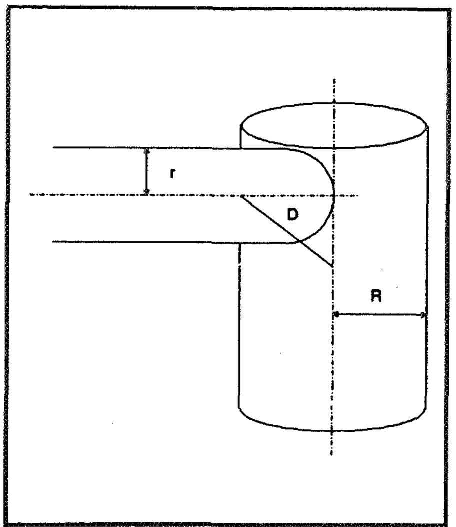


Fig. 10 - Payload Link Distance

## Payload-Link Module

In the payload-link collision module the links which can collide with the payload, i.e. the first four links, are examined. If a sphere is circumscribed to the payload, modelled as a cylinder, the situation is the same as in the link-obstacle collision module. If there is the possibility of a collision the determinant of the second order equation formed by the intersection between the straight line connecting the extreme of the link with the medium point of the payload axis and the sphere is computed. If it is $\geq 0$, i.e there is an intersection between the straight line and the sphere, the roots equation corresponding to the intersection points are computed. If there is an intersection, two cylinders, as in the link-link collision module, having different diameters are considered and the minimum distance between two segments, representing the link axis and the payload axis, in the space is computed, as shown in **Figure 10.** If the distance is less than the sum of the payload radius and the link radius, then there is a collision.

The whole payload-link collision module is summarized in the flow-chart of **Figure 11.**

## Forbidden Values Method

The forbidden values method precomputes the forbidden joint ranges for the first three links and stores them, off-line, as a set of trees, depicted in **Figure 12.** The forbidden ranges represent the link-obstacle collision positions which are accessed, on-line, along the tree. Only a small subset of the data structure, which is represented by the neighbourhood of the current set, is loaded in the central memory. Each substructure represents the forbidden values for an interval of the first two joints, e.g., as depicted in **Figure 13,** when the first joint sweeps from 20 to 30 degrees and the second joint sweeps from 70 to 80 degrees. For each discretized position of the first link, which cannot collide with the obstacles, the intersection angles of the second link with the obstacles are found and the allowed ranges are stored. Then, for each discretized position of the second link within an allowed range, the intersection angles of the third link with the obstacles are computed and the allowed ranges are stored. In the same way the third link is checked for collisions with the first link.

In this method,based on a work of Lozano-Perez [9], the intersection between a link, modelled as a segment, and an obstacle, an enlarged polyhedron, are found computing the rotation center and the plane in which the second and third links rotate and finding the intersections between the sweeping circle and the obstacle faces.

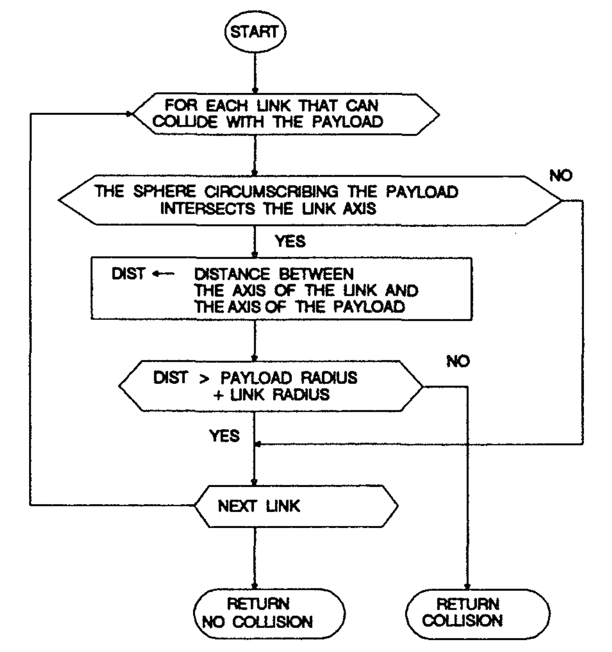

START
FOR EACH LINK THAT CAN COLLIDE WITH THE PAYLOAD
THE SPHERE CIRCUMSCRIBING THE PAYLOAD INTERSECTS THE LINK AXIS
NO
YES
DIST ← DISTANCE BETWEEN THE AXIS OF THE LINK AND THE AXIS OF THE PAYLOAD
DIST > PAYLOAD RADIUS + LINK RADIUS
NO
YES
NEXT LINK
RETURN NO COLLISION
RETURN COLLISION

Fig. 11 - Payload Link Module Flow-Chart

## 4. Computational Results

In this Section the performances of the proposed collision avoidance system are analyzed and some conclusions regarding trade-off between precision, memory and time are presented.

Different running tests of the algorithms have been performed on the basis of 1,000 configurations, randomly chosen in the working area of the arm, to be checked for collision. The main sampled data, related to the geometric method, are the following:

|  |  |  |
|---|---|---|
| • Average # of multiplications | : | 1651 |
| • Standard Deviation | : | 195 |
| • Maximum # of multiplications | : | 6064 |

It is important to emphasize that the maximum and minimum number of multiplications are related, respectively, to the payload-obstacle and payload-link tests. Moreover, the maximum and minimum number of collisions are detected, respectively, in the link-obstacle and payload-obstacle tests.

Assuming that a multiplication requires 1 time unit, a sum 1/3 time unit, a square root 3 time units and a trascendental call 7 time units, a conservative evaluation of the number of FLOP required by the overall algorithm is, in the worst case, about 11,000.

Using a variable stopping distance, an obstacle growing algorithm is also used. For the Hermes model described above, this algorithm requires 325 FLOP. This number grows linearly with the complexity of the model. As the FLOP required by the obstacle growing is about 3% of the total, and because the frequency of activation of this program can be 1/10 of the frequency of activation of the collision avoidance checks, this contribution can be neglected.

The computational cost C (kFLOPS) of the geometric algorithms is given by:

$$C = 11 * f \tag{4-1}$$

where f (Hz) is the frequency of activation of the algorithms, related to the discretization error through the speed by:

$$Ed = v/f \tag{4-2}$$

where v is the speed (m/sec) of the fastest part of the arm and Ed is the discretization error (mm). The discretization error is due to the fact that all the checks are performed only in discrete positions. Assuming a computer power of about 18 kFLOPS, which allows a maximum frequency of activation of 1.7 Hz, it is possible to evaluate this error on the basis of the formulas 4-1 and 4-2. The discretization error for different arm speeds is herebelow reported. The values are related to all the algorithms running together, included the kinematic transform:

|  |  |  |
|---|---|---|
| 50 (mm/sec) | ⇒ | 29 (mm) |
| 100 (mm/sec) | ⇒ | 59 (mm) |
| 200 (mm/sec) | ⇒ | 118 (mm) |

As it has been already mentioned, the precision of the method is affected also by the stopping distance and the geometric model.

The stopping distance is treated as a stepwise function of speed and load of the arm. Letting the maximum value of the stopping distance be 100 mm, i.e. the typical value for maximum speed movements, and using 10 ranges, the error due to this contribution can be assumed, in the worst case, as 10 mm.

The chosen geometric model approximates the curve surfaces of the obstacles with polyhedra; the error due to this contribution depends on the exact shape of the obstacle and on the chosen polyhedrical approximation and can be reduced using a more detailed model of the obstacle and rounding sharp edges and vertices. In the model used for the experiments the worst case error can be estimated as about 200-250 mm. This is a very large value: the use of a more detailed model, that can be easily computed on the baseline of Section 3, is hence recommended.

It is worth to remark that a cylindric model of the payload is too difficult to be treated with a good precision without a too heavy computational charge. Infact, a more detailed model increases the required computing power. This increase cannot be evaluated a priori but only a careful trade-off between precision and time can give acceptable results. The preliminary test (pruning) plays a foundamental role in this trade-off, allowing to obtain a better resolution without a great increase in the required computing power.

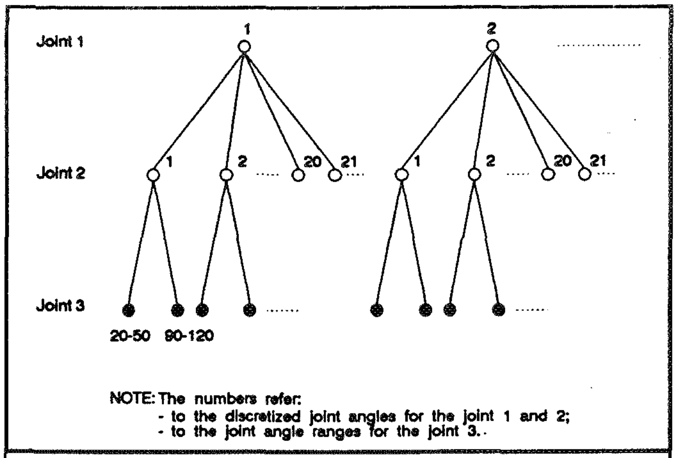

Joint 1
Joint 2
Joint 3
1
2
1
2
20
21
1
2
20
21
20-50
90-120
NOTE: The numbers refer:
- to the discretized joint angles for the joint 1 and 2;
- to the joint angle ranges for the joint 3..

Fig. 12 - Data structure of the forbidden values method

The memory required for the code, using the Fortran 77 language, is about 20 kbyte on a VAX. The memory required for the database, in the case of the Hermes model described above, is about 3 kbytes. Then the use of a quite refined and complex model does not appear to cause problems in terms of memory occupation.

As far as the forbidden values method is concerned, the memory required for the code is about 18 kbyte. The overall structure for the first three joints, with a discretization of 1 degree for the first two joints sweeping over the ranges 0 to 360 and -30 to 210 respectively, has required about 500 MFLOP. The memory required to store this structure is about 600 kbyte.

Also in this case, the dimension of this structure depends on the chosen discretization and on the number of convex obstacles. The dependence on the number of convex obstacles is quite complex and is mainly related to the position and to the size of the obstacles. A reasonable estimate of the memory required by a convex obstacle is 75 kbytes. Then the dimension D of the world model (kbyte) is given by the following formula:

$$D = 75 * N * (1/n1) * (1/n2) \qquad (4\text{-}3)$$

where n1 and n2 are, respectively, the discretization in degrees of the first and second joint angles and N is the number of obstacles.

The chosen discretization affects the precision of the method. The error due to this contribution is given by:

$$Ed = 2((L3 * \tan(n2/2))^2 + (L2 * \cos(n2) * \tan(n1/2))^2)^{1/2} \qquad (4\text{-}4)$$

where L2 and L3 are the length of the second and third link, respectively.

Assuming n1 = n2 = n and considering cos(n) about equal to 1 and tan(n/2) about equal to $n\pi/360$, the discretization error Ed (cm) is given by:

$$Ed = n\pi/180 * (L2^2 + L3^2)^{1/2} = 11.4 * n \qquad (4\text{-}5)$$

In the proposed model the overall structure is composed of 864 substructures, each of them describing a range of 10x10 degrees (see Fig. 13). The maximum dimension of this structure sufficient to store all the ranges is 310 integer pointers and 400 angular values, which implies a memory request for each substructure of about 1.5 kbytes; assuming to have in memory 9 substructures, according to the chosen loading strategy, 13.5 kbytes are hence required.

The central memory required for some typical values of the discretization error are herebelow reported, considering the case of 8 obstacles:

| | | |
|---|---|---|
| 5 (cm) | ⇒ | 70 (kbyte) |
| 10 (cm) | ⇒ | 18 (kbyte) |
| 15 (cm) | ⇒ | 8 (kbyte) |
| 20 (cm) | ⇒ | 4 (kbyte) |

It has to be remarked that, because the forbidden values structure is built off-line, the stopping distance must be assumed as the worst value.

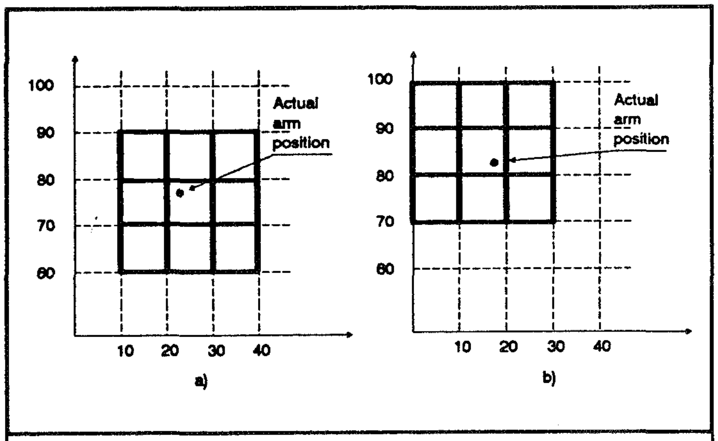


Fig. 13 - Loading Strategy

As a general performance analysis result of both the geometric and the forbidden values method, it is possible to state that:

- the processing power increases linearly with the number of solids
- the errors decrease linearly with the number of solids.

## 5. Conclusions and Future Work

The overall collision avoidance system architecture is shown in **Figure 14**: basically, it is a merging of the geometric method and the forbidden values method. This architecture applies to both the off-line and the on-line computation assuming a proper system initialization.

The current Tecnospazio study on collision avoidance has shown that such a collision avoidance software system is feasible with respect to the resources available on board, considering its performances.

On the basis of this assumption in the next future the proposed collision avoidance system will be enhanced and completed. Moreover, the generation of the HERA control system software for collision avoidance along with the implementation of a prototype on the HERA simulation facility will constitute the future activities of Tecnospazio in the area of collision avoidance with respect to the HERMES manipulator.

Furthermore, the developed software will be used, with the proper modifications, in the more complex problem of collision avoidance between two manipulator arms which operate within the same working area. This implementation will require a further refinement of specific algorithms together with a modeling improvement in order to allow tighter arm trajectories without falls or unwanted collision detections.

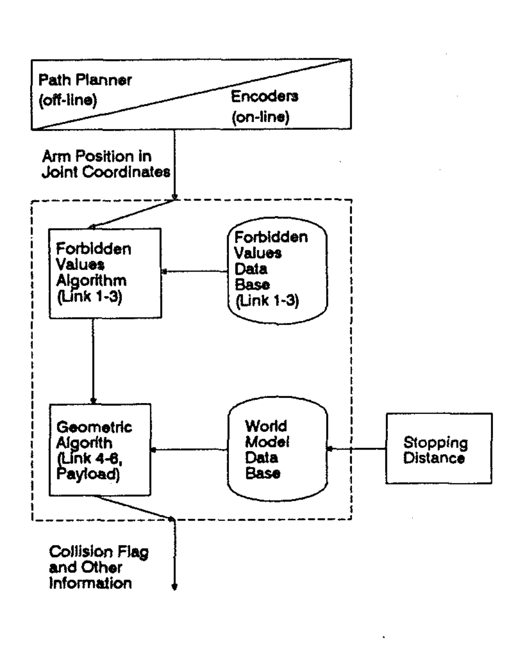

Path Planner (off-line)
Encoders (on-line)
Arm Position in Joint Coordinates
Forbidden Values Algorithm (Link 1-3)
Forbidden Values Data Base (Link 1-3)
Geometric Algorith (Link 4-6, Payload)
World Model Data Base
Stopping Distance
Collision Flag and Other Information

Fig. 14 - Overall System Architecture

## REFERENCES

[1] Matra Espace, "Survey of the State of the Art in Robotics and Artificial Intelligence", Technical Report, 1981

[2] NASA, "Advancing Automation and Robotics Technology for the Space Station and for the U.S Economy", TM 87566, 1985

[3] Fokker Space & Systems, "FSS-N-88-HERA-042", 1988

[4] Fokker Space & Systems, "FSS-N-88-HERA-051", 1988

[5] Tecnospazio S.p.A, "HPP: Study on Collision Avoidance", Final Report, STUD005-87, 1987

[6] Tecnospazio S.p.A, "HERA Bridging Phase: Study on Collision Avoidance", Final Report, STUD009-88, 1988

[7] J. Denavitt, R.S. Hartemberg, "A Kinematic Notation for Lower-pair Mechanism Based on Matrices", ASME J. Apll. Mech., 22, 1955.

[8] D.G. Luenberger, "Introduction to Linear and Nonlinear Programming", Addison-Wesley Publishing Company, 1983

[9] T. Lozano-Perez, "A Simple Motion Planning Algorithm for General Robot Manipulators", IEEE J. of Robotics and Automation, June 1987.

# ISSUES IN AI SYSTEMS

# Generic Task Problem Solvers in Soar

Todd R. Johnson, Jack W. Smith, Jr., B. Chandrasekaran

Laboratory for AI Research (LAIR)
Department of Computer and Information Science
The Ohio State University
Columbus, Ohio 43210

## 1 Introduction

Two trends can be discerned in research in problem solving architectures in the last few years: On one hand, interest in *task-specific architectures*[3, 8, 2] has grown, wherein types of problems of general utility are identified, and special architectures that support the development of problem solving systems for those types of problems are proposed. These architectures help in the acquisition and specification of knowledge by providing inference methods that are appropriate for the type of problem. However, knowledge-based systems which use only one type of problem solving method are very brittle, and adding more types of methods requires a principled approach to integrating them in a flexible way.

Contrasting with this trend is the proposal for a flexible, general architecture contained in the work on Soar[4]. Soar has features which make it attractive for flexible use of all potentially relevant knowledge or methods. But as a theory Soar does not make commitments to specific types of problem solvers or provide guidance for their construction.

In this paper we investigate how task-specific architectures can be constructed in Soar to retain as many of the advantages as possible of both approaches. We will be using examples from the Generic Task (GT) approach for building knowledge-based systems in our discussion since this approach had its genesis at our Laboratory where it has further been developed and applied for a number of problems; however the ideas are applicable to other task-specific approaches as well.

## 2 The GT Paradigm

The GT paradigm is a theory of types of goals and the problem solving methods needed to achieve each type. By problem solving method we mean the specification of behavior to achieve a goal. The paradigm has three main parts:

1. The problem solving of an intelligent agent can be characterized by generic types of goals. Many problems can be solved using some combination of these types.

2. For each type of goal there are one or more problem solving methods, any one of which can potentially be used to achieve the goal.

3. Each problem solving method requires certain kinds of knowledge of the task in order to execute. These are called the *operational demands* of the method[7].

The term *generic task* refers to the combination of a type of goal with a problem solving method and the kinds of knowledge needed to use the method. The GT for classification by establish-refine (called the E-R GT) is given as:

**Type of Goal** Classify a (possibly complex) description of a situation as a class in a set of categories. An instance of this goal is the classification of a medical case description as one of a set of diseases.

**Problem Solving Method** This is a hierarchical classification method that works by creating and testing hypotheses about the plausibility that the description of the situation represents an instance of one or more of the classes.

1. If there are no initial hypotheses about what class the description is an instance of, then try to suggest at least one.
2. Try to confirm or reject any hypothesis that is suggested.
3. If a hypothesis is confirmed, determine the possible refinements of the hypothesis and suggest them.
4. If the goal is not met, go to step 2.

**Kinds of Knowledge** These consist of a refinement hierarchy, hypotheses about the presence of classes, confirmation/rule-out knowledge for these hypotheses, and knowledge to determine when the goal of classification has been achieved.

In addition to classification by establish-refine, GT's have been created for pattern directed hypothesis matching[5], assembly of explanatory hypotheses[6], and object synthesis by plan selection and refinement[1].

# 3 GT Systems

A specialized architecture or shell has been constructed for each GT. Each architecture is a combination of an inference engine with a knowledge base. The inference engine is a procedural representation of a GT's problem solving method. The knowledge base provides primitives for encoding the domain specific knowledge needed to instantiate the procedure. We refer to the combination of the encoding of the domain knowledge in the knowledge base and the method that can use it as a problem-solver.

This system building approach offers a number of advantages: First, it is easy to decide when a GT architecture can be used because the knowledge operationally demanded by the method is explicit in the definition of the GT. Second, knowledge acquisition is facilitated because the representational primitives of the knowledge base directly correspond to the kinds of domain knowledge that must be gathered. Third, explanation based on a run-time trace can be couched in terms of the method and knowledge being used to apply it.

# 4 Problems with GT Systems

Many flexibility problems arise because a GT architecture contains assumptions not present in the original GT problem solving method. For example, our architecture for hierarchical classification assumes that hypotheses are generated from a pre-defined hierarchy. While this is a common way to generate refinements, other ways exist and might be useful in certain domains. Second, the architecture immediately generates refinements for a confirmed hypothesis. An alternative is to test all the hypotheses in the current state before refining any that were confirmed. Third, the architecture assumes that any problem solver it calls will correctly function. We cannot easily modify the architecture to gracefully handle these situations.

Another set of problems involves the integration of multiple GT problem solvers. The simplest kind of integration is when one problem solver calls on another as a direct means to achieve a subgoal. This is easily done using our current architectures by directly invoking the method and domain knowledge needed to achieve a subgoal. However, sometimes we require more interaction between the problem solvers. For example, in our medical diagnosis systems the hypothesis assembly problem solver has knowledge about those diseases that can occur together and those that are mutually exclusive. This knowledge can be used help guide the classification of diseases; however, it is difficult to implement because the classification architecture has no place for representing or using this knowledge. Our only solution was to specially modify both architectures so that they could interact in the desired way.

Finally, new methods are difficult to add to existing problem solvers; each problem solver must be modified to recognize and use a new method. We would like to have the system automatically consider methods based on the type of goal a method is designed to achieve.

# 5 How can Soar Help?

In Soar, all problem solving is viewed as search for a goal state in a problem space. Operators are used to move from state to state. Knowledge in the form of productions is used to select problem spaces, states, and operators. Productions generate *preferences* for an object (ie. a problem space, state, or operator) that indicate how the object relates to the current situation or other objects. Whenever the directly available knowledge is insufficient to make progress Soar automatically generates a subgoal. Therefore, every decision that needs to be made can become a goal to be achieved by searching a problem space. This is called *universal subgoaling.* In knowledge lean situations Soar can make progress by using an appropriate weak method. The weak methods are not explicitly programmed in Soar, but arise from the knowledge available to solve a problem. If the processing in a subgoal is no longer needed, Soar will automatically terminate the subgoal and resume problem solving in a higher level goal. This is called *automatic goal termination.*

Each of these features directly relate to one or more of the limitations with GT systems. The selection of alternatives via preferences allows new options and knowledge to be easily added to existing systems. Brittleness is decreased because of Soar's ability to automatically create subgoals to overcome failures and its ability to fall back on weak methods. Finally, automatic goal termination eliminates unnecessary computation and provides a more natural flow of control.

# 6 Mapping GT's to Soar

We have begun to map GT's to the Soar architecture in a straight-forward manner. Each GT is implemented as a problem space; the states represent the developing solution and the operators and operator suggestion rules implement the problem solving method. The required kinds of knowledge can either be represented directly by productions or generated at run-time using additional problem spaces.

To illustrate, we present ER-Soar, an implementation of the E-R GT in Soar. We use a single problem space with three operators: suggest-initial-hypotheses, establish, and generate-refinements.

**State Representation** The state contains those hypotheses that have been considered and those that are worth immediately considering. Any hypotheses in the state that are refinements of other hypotheses (also in the state) are linked together to form a refinement hierarchy. Each hypothesis also has an indication of whether it has been confirmed, rejected, or not yet judged, and whether it has been refined or not.

**Operators** The classify problem-space uses 3 operators:

**suggest-initial-hypotheses** Determine one or more initial hypotheses.

**establish <hyp>** Determine whether the hypothesis, <hyp>, should be confirmed or rejected.

**generate-refinements <hyp>** Generate (add to the state) those hypotheses that should be considered as a refinement of <hyp>.

**Operator Instantiation** A suggest-initial-hypotheses operator is created if there are no hypotheses in the current state. An establish operator is created for any hypothesis in the current state that has not yet been judged. A generate-refinements operator is created for any hypothesis that is confirmed but not refined.

**Domain knowledge** To use the E-R strategy in a particular domain, knowledge to perform the following functions must be added to the Soar implementation.

- Create the initial state containing one or more initial hypotheses.
- Detect when classification is complete.
- Implement the three operators.

**Operator Implementation** There are many ways to implement the operators used in the classify space. To make ER-Soar easy to use we have implemented a method for generating refinements from a pre-defined hierarchy and a method for establishing hypotheses based on a confidence value.

## 6.1 Discussion

ER-Soar combines the advantages of the GT approach with the advantages of the Soar architecture. Knowledge acquisition, ease of use, and explanation are all facilitated in ER-Soar because subgoals of the problem solving method and the kinds of knowledge needed to use the method are explicitly represented in the implementation. The subgoals of the method are directly represented as problem space operators. The kinds of knowledge needed to use the method are either encoded in productions or computed in a subgoal. The same advantages apply to the supplied methods for achieving subgoals. Finally, the implementation mirrors the GT specification quite closely making ER-Soar easy to understand and use.

ER-Soar overcomes many of the problems suffered by previous GT systems. Automatic subgoaling allows unanticipated situations to be detected and handled. If no specific method for handling the situation is available, an appropriate weak method can be used. Whenever a goal needs to be achieved it is done by first suggesting problem-spaces and then selecting one to use. This allows new methods in the form of problem-spaces to be easily added to existing problem solvers. If no specific technique exists to determine which method to use, Soar will try to pick one using a weak method. Automatic goal termination provides an integration functionality not available in previous GT architectures. In general, the integration capabilities of ER-Soar are greatly enhanced. Because of preferences and the additive nature of productions, new knowledge can be added to integrate ER-Soar with other methods without modifying existing control knowledge.

# 7 Conclusion

ER-Soar illustrates how the advantages of task-specific architectures can be combined with the advantages of a general architecture. The approach used to create ER-Soar is simple and can easily be applied to other task-specific architectures. We are currently using this approach to create Soar versions of the GT's for hypothesis matching and hypothesis assembly. Following this, we will investigate various ways of integrating the three methods.

# 8 Acknowledgements

This research is supported by National Heart Lung and Blood Institute grant HL-38776, Air Force Office of Scientific Research grant 82-0255, and Defense Advanced Research Projects Agency, RADC Contract F30602-85-C-0010. Dr. Jack W. Smith, Jr. is supported by National Library of Medicine Career Development Award LM-00083. Computer facilities were enhanced by gifts from Xerox Corporation.

# References

[1] D. C. Brown and B. Chandrasekaran. Plan selection in design problem-solving. In *Proc. of AISB85 Conference*, Warwick, England, 1985. The Society for AI and the Simulation of Behavior (AISB).

[2] B. Chandrasekaran. Generic tasks in knowledge-based reasoning: High-level building blocks for expert system design. *IEEE Expert*, 1(3):23–30, 1986.

[3] W. J. Clancey. Heuristic classification. *Artificial Intelligence*, 27(3):289–350, 1985.

[4] Paul S. Rosenbloom John E. Laird, Allen Newell. SOAR: An architecture for general intelligence. *Aritificial Intelligence*, 33:1–64, 1987.

[5] Todd R. Johnson, Jack W. Smith Jr., and Tom Bylander. HYPER: Hypothesis matching using compiled knowledge. Technical report, Lab. for AI Research, CIS Dept., The Ohio State University, Columbus, Ohio, 1988.

[6] J. R. Josephson, B. Chandrasekaran, J. W. Smith, and M. C. Tanner. A mechanism for forming composite explanatory hypotheses. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-17(3):445–454, 1987.

[7] John Laird, Paul Rosenbloom, and Allen Newell. *Universal Subgoaling and Chunking.* Kluwer Academic Publishers, Massachusets, 1986.

[8] Marcus, Sandra McDermott, and John McDermott. Salt: A knowledge acquisition tool for propose and revise systems. Technical report, Carnegie-Mellon University, 1987.

# TEMPORAL LOGICS MEET TELEROBOTICS

Eric RUTTEN , Lionel MARCÉ

IRISA / INRIA - Rennes
F-35042 RENNES Cedex FRANCE
e-mail: rutten@irisa.UUCP, marce@irisa.UUCP

**Abstract**

The specificity of telerobotics being the presence of a human operator, decision assistance tools are necessary for the operator, especially in hostile environments. In order to reduce execution hazards due to a degraded ability for quick and efficient recovery of unexpected dangerous situations, it is of importance to have the opportunity, amongst others, to simulate the possible consequences of a plan before its actual execution, in order to detect these problematic situations. Hence the idea of providing the operator with a simulator enabling him to verify the temporal and logical coherence of his plans. Therefore, we intend to use the power of logical formalisms for representation and deduction purposes. Starting from the class of situations that we want to represent, we will adapt a *STRIPS*-like formalism and its underlying logic to the simulation of plans of actions in time. The choice of a temporal logic enables us to build a world representation, on which the effects of plans, grouping actions into control structures, will be transcribed by the simulation, resulting in a verdict and information about the plan's coherence.

## 1 Introduction

The specificity of telerobotics is the presence of a human operator in the classical *perception-decision-action* loop. His work is then to supervise execution and, most importantly, to decide of the tasks to be completed by the teleoperated robot, and of their ordering. It is in this context that decision-making assistance to the operator seems necessary, in order to test and validate the plans he conceives, before executing them, and this especially in hostile environments. In the case of space telerobotics, when communications take place with a very remote robot site, the time elapsing between operator reaction and decision, and robot action, can be very long; furthermore, manipulation is made difficult by the physical characteristics of the environment, unnatural to the human operator. All this degrades the ability for quick or efficient recovery, by the operator, of errors or unexpected accident.

Hence the idea of providing the operator with a decision assistance system, comprising functionalities such as planification, memorizing of events, managing of warnings and signals, and simulation, in order for him to have an information as complete as possible about the state of the environment he is working upon, including the robots he teleoperates, its past evolutions, and the possible futures entailed by the actions he could undertake. The decision assistance function of a teleoperation system goes along, and works together with other essential features: execution control and monitoring, at various levels between autonomy and manual command, sensor information treatment and fusion, or interactive modelling of the environment. In the frame of a system comprising all these elements, a simulator can enable the operator to verify the logical and temporal correctness of his plans with regard to their effects on the world being worked upon.

A decision assistance tool needs as its base a representation of knowledge, on which various required treatments can be performed, providing a formal frame in which to store the gathered information. The expressiveness and inferential power of logics make them a particularly adapted tool therefore, especially as recent logical systems have been built in order to capture notions otherwise only difficultly mastered, such as time by temporal logics. Our work is thus directed towards a meeting of the interests of telerobotics and of temporal logics, being branches respectively of robotics and artificial intelligence, through characteristic, paradigmatic problems to be solved, leading to the construction of meaningfull formalisms. As to what knowledge we have to represent, in the particular case interesting us, it consists of the world, the environment in which things happen (its state, and the laws describing what can happen into it, and what not), and the changes that can occur, especially the actions that the robot can execute. Research in artificial intelligence associated with robotics has already provided us with concepts and techniques addressing these knowledge representation issues.

## 2 Knowledge representation issues

A classical model of robots acting on an environment is the *STRIPS* planner (the *STanford Research Institute Problem Solver* ) [7], using a formalism based on first-order logic. In *STRIPS*, the world is represented by a set of predicates describing its state, and an action is represented by three lists: the *preconditions list*, specifying the predicates that must be found to hold in the current state of the world, in order for the action to be executable; the *delete list*, composed of the predicates that do not hold after the execution of the action, and are retracted from the world state, as a negative effect of it; the *add list*, of which the predicates are added to the world state, as a positive effect of the action. In this way, an action is an operator transforming a situation of the world into another. The plans generated by *STRIPS* are sequences of such actions, forming a transformation from the initial situation to a final situation including the goal of the plan.

The limitations of this model are in the strict linearity of the plans produced, and in the underlying logics. Since then, other studies have achieved the construction of models allowing a less restricted structure of the plans [6], through less commitment as to the order between actions, e.g. using a partial order on their set ... One of the directions of extension of these formalisms has been that of **temporal logics**, dealing explicitely with time [2, 10], in order to have a representation of duration, change and simultaneity. These temporal logics have been used for plan generation, the planning process then consisting in mapping the actions in time in order to obtain the goal state from the initial one through the execution of the plan. We will explore them here as a knowledge representation and deduction tool. A bibliographical study shows a great variety of existing formalisms, many of them still in evolution. Amongst them, a choice has to be made of the concepts applicable to the class of problems interesting us.

Our approach deals with the simulation of plans composed by the human operator in teleoperation. In order for him to have at his disposal a facility for writing, reading, and modifying the plans, we will provide him with a representation for the control structures of **plans and actions**. The representation of actions is expanded with a durational information, and possibilities of a hierarchy of subplans provide modularity in plan construction. A language is defined for plan writing, introducing basic operators, as well as more customized ones, to build complex, yet clear, control structures.

The process of **simulation** then reflects the effects of the plan given as input on the world representation. Each of the primitive actions of the plan is simulated, following the control structure of the plan framing them. The output consists in the verdict about the plan executability, and information about the states of the world reached by the possible alternative executions, due to imprecisions in the actions ordering. The state of the world can be examined along its changes, that are traced. In case of an event compromising the plan's executability, warnings and error messages can be issued, guiding the fixing of the plan.

# 3 Temporal logics

We will here briefly present the temporal logic formalism that we have chosen, inspired by Allen [3] and Shoham [11]. The reasons for the choice of this kind of formalism, in preference to modal tense logics like those used in program specification and verification, is that their origin in "*theoretical robotics*" gives them a more immediate expressivity for the class of problems interesting us. Furthermore, a complex logical system is not yet needed in our approach, even if the use of more elaborate logic operators can provide interesting perspectives.

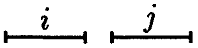
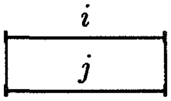
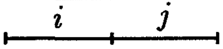
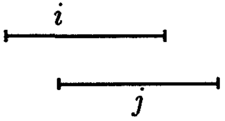
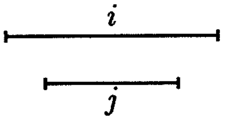
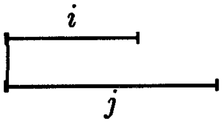
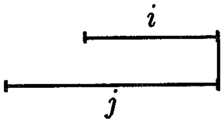
**Intervals** We will adopt the notion of intervals being "chunks" of time, between which relations have been determined by Allen, and are shown in figure 1. These relations can be grouped in a disjunctive relation if the precise relation is not known; e.g. $((I_1\ r_1\ I_2) \vee (I_1\ r_2\ I_2))$ can be written: $(\ I_1\ (r_1\ r_2)\ I_2\ )$. Moreover, a transitivity table has been built on these bases [2], allowing to determine, knowing two relations $r_1$ and $r_2$ between three intervals $I_1$, $I_2$ and $I_3$, i.e. $I_1\ r_1\ I_2$ and $I_2\ r_2\ I_3$, a third relation $r_3$ obtained transitively: $I_1\ r_3\ I_3$.

**Temporal facts** As summarized by Shoham, facts represented in classical first order logic by predicates, are here "reified" into temporal facts. Those take the form: *true( I, P )*, meaning that the fact $P$ holds on the interval $I$. The rules of classical logic are assumed to hold, in the way established by Shoham [11], stating that, for example, *true( I,* $P_1 \wedge P_2$*)* is satisfied if and only if both *true( I,* $P_1$*)* and *true( I,* $P_2$*)* are, and that *true( I,* $\neg P$*)* holds if and only if, for no subinterval $I'$ of $I$, we can have *true(* $I'$*, P)*.

**Classification by inheritance** Apart from the relations transitivity table, Shoham gave a definition for another way of getting new temporal information from known facts, by classifying temporal facts according to the relation of their truth over one interval to their truth over other intervals [11]. Examples of classes are: *downward-hereditary* (e.g. "The robot travelled less than two miles.", when true over an interval, is true over all its sub-intervals), *upward-hereditary* (e.g. "The robot travelled at the speed of two miles per hour.", when holding for all the proper sub-intervals of a nonpoint interval, holds for the nonpoint interval itself), *liquid*:

both *upward-hereditary* and *downward-hereditary* (e.g. "The robot's arm was in the GRASPING state"), *solid* (e.g. "The robot executed the NAVIGATE procedure (from start to finish)" never holds over two properly overlapping intervals), ...The temporal facts ( *true(I,Fact)* ) described in the previous paragraph can be incorporated to the *liquid* class. The *solid* class can be used to describe the execution of an action or a plan (e.g. by *exec(I,Plan)*.)

We are thus provided with elements, from which a representation formalism can be developped, allowing us to describe a world, and to manipulate this information, using rules with which further information can be obtained, i.e. reasoning can be made.

| **relation** | **symbol** | **symbol for the inverse relation** | **graphical example** |
|---|---|---|---|
| *i before j* | < | > |  |
| *i equals j* | = |  |  |
| *i meets j* | m | mi |  |
| *i overlaps j* | o | oi |  |
| *i during j* | d | di |  |
| *i starts j* | s | si |  |
| *i finishes j* | f | fi |  |

After Allen [2].

Figure 1: The 13 possible relations between two intervals $i$ and $j$.

## 4 Plans and actions

As in our approach, a human operator is involved in the process, i.e. he writes himself the plans for the robot, we have to provide him with a language in which to express the plans, that will allow him to build complex, yet clear, control structures. The basic elements of this language are the *primitive actions*, which have a form inspired by *STRIPS.* These actions can be grouped into structures determined by *control operators* specifying in what way the actions they frame will take place in time.

**Primitive Actions** A classical representation for an action is that of *STRIPS.* We however want to add a temporal information to this, which is absent in *STRIPS.* We will therefore associate, with each action occurence or execution, an interval, the extent of which is the duration of the action. An action is noted as shown in figure 2.

| **representation** | **example** |
|---|---|
| *action( name(parameters),*<br>*duration,*<br>*preconditions,*<br>*negative effects,*<br>*positive effects ).* | *action( putoff(Cube1,Cube2),*<br>3.14 *s,*<br>*[clear(Cube1),on(Cube1,Cube2)],*<br>*[on(Cube1,Cube2)],*<br>*[clear(Cube2),on(Cube1,table)] ).* |

Figure 2: Actions representation.

As Vere did [12], we make the "changes on termination" assumption, deciding that all the changes entailed by the action occur at the end of its occurence interval. Nevertheless, the preconditions have to be holding on all the interval of the action. Such an action takes place in time in the way shown in figure 3. The "changes on termination" assumption, however, is not limitative: the definition of *compound actions*, presented a little further, allows to define actions having effects in another way. At the moment of simulating such an action, the deduction tools of the temporal logics will be useful to find out wether the preconditions hold over the interval, and how the effects interact with other facts in the world representation.

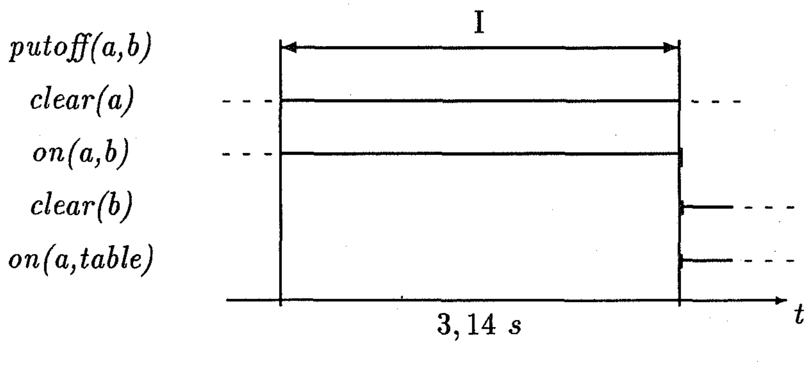


Figure 3: Example of an action: *putoff(a,b)* in time.

**Plans** A plan, as said earlier, is considered here as a set of actions, provided with a *control structure.* A plan is then composed of subplans, which are plans themselves, recursively, down to the *primitive actions* defined earlier, which each constitute a plan themselves. The basic constructs of the language are:

**sequentiality** : noted *seq(<subplans list>)*. For a subplans list $[P_1|P]$, where $P_1$ is the first subplan of the list, and $P$ the remainder of the sequence, this control operator states that the subplans of the list are executed one after the other, in the order of the sequence, as shown in figure 4.

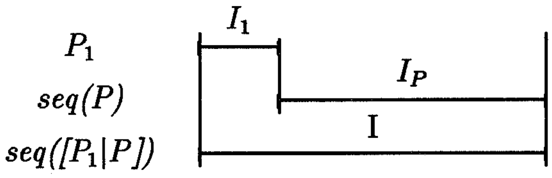


Figure 4: A sequence in time.

**parallelism** : noted *par(<subplans list>)*, where the list of subplans contains the plans constituting each a branch concurrent to the others. In this construct, all branches $B$ start together, and the parallelism ends when all branches have ended, i.e. with the longest lasting branch, $B_P$, as shown in figure 5.

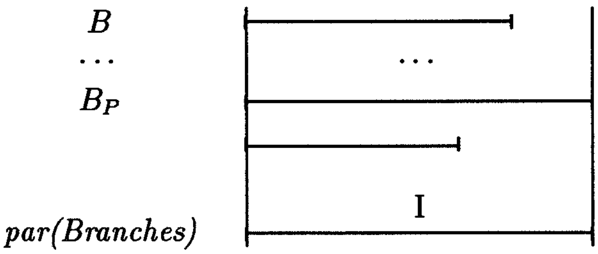


Figure 5: Parallelism in time.

**conditionality** : noted *cond(C,$P_{true}$, $P_{false}$)*. The condition $C$ is evaluated at the beginning of the interval, and according to the result, the corresponding subplan is executed.

**Compound Operators and Actions** From these operators, *compound operators* can be constructed, like, for example, *conditional iteration*, defined recursively as:

```
while( condition, iteration-body )
      ≡
cond( condition,
        seq([ iteration-body,
              while(condition, iteration-body) ]),
        nothing ).
```

where *nothing* is a null action, taking no time, requiring no precondition, and entailing no effect. In the same manner, *compound actions* can be defined, using a plan, in the following way: *action( name(arguments) , plan )*. This allows to consider actions with effects dispatched along their duration, or depending on the context where they are executed. For example, a way of defining an action $a$ realizing effects inside (e.g. at a time $d_1$ after the beginning of) its interval (of total duration $d = d_1 + d_2$), is to decompose it in the following way: *action( a, seq([$a_1, a_2$]) )*, where we have an action $a_1$ of duration $d_1$, with all the preconditions and effects of the action $a$, and another action $a_2$, having the preconditions and not the effects of $a$, being there to continue the verification of the precondition.

This language should enable the writing of complex plans, composed of actions defined by their effects on the world. The temporal dimension of these actions and plans having been specified, we will now see how simulation on these bases is considered.

# 5 Simulation

The simulation of a plan then consists, given a representation of the world to which it is applied, in a modification of this representation by transcribing onto it the effects of the actions of the plan, taken following the control structure leading their flow.

**The simulation process** The world representation consists of a set of temporal facts of the kind described earlier. A classical example is that of a blocks world, as shown in figure 6. The intervals associated to the facts ($I_1$, $I_2$, $I_3$) have unconstrained ends: the truth of the facts holds from the beginning of each interval (in the "past" or "present", before the action) to an undetermined, or uninstanciated "future" (which can be determined by the events in the world, later in time.) A set of predefined actions is at the disposal of the planner, who can build a plan, constructed with control operators like those seen before. This plan is given as input to the simulation process, that begins with decomposing it down to primitive actions in basic control structures. Then, primitive actions are simulated one after the other, in a succession determined by the control structure of the plan.

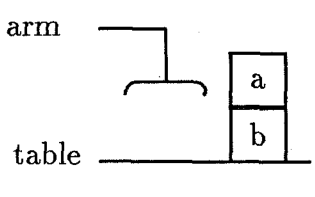
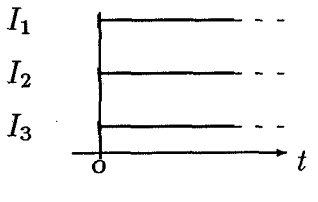
| The blocks world | The temporal facts | The intervals |
|---|---|---|
|  | *true($I_1$, clear(a))*<br>*true($I_2$, on(a,b))*<br>*true($I_3$, on(b,table))* |  |

Figure 6: Example of an environnement and its representation.

**Primitive Actions** The simulation of each primitive action, on an interval of which the extent is given by the duration, consists in constraining the truth intervals of the temporal facts concerned so that they verify the preconditions and transcribe the effects in the world representation. For example, the application of the action *putoff(a,b)* shown earlier in figure 3, on the blocks world of figure 6, gives the new set of temporal facts, after the action, shown in figure 7. In this example, the application of the action *putoff*, on an interval $I$, entails that:

- $I_1$ and $I_2$, being the intervals corresponding to the preconditions, must contain $I$;
- $I_2$, as a result of the negative effect of *putoff*, is constrained to end with $I$ (i.e. $I_2$ (*finishes equals finished-by* ) $I$);
- two new intervals, $I_4$ and $I_5$, are associated with the positive effects, and constrained to be *met-by* $I$ (i.e. $I$ *meets* $I_4$ and $I$ *meets* $I_5$), which determines their beginning, but leaves their end unconstrained yet.

$I_3$, not being concerned by this action, is not imposed any particular constraint.

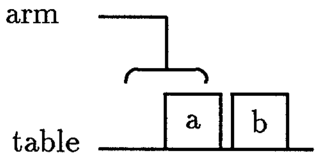
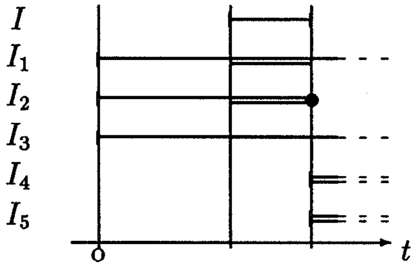
| The blocks world | The temporal facts | The intervals |
|---|---|---|
|  | *exec(I, putoff(a,b))*<br>*true($I_1$, clear(a))*<br>*true($I_2$, on(a,b))*<br>*true($I_3$, on(b,table))*<br>*true($I_4$, clear(b))*<br>*true($I_5$, on(a,table))* |   $I$<br>$I_1$<br>$I_2$<br>$I_3$<br>$I_4$<br>$I_5$<br>0<br>$t$ |

Figure 7: The environment after the execution of *putoff*.

At this stage, the temporal logic is used to check the coherence of the new constraints introduced with the others, and to verify the compatibility of the effects with the temporal facts according to rules specified in the world representation. This can be done using classical deduction methods. Another method for maintaining a coherent set of temporal relations is to build a lattice of time-points, and to define retrieval and updating operations upon it [9]. An *incoherence* is said to be encountered in the simulation, if either a precondition fails to be satisfied, or a contradictory constraint or an incompatibility of the effects is detected.

**Plans** The simulation of plans such as those described before, then consists in the succession of simulations of the primitive actions they contain, following the order specified by the control structure. The task of simulating the plans can then be described as that of choosing the next primitive action to be simulated, and determining the associated interval:

**sequentiality** : the subplans are simulated in the specified order (see figure 4);

**parallelism** : here the choice is made between the actions of the different branches, the criterion being the following of the chronological order, the earliest terminating primitive action being simulated first. An interesting case is that of orderings between actions that are not uniquely determined: their intervals can have one of several possible relative positions. In this case, the simulator will try each relation, "forcing" the intervals into an order, and simulating further for this possibility, until reaching the end of that simulation. Then a backtracking takes place, back to the last choice made, where, if another relation is left untried, it is taken as a new choice, and the simulation is resumed from this point.

**conditionality** : the satisfaction of the condition is checked for in the world representation, and according to the result, the corresponding subplan is simulated.

**compound operators and actions** are rewritten, following their definitions in terms of basic elements, and simulated in that state, as normal plans.

**Results** In this way, the tree of possible executions of the plan, each one related to a possible disposition of the actions in time, is explored. The progression in a branch of this tree stops in two cases: either the end of the plan is reached: we then have the result that it is executable, and that the consequent state of the world is the one reached at this point; either an *incoherence* is encountered, the result then consists in the verdict of unexecutability of the plan, with some information on the reasons of this failure, in order to help modifying the plan so that it would give a success. In the problem of detecting incoherences through simulation, it is important to make an exhaustive examination of the possible situations accessible in the model [8]. This stresses the need for methods of exploration through the tree of possible executions, in order to sort out the incoherent ones.

To summarize, a plan built as seen before can be simulated with regard to its effects on the environment being worked upon, and also to the "internal" coherence of the temporal ordering of its actions, along its control structure. Exploring the possible consequences of that plan provides an operator with an assistance for the evaluation of the correspondance between the effects of the plan and what is expected of its execution.

# 6 Perspectives

The problems addressed by our work are manifold, and the solutions proposed can be improved. The knowledge representation aspects of it must be extended and strengthened, in order to achieve the detection of difficult (e.g. indirect) incoherences. The treatment of indeterminacy in the actions order entails a problem of exhaustivity of the search in the tree of possibilities. All these problems are still worked upon, and will be confronted to experimentations on situations studied in collaboration with MATRA-Espace (in link with the European Space Agency), in the case of a teleoperated arm [4], and with the C.E.A. (the French Nuclear Energy Agency), for mobile telerobots. These experimentations will provide us with a testbed for our models, which will enable us to adapt them to the representational needs of the real world problems, and to develop the functionalities of our representation according to this feedback.

The extensions expected, and further studies to be made, will thus concern the representation of actions (alternative, or even non-deterministic in their consequences, ...), the development of control operators not derivable from the basic ones (synchronization on the satisfaction of a condition, ...), "intelligent backtracking" for a better search in the tree of possibilities, other logical formalisms (modal logics, tri-valued logics, non-monotonic logics ...), as well in the way they are used in approaches close to ours [5], as in their application to less directly linked problems, such as distributed algorithms verification, ... The motivation of the application of logical formalisms to this kind of problems lies in their providing us with a clear knowledge representation frame as well as tools for manipulating this knowledge, i.e. reasoning. Our approach consists in applying these formalisms to simulation, as a hopefully profitable alternative to the more classical, yet difficult, plan generation paradigm [1]. The results will apply to the same issues, from the point of view of knowledge representation, and a connection can be made with temporal relations management techniques [9].

## References

[1] J.F. Allen, J. Koomen. Planning using a temporal world model. In *Proceedings of the IJCAI '83*, pages 741–747, Karlsruhe, August 1983.

[2] J.F. Allen. Maintaining knowledge about temporal intervals. *Communications of the A.C.M.*, 26(11):832–843, November 1983.

[3] J.F. Allen, P.J. Hayes. *Moments and points in an interval-based temporal logic.* Technical Report TR 180, University of Rochester, December 1987.

[4] G. André, T. Blais. Space teleoperation and control concept, experimental evaluation for the Hermes robot arm (HERA). In C.A. Mason, editor, *Proceedings of the International Symposium on Teleoperation and Control*, Bristol, England, 12-15 July 1988.

[5] G. Boy, N. Mathé. Using non-monotonic reasoning for operator assistant systems in reactive environments. In *Proceedings of the ESA-ESTEC Workshop on Artificial Intelligence Applications in Space Projects*, 15-17 November 1988.

[6] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32:333–377, 1987.

[7] R. Fikes, N. Nilsson. STRIPS : a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[8] K. Forbus. Intelligent computer-aided engineering. *AI magazine*, 9(3):23–36, Fall 1988.

[9] M. Ghallab, R. Alami, R. Chatila. Dealing with time in planning and execution monitoring. *Robotics Research, R. Bolles (Ed.), MIT Press*, 4, 1988.

[10] D. MacDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.

[11] Y. Shoham. *Reasoning about change : Time and causation from the standpoint of artificial intelligence.* PhD thesis, Yale University, December 1986.

[12] S.A. Vere. Planning in time : windows and durations for activities and goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(3):246–267, May 1983.

# AN EFFICIENT TEMPORAL LOGIC FOR ROBOTIC TASK PLANNING

Jeffrey M. Becker
Martin Marietta Astronautics Group
P.O. Box 179, M.S. 4372
Denver, CO 80201

## ABSTRACT

Computations required for temporal reasoning can be prohibitively expensive if fully general representations are used. Overly simple representations, such as a totally ordered sequence of time points, are inadequate for use in a nonlinear task planning system. This work identifies a middle ground which is general enough to support a capable nonlinear task planner, but specialized enough that the system can support online task planning in real time. A Temporal Logic System (TLS) was developed during the Martin Marietta Intelligent Task Automation (ITA) project to support robotic task planning. TLS is also used within the ITA system to support plan execution, monitoring, and exception handling.

## 1. INTRODUCTION

Most task planning systems that have been developed to date have represented the change in the truth of propositions over time within the representation used for plans [Chapman84], [Wilkins84], [Sacerdoti77], [Sacerdoti73]. Some systems have also represented temporal durations within the plan representation [Vere83]. By using a temporal logic system to support planner development the planner itself becomes conceptually much simpler. Temporal truth maintenance and duration constraint issues can be separated from planning issues. Using a temporal logic system also simplifies state projection for simultaneous planning and execution and other temporal representation problems related to plan execution. The temporal logic system described here is used within the Intelligent Task Automation (ITA) system developed at Martin Marietta. An overview of the ITA system is given in [Becker87].

To be able to formulate plans, a planner must be able to represent the effects of the proposed actions that constitute a (partial) plan. Many facilities for representing the effects of actions can be provided in a temporal logic system, but not every conceivable facility is needed to support planning. Temporal logic systems provide two primary functions: reasoning about duration constraint relations, and reasoning about the persistence of facts - also referred to as *temporal truth maintenance* [Dean87]. The combination of duration constraint relations and logical assertions is a kind of database referred to as a *time map* after [McDermott82]. Mechanisms must be provided to define time points, time intervals and the relationships between them, and to associate facts with times.

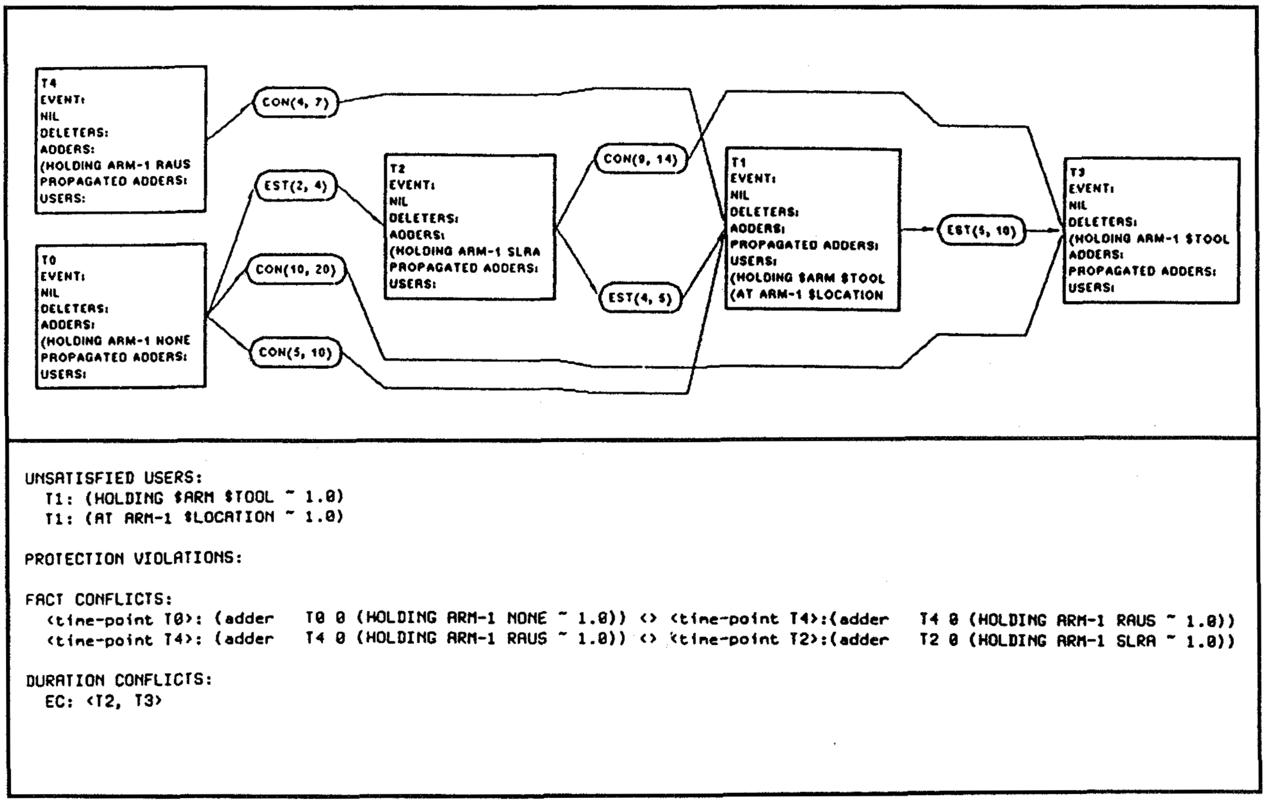
Time intervals may be specified qualitatively or quantitatively. For many planning problems quantitative duration information is needed so a quantitative representation is used in TLS. Typical operations on durations include consistency checking and deriving implied temporal relationships between time points that are not directly connected by a user-specified time interval.

This work was performed at the Intelligent Task Automation project facilities of Martin Marietta Information and Communications Systems. This work was supported by the Air Force Wright Aeronautical Laboratories and the Defense Advanced Research Projects Agency under contract F33615-82-C-5139.

The primary concern in temporal truth maintenance is managing the *persistence* of facts. A fact indexed in the time map persists until it is *clipped* at a later time in the time map. Two additional mechanisms of particular importance to planning are *protections* [Sussman 75] and *floating queries.* If a fact is protected at a certain time (or over a given time interval) then the temporal logic system will flag a warning if it becomes untrue. Floating queries are similar - if the query pattern is not matched by a fact at the time point where the query is indexed, then a flag is raised. Backward and forward chaining inference mechanisms can also be provided to support reasoning about facts asserted in the time map. It may also be desirable to represent alternative futures.

## 2.0 TEMPORAL LOGIC SYSTEM FEATURES

A time map in TLS (*Figure 2.0-1*) consists of time points, time intervals, assertions, and inconsistency records. Assertions can be declarations, adders (facts), users (floating queries), deleters (floating retractions), or rules. Assertions are indexed to time points. Time is represented as a directed acyclic graph where nodes are time points and edges are time intervals. There are two distinguished time points in the time map: *always* and *now*. Rules and declarations are indexed at *always*. *Now* is the default time used for assertions and queries. Interval durations are represented numerically by a maximum and minimum value and can indicate either an estimate or a constraint. Inconsistency records are kept for duration constraint violations, fact conflicts, unsatisfied users, and protection violations. TLS is implemented in Common Lisp and is used by invoking Common Lisp functions. TLS functions can be procedurally attached so Common Lisp functions can be called from within TLS rules as well.



```
UNSATISFIED USERS:
  T1: (HOLDING $ARM $TOOL ~ 1.0)
  T1: (AT ARM-1 $LOCATION ~ 1.0)

PROTECTION VIOLATIONS:

FACT CONFLICTS:
  <tine-point T0>: (adder   T0 0 (HOLDING ARM-1 NONE ~ 1.0)) <> <tine-point T4>:(adder   T4 0 (HOLDING ARM-1 RAUS ~ 1.0))
  <tine-point T4>: (adder   T4 0 (HOLDING ARM-1 RAUS ~ 1.0)) <> <tine-point T2>:(adder   T2 0 (HOLDING ARM-1 SLRA ~ 1.0))

DURATION CONFLICTS:
  EC: <T2, T3>
```

*Figure 2.0-1: A Typical Time Map.*

## 2.1 Logic System Language

The logic language supported by TLS consists of declarations, facts, and rules. Facts come in two flavors - functions and relations. A function, such as `color(object,red)`, can only have one value at a time. The last symbol in a function pattern represents its value. This is used within TLS for persistence clipping and to identify conflicting assertions. For example, the assertion `color(object,red)` would be clipped by `color(object,blue)` at a later time point, perhaps the result of a painting operation. This rule does not apply to relations such as `brother(Ed,John)`. Relations are handled internally as functions with boolean values, e.g. `brother(Ed,John,T)`. TLS provides syntactic sugar so that the boolean value may be omitted by the user when entering relation patterns.

Each functor (first symbol in a fact pattern) must be declared. The declaration specifies whether the functor denotes a function or a relation, the type of each argument, the value type, and procedural attachment bindings for procedures to be activated on assertions, retractions, and queries. The built-in functors include the logical connectives and a number of metalogical operations. Some example functor declarations from the domains of list and math operations are:

```
functor (member object list boolean)
functor (append list list list)
functor (+ number number number) :procedure +
```

The first example declares a predicate that could be defined using rules to determine if an object is a member of a list, the second defines a function that could be defined using rules to combine two lists into a third list. The third example declares the mathematical addition function to be procedurally attached to the Lisp + function. The Lisp function is called during queries to determine the sum of two numbers.

Besides regular fact assertions as provided in Prolog, which are called *adders* in TLS, TLS provides *user* assertions and *deleter* assertions. An adder assertion is propagated forward through the time map according to persistence rules. A user assertion is a query pattern that is indexed to a time point. Whenever the time map is modified, a record of whether the user is *satisfied* (unifies with some adder at the same time point) is updated. A user assertion can be *protected* so that a flag will be raised when it becomes unsatisfied. A deleter assertion is a retraction pattern that is indexed to a time point. A propagated adder that matches the deleter will be clipped by it. Neither users nor deleters are propagated. However, an unsatisfied user may have an associated *ghost adder* that is propagated to represent what might be true at later times in the time map if the user were satisfied. Confidence values can also be associated with adder facts and rules. Adder confidences are combined according to user-defined functions for disjunction, conjunction, and implication during forward and backward chaining.

Query operations accept a time parameter that can be a single time point or a set of time points. If a set of time points is given, then the user can specify whether the proof must hold at *every* time point in the given set, or at *some* time point. The theorem prover used in TLS uses a backward chaining algorithm as in Prolog. Two important extensions are provided. First, since facts have associated confidence values, the prover can keep track of the accumulated confidence associated with a branch of the proof search tree and prune that path if the confidence goes below some threshold. Second, relations can be declared `TRANSITIVE`, in which case the prover will check for and prune circularities. This allows the prover to handle rules for such things as transitive equality that would otherwise cause infinite loops in the proof. A limited form of forward chaining is also supported.

## 2.2 Duration Consistency Checking

A time map is a directed acyclic graph where nodes are time points and edges are time intervals. A time map is created by asserting the time points and time intervals of interest. Each time interval is associated with two time points - its start and end. There are two types of time intervals, estimated ("it takes 30 to 40 minutes to get to the ticket office") and constrained ("tickets will be available from 1 to 3 P.M. only"). Time intervals have a minimum and a maximum duration that is expressed numerically. The notation used is `EST(min,max)` for estimated intervals and `CON(min,max)` for constrained intervals.

Duration consistency checking is based on two operations on time intervals: serial composition, denoted by "&", and parallel composition, denoted by "||". Serial composition is the process of finding the most constraining interval to represent the combination of two intervals linked end to end. Parallel composition is the process of finding the most constraining interval to represent two intervals connected in parallel between the same endpoints, if a consistent composition exists. The rules for serial and parallel composition are given in *Figure 2.2-1* .

```
SERIAL  COMPOSITION:
   EST(min1,max1) & EST(min2,max2)  ->  EST ((min1 + min2), (max1 + max2))

   CON(min1,max1) & CON(min2,max2)  ->  CON ((min1 + min2), (max1 + max2))

   If ((max1 - min1) < (max2 - min2))
    then EST(min1,max1) & CON(min2,max2)  ->  CON ((max1 + min2), (min1 + max2))
    else EST(min1,max1) & CON(min2,max2)  ->  EST ((min1 + max2), (max1 + min2))
PARALLEL  COMPOSITION:
   EST(min1,max1) || EST(min2,max2)  ->  EST(min1,max1)
    consistent when: min1 = max1 = min2 = max2

   CON(min1,max1) || CON(min2,max2)  ->  CON (max (min1, min2), min (max1, max2))
    consistent when: max (min1, min2) ≤ min (max1, max2)

   EST(min1,max1)  || CON(min2,max2)  ->  EST(min1,max1)
    consistent when: min2 ≤ min1 ≤ max1 ≤ max2
```

*Figure 2.2-1: Duration Composition Rules*

*Figure 2.2-2* shows a simple time map with a duration conflict due to two parallel paths from T2 to T3. One path (a single interval) constrains the maximum duration to 14, but the other path composes to an estimate that the duration can be as long as 15. The estimates must be tightened or the constraint relaxed in order to resolve the conflict.

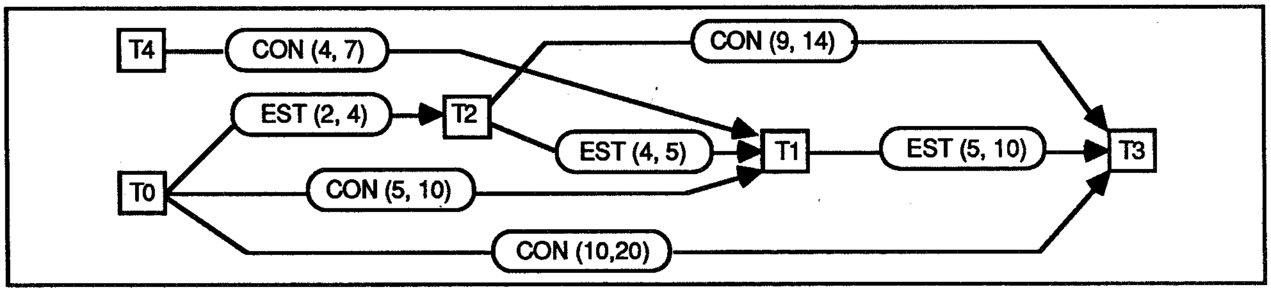


*Figure 2.2-2: Duration Consistency Example*

## 2.3 Managing Persistences

Adders are propagated through successor times in the time map until a time point with a conflicting assertion is reached. Adders can be clipped by other adders and by deleters, but not by users. Adder-adder clipping occurs when fact patterns have the same arguments, but different values. Adder-deleter clipping occurs when the patterns unify. For example, the adder `location(table,hall)` will be clipped by the deleter `location(table,$x)`, where "`$<symbol>`" denotes a pattern variable.

Two types of clipping can occur, serial and parallel. Serial clipping occurs when an adder propagates to a time point at which there is a conflicting assertion. Parallel clipping occurs when an adder propagates to a time point in parallel with a time point at which there is a conflicting assertion. The two types of clipping are illustrated in *Figure 2.3-1*.

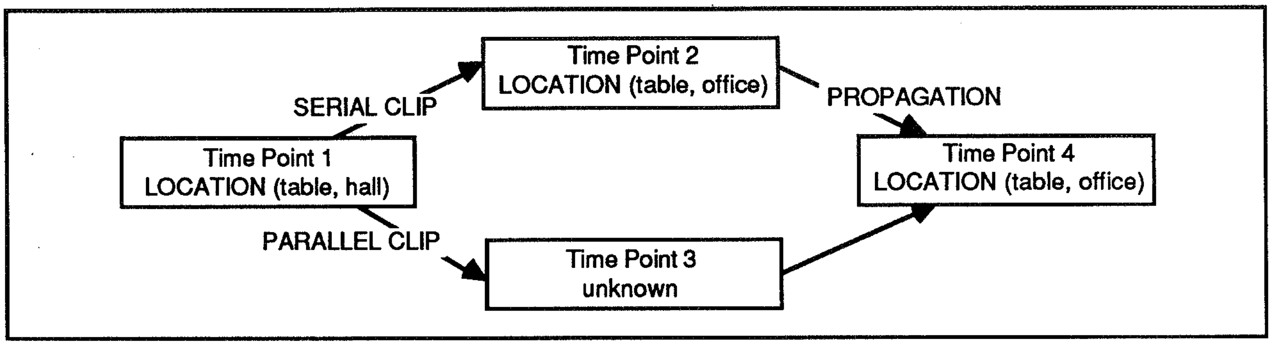


*Figure 2.3-1: Clipping of Persistences*

Fact conflicts can occur between assertions at the same or parallel time points. Fact conflicts never involve propagated adders because of the clipping mechanism. Fact conflicts can occur between any pair of adder, user, or deleter assertions. When a fact conflict involves an adder, it is depropagated (unindexed from all time points except the point it was originally asserted at) to make the time map reflect its uncertain status. When a fact conflict is found by the time map mechanism, it is recorded in a slot of the time map. *Figure 2.3-2* shows a time map with two fact conflicts, both involving the assertion `holding(arm-1,hammer)` of time point T4. Time point T4 must be ordered with respect to time points T0 and T2 in order to resolve the conflicts.

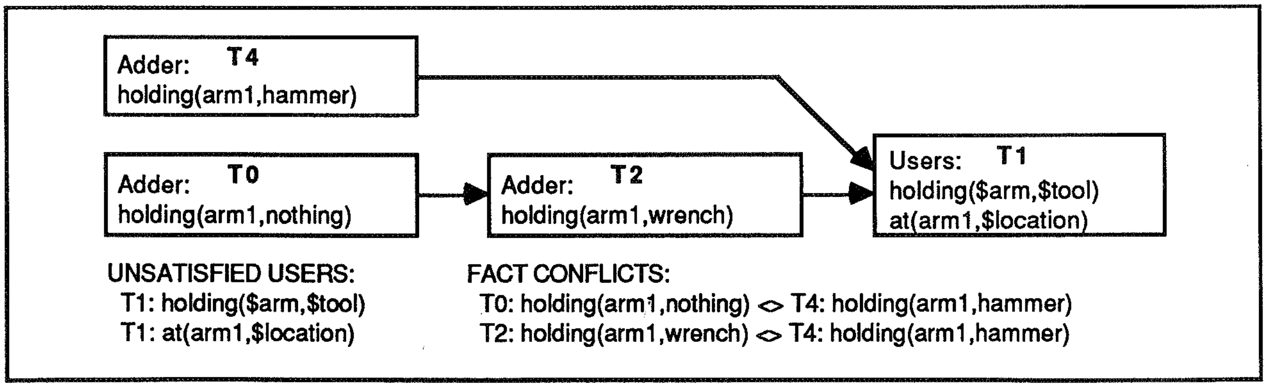


*Figure 2.3-2: Time Map With Fact Conflicts*

## 3.0 IMPLEMENTATION DETAILS

Key details for implementation of the capabilities described above will now be discussed. This discussion is not complete but does illustrate many aspects of the approach taken in the implementation of TLS.

### 3.1 Dynamic Sets

The foundation for the assertion indexing scheme and directed acyclic graph abstract data type used in TLS is an abstract data type called *Dynamic Sets*. Dynamic Sets are like the set data type defined in Pascal, except that the elements of a set type may be defined and changed as a program runs. This makes it possible to represent a set of assertions or a set of graph nodes. Set type operations include declaring a new type and adding and removing elements in the type. Set instance operations include creation, deallocation, and the usual boolean combiners (e.g. union, intersection, ...) and predicates (e.g. subset, empty-set?, ...). A set instance is a record structure with slots for: a pointer back to the set type descriptor, the current size (number of bits) of the set instance, and a bit vector for specifying members of the set instance. A set instance may be placed on an "active" list in which case it is automatically updated when an element is removed from the set type. The bit vectors of set instances are automatically grown as the number of elements in the set type increases.

### 3.2 Directed Acyclic Graphs

A directed acyclic graph (DAG) data type was defined to support operations on time maps such as interval installation and removal, finding predecessor, successor, and parallel time points, and traversals for assertion persistence computations. The DAG abstract data type is built on top of the Dynamic Sets abstract data type. Set operations are used instead of mark-and-sweep techniques for all graph operations. A graph node includes slots for: lists of immediate predecessors and successors of the node, sets of all predecessors and successors of the node, the node id, and data to be associated with the node. Updating the sets of all predecessors and successors of each node when the graph is modified requires a single traversal of the graph. Then, for example, the graph nodes between two ordered nodes can be determined by intersecting the set of all successors of the first node and the set of all predecessors of the second node. A mark-and-sweep algorithm would require marking the successors of the firt node and collecting the marked predecessors of the second node, i.e. two partial traversals. Since queries of this type far exceed graph modifications the net computation time savings is significant.

### 3.3 Assertion Indexing

The indexing scheme used in TLS is illustrated in *Figure 3.3-1*. Associated with each time point in the time map is a set of assertions - the assertions that are indexed at that time point. Each assertion is also indexed according to the elements of the pattern that represents it. Associated with each position of a pattern is a table of symbols and associated with each symbol is a set of assertions that contain that symbol in that position in some pattern. Assertions are also indexed in one of three sets according to their type: adder, user, or deleter.

Rules are indexed by their conclusion part. Finding all assertions that occur at every (some) time point in some set of time points is simply an intersection (union) operation. Finding all patterns that contain certain literals in certain positions is also done by intersecting the appropriate assertion sets. Pattern variables in assertions are also accounted for in the indexing scheme. All variables are mapped to a single symbol ($X). When looking up matches to a query pattern the union of the assertion set for data patterns with a variable in a particular pattern position and the assertion set for data patterns with the same literal in that position as the query pattern gives the set of assertions that match the query pattern at that position. The intersection of these sets over all positions gives the set of patterns that match the query pattern. This match is not equivalent to unification since variable binding consistency is not checked so unification test generally follows the initial lookup.

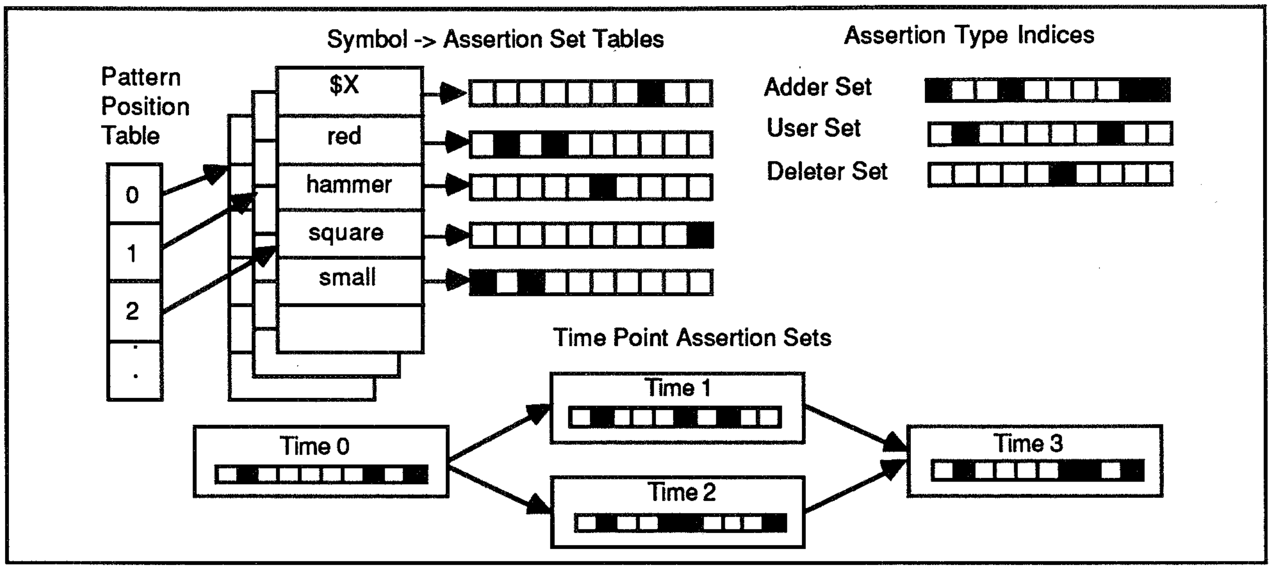


*Figure 3.3-1: TLS Indexing Scheme*

### 3.4 Duration Consistency Checking

Duration consistency checking is performed using a dynamic programming approach. This computation is non-incremental so it is possible to make many changes to the time map before checking for duration consistency. The first step is to partition the time map by removing redundant time intervals (shortcut constraint intervals with duration zero to infinity) then removing intervals with no parallel neighbors. For each remaining subgraph, paths of length 2 (two intervals) are built from paths of length 1 using serial composition. The most constrained path (MCP) of length 2 is then found between each pair of predecessor/successor time points using parallel composition. Paths of length 3 are constructed from MCPs of length 1 and 2, and MCPs of length 3 are found, and so on. This reduction builds new data structures rather than actually modifying the time map so no constraint information is lost in the process. Duration conflicts found according to the parallel composition rules are flagged on a slot on the time map.

## 4. EXAMPLE APPLICATION: THE ITA TASK PLANNER

Task level planning and plan execution functions in the ITA system use TLS as their knowledge representation substrate. As shown in *Figure 4.0-1*, the planner adds new states to a projected future view of the state of the world as a plan is constructed for a given goal. When a plan is completed, it is decomposed into a set of commands which are added to pending command queues along with additional synchronization steps. After execution of a command is completed, an entry is added to a separate history time map. Using this scheme, planning can occur concurrently with execution. If an exception occurs, planning stops and is resumed only after the exception handler has modified the current state model to correspond to the actual current state.

The task planner is a hierarchical, nonlinear planner in the same family as TWEAK [Chapman84], Nonlin [Tate77], and SIPE [Wilkins84]. It is described in more detail in [Garrett88]. When the task level of the controller receives a new set of goal conditions start and end time points for the plan to be constructed are installed in the time map after the end time point of the previous plan. The expected state of the world is automatically projected by the time map mechanisms. The goal conditions are installed as *users* at the end time point of the plan. Any goals not satisfied by conditions at the start time point of the plan will now appear on the unsatisfied users slot of the time map. First, a check is made of the conflict slots of the time map. Any conflict at this point indicates that conflicting goals were given to the planner, so the planner fails without doing any work.

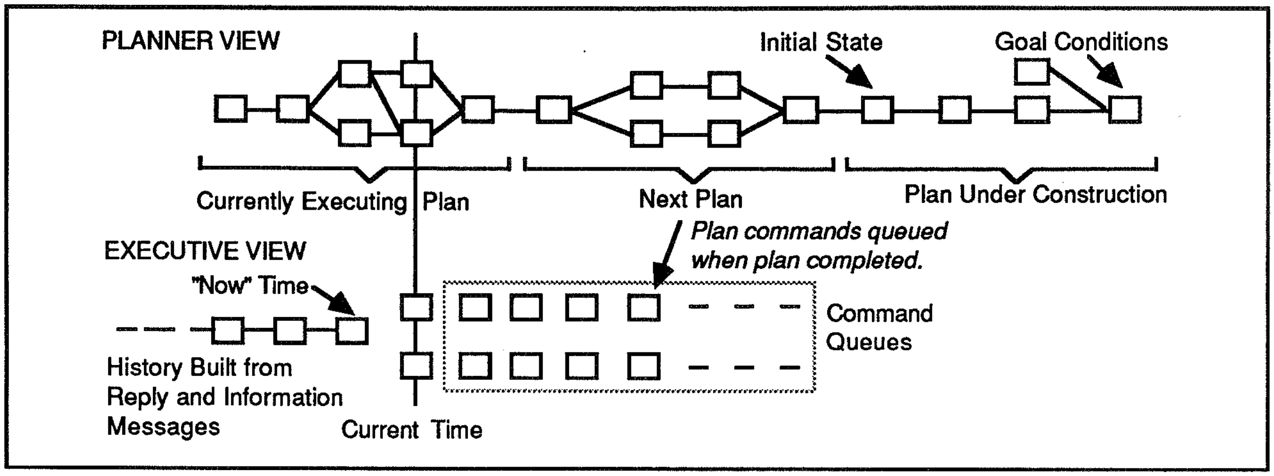


*Figure 4.0-1: Use of Time Map for Planning and Plan Execution*

The task planner finds the set of highest level unsatisfied goals and then finds the set of operators that can plan for any of the goals. The operators are ranked according to user-provided heuristics and the "best" operator is installed in the time map by creating a time interval that represents the start, end, and duration of the operator. Additional intervals are asserted so that the operator is constrained to occur within the plan and before all operators whose subgoals it plans for. Goals that the new operator plans for are protected so that the time map will indicate when those goals are *clobbered.* Goals that are satisfied but not explicitly planned for are not protected. When these goals are clobbered they are said to be *reactivated.* Preconditions of the new operator are installed as *user* assertions in the time map at the start time point of the operator. Postconditions are installed as *adder* and *deleter* assertions in the time map. If the fact conflict slot of the time map is not empty, the planner uses heuristics to choose a preferred ordering and installs additional intervals in the time map. Backtracking choices are recorded for alternative operators and orderings. Backtracking occurs when a planned-for goal is clobbered, or when no satisfactory operator or ordering can be found.

## 5. RELATED WORK

Much of the work done on temporal reasoning has been concerned with computing closures of temporal relations expressed in a qualitative interval logic [Allen83], [Vilain86], [Ladkin88]. However, TLS is closest in spirit to the Time Map Manager (TMM) system described in [Dean87]. Durations in TLS are represented numerically rather than qualitatively as in TMM. For many real world problems, such as travel planning, the ability to represent durations numerically is a necessity. With a numerical representation consistency checking can be performed without computing the closure of implied duration relationships between all time points thus avoiding exponential computation time. The consistency checking mechanism used in TLS is based on a dynamic programming rather than a propagation approach as in TMM. An advantage of the dynamic programming approach is that many changes can be made to the time map before checking the constraints again. This can result in a savings in computation time. In addition, TLS supports *estimates* as well as constraints in its representation of duration. In [Brooks82] a similar though more powerful mechanism is applied in the domain of geometric error analysis for robot planning.

Assertions in TLS are indexed to time points as in TWEAK [Chapman84] rather than to time intervals as in TMM - this is primarily an implementation detail since the two methods are conceptually equivalent. The modal truth criterion defined by Chapman is not fully supported by either TMM or TLS since neither supports representation of *possible* persistence of assertions. The deductive proof mechanism used in TLS is similar to that used in Prolog [Clocksin84] with extensions for proof subtree pruning based on recurring goals in the proof tree as described in [Smith85]. TMM provides more complex mechanisms to support temporal imagery than TLS which only provides simple forward chaining. Some ideas about inheritance between theories used in TLS are based on experience with MRS [Genesereth84].

## 6. CONCLUSIONS

A temporal logic system that was implemented to support task level planning and plan execution in a hierarchical robot controller has been described. The system is efficient enough to support online real-time planning. Some features have been omitted that might be desirable for supporting other applications. Careful consideration is being given to possible extensions that could be implemented without significantly degrading performance. Other applications of TLS are being investigated, such as causal modeling of electrical power distribution systems to support fault diagnosis. The availability of a temporal logic system provides not only a new way of thinking about how to build a planning system, but provides a new way of thinking about solving many different kinds of problems.

## ACKNOWLEDGEMENTS

Special thanks to Fred Garett and Don Mathis for discussions that lead to a number of design decisions in TLS. Fred was also the first user of the system and uncovered many bugs that otherwise would have been missed. Thanks also to Dennis Haley (ITA Project Manager) and Roy Greunke (ITA Software Lead) for shielding us from distractions during the development of this system.

## REFERENCES

[Allen83] Allen, J., and Kooman, J., "Planning Using a Temporal World Model", *Proceedings IJCAI-83*, pp. 741-747.

[Becker87] Becker, J., and Garrett, F., "An Architecture for Intelligent Task Automation", *Proceedings AAAI-87*, 1987, pp. 672-676.

[Brooks82] Brooks, R. A., *Symbolic Error Analysis and Robot Planning*, A.I. Memo No. 685, Massachusetts Institute of Technology Artificial Intelligence Laboratory, September, 1982.

[Chapman84] Chapman, D., *Planning for Conjunctive Goals*. MIT Technical Report 802, January, 1984.

[Clocksin84] Clocksin, W. F., and Mellish, C.S., *Programming in Prolog*, Springer, Berlin, 1984.

[Dean87] Dean, T., and McDermott, D., "Temporal Data Base Management", *Artificial Intelligence 32*, 1987, pp. 1-55.

[Garrett88] Garrett, F., and Becker, J., Task Level Planning in the Intelligent Task Automation System, publication pending - available on request from authors.

[Genesereth84] Genesereth, M., Greiner, R., Grinberg, M., and Smith, D., *The MRS Dictionary,* Heuristic Programming Project Report No. HPP-80-24, Stanford University, Stanford, CA, January 1984.

[Ladkin88] Ladkin, P. B., "Satifying First-Order Constraints About Time Intervals", *Proceedings AAAI-88*, 1988, pp. 512-517.

[McDermott82] McDermott, D.V., "A Temporal Logic for Reasoning about Processes and Plans", *Cognitive Science 6*, 1982, pp. 101-155.

[Sacerdoti73] Sacerdoti, E., "Planning in a Hierarchy of Abstraction Spaces", *Proceedings IJCAI-73*, 1973, pp. 412-422.

[Sacerdoti77] Sacerdoti, E., *A Structure for Plans and Behavior,* North-Holland, New York, 1977.

[Smith85] Smith, D. E., Genesereth, M. R., and Ginsberg, M. L., *Controlling Recursive Inference*, Report No. STAN-CS-85-1063 (also numbered HPP-84-6), Stanford University, Stanford, CA, June 1985.

[Sussman75] Sussman, G. J., *A Computer Model of Skill Acquisition*, American Elsevier, New York, 1975.

[Tate77] Tate, A., "Generating Project Networks", *Proceedings IJCAI-77,* 1977, pp. 888-893.

[Vere83] Vere, S., "Planning in Time: Windows and Durations for Activities and Goals", *IEEE Trans. Pattern Anal. Mach. Intell. 5*, 1983, 246-267.

[Vilain86] Vilain, M., and Kautz, H., "Constraint Propagation Algorithms for Temporal Reasoning", *Proceedings AAAI-86*, 1986, pp. 377-382.

[Wilkins84] Wilkins, D., "Domain-Independent Planning: Representation and Plan Generation", *Artificial Intelligence* 22, 1984, pp. 269-301.

# The Indexed Time Table Approach for Planning and Acting

**Malik Ghallab and Amine Mounir Alaoui**
LAAS-CNRS
7, Av Colonel Roche, 31077 Toulouse, France

**Abstract:**

We are interested here in a representation of symbolic temporal relations, called **IxTeT**, that is both powerful enough at the reasoning level for tasks such as plan generation, refinement and modification, and efficient enough for dealing with real time constraints in action monitoring and reactive planning.

Section 1 argues that such representation for dealing with time is needed in a teleoperated space robot. After a brief survey of known approaches, section 3 introduces the proposed representation, and shows its computational efficiency for managing a large data base of temporal relations. Reactive planning with IxTeT is described in section 4 and exemplified through the problem of mission planning and modification for a simple surveying satellite.

## 1. Introduction

This paper addresses the problem of real-time acting and reacting for robot in a dynamic environment, at the level of representation, reasoning, and decision making aspects.

Such problems arise in robotics applications that are either too complex and changing to allow complete and reliable hand programming, or that require some level of autonomy. Most space robotics applications are in both cases. Applications in *unstructured environments*, such as robots for planetary exploration, will require a high level of autonomy in order to cope with a large set of tasks in a broad spectrum of conditions. Robots in *structured environments*, such as space stations, should also be versatile, even if less autonomous. In both cases, teleoperation is complementary to and does not contradict decision making abilities, intelligence, and autonomy in tasks ranging from sensing and environment interpretation to planning and acting. This is more true for real-time tasks such as reacting to unexpected events. Indeed, manned control of a space robot should be thought of mainly as providing the system with goals, eventually decomposed into a flexible structure of subgoals and tasks, that are programming the robot at the task level.

Plans, either given or autonomously generated, need refinement or revision at execution time. Conditional plans involve choices between alternatives, eventually only one of which (the most likely one) has been fully pursued at planning time but another one may need to be considered at execution time. Unexpected events may require partial modification or rejection of the current plan. Goals may have to be discarded or postponed for more urgent ones.

Thus in addition to planning, acting and reacting means: monitoring and keeping track of the state of achievement of current plan, refining adequately actions and implementing them, choosing from alternatives, deciding about short term reactions to unexpected events. Those tasks involve mainly 2 temporal aspects: how one represents time and reasons on temporal relations (e.g. for keeping track of the present time), and how one deals with real-time.

Any action involves time as a particular resource. A good representation should exhibit the rich temporal structure relating an action to its effects, and to expected events and pursued goals. In that sense, monitoring is keeping track of the present time relatively to the projected future. The advance of time is discontinuous and driven by asynchronous events (although a robot may have a clock that gives it the absolute time). The happening of an expected event instanciate one particular future among the alternatives planned.

The real-time requirement is mainly due to the dynamic nature of the environment, *i.e.* to unexpected changes and events that happen asynchronously, are not under the robot control, but that require from it adequate reactions. Real-time reaction to asynchronous events means an a priorily bounded response time, at least for the first of a hierarchy of actions, such as:

- immediate reflex action,
- short term adaptation to the situation (for assessment, goals evaluation and replanning),
- long term reaction according to the new plan.

For the problem considered we thus need a representation of time and temporal relations that is both powerful from the reasoning point of view and efficient from the computational point of view.

After a brief survey of known approaches, section 3 introduces the proposed representation, called IxTeT, and shows its computational efficiency for managing a large data base of temporal relations. Reactive planning with IxTeT is described in section 4 and exemplified through the problem of managing a simple surveying satellite.

## 2. State of the Art

Classical situation calculus and state-space representations consider actions as instantaneous transitions. They cannot represent time or forthcoming events not resulting from the planner's activity.

Procedural or tasks networks [11][13] suffer from the same limitations, although they deal with partially ordered flexible plans. Some extensions, such as that of DEVISER [15], add a numerical representation of time, like durations and bounds (windows) on activities, that are managed by a mixing of Operations Research (e.g. PERT-chart) and AI techniques. However symbolic temporal representations are not allowed by such approaches, they cannot deal with relative relationships (e.g. between goals, events, actions and their effects), and hence offer very restricted reasoning capabilities.

Three approaches can be used for a symbolic representation of time:
- time as a term of a predicate in classical logic: a limited representation;
- time as a modality in temporal logic: seems to be very complex for planning tasks;
- time as an element in couples <logical formula F, temporal qualification of F> : this "reified logic" approach [9], [3], and its extensions [11], was found to have some nice properties. It manages separately the temporal qualifications through a so-called *Time-Map Manager* (TMM) that is in charge of retrieval and updating of a knowledge base of temporal relations. In a typical application, a TMM will be put at a very low level and in heavy use, it has to be very efficient.

Of this last approach, the *Algebra of Temporal Interval* [1] is the most popular representation: it is appealing for its ease of implementation and expressive power in planning tasks [2]. It has however a major drawback: the consistency problem for a set of Interval Algebra relations was proven to be NP-Hard [16]. But of resorting to exponential algorithms, this leads to the use of a transitive closure propagation algorithm that is defective because of:
- a completeness problem: it may accept an inconsistent set of relations as being consistent; and
- a complexity problem: it runs in $O(n^3)$, a too high complexity for large applications.

As it was argued in [16] one can solve the completeness problem of this algorithm by restricting the expressive power to a sub-class of Interval Algebra. That class is equivalent to the *Time Point Algebra.* For those two representations we are proposing a complete and much more efficient method than the transitive closure propagation algorithm.

## 3. Managing a Time Map with IxTeT

A natural representation for a set of temporal relations is a network where nodes are time tokens (i.e. intervals or instants) and arcs are labeled by the constraints relating two nodes. Two directions can be pursued for performing the two tasks of:

(i) retrieval (whether and how two events are related) and
(ii) updating (add new events and temporal relations)

- either using a complete graph where all possible relations between all pairs of nodes are propagated and explicitly maintained: this makes retrieval trivial in O(1), and requires a costly propagation algorithm in $O(n^3)$ for the updating task;
- or using a network where the only arcs are those of the explicit knowledge of the problem: this simplifies updating but requires for retrieval a costly search through possible paths of the network.

The approach proposed here is a trade-off between these two directions. It relies on the efficient combination of 2 principles:
- adding to the time-network a particular data structure, a maximal spanning tree with an adequate indexing scheme, that permits the computation of ancestral information in O(1), this greatly simplifies task (i), and

- restricting the propagation of new relations to a small subset of nodes in the network in order to perform task (ii) efficiently.

In a time-point algebra 3 elementary relations, *before*, *equal* and *after*, and their 5 disjunctive combinations relate a finite set of instants or time-points. Instead of a network with arcs labeled by relations, we use 2 different types of unlabeled arcs:
- arc $\langle$ standing for the relation (*before* or *equal*), and
- arc $\neq$ meaning the relation (before or after).

The 8 possible relations between 2 time-points are easily expressed as 0, 1 or 2 arcs relating two nodes.A network of $\langle$ and $\neq$ arcs corresponds to a consistent set of relations if no pair of nodes, connected by a $\neq$ arc, are involved in a loop through $\langle$ arcs. Such a loop describes a set of identical time-points that should be collapsed to a single node. Individual events corresponding to this set are kept distinct but their simultaneity is recorded by connecting all of them to the same node in the time-map. Arcs $\neq$ do not require any propagation mechanism; they are looked for only when a collapsing decision has to be taken. For that reason we can keep arcs $\neq$ implicit in the network representation.

A consistent network where all possible collapsing operations have been performed contains only $\langle$ arcs and is loop-free. It thus defines a partial order over the set of nodes. Since we can always add for convenience an origin time-point, we endup finally with a network that is a rooted DAG, *i.e.* a time-lattice.

Let us denote it $L=(U,A)$ where: $U=\{t_0, t, u, v, w, ...\}$ is the set of time-points, $t_0$ being the root of L; and A is the set of $\langle$ arcs in L.
Point u precedes temporally (is *before* or *equal*) point v if there is a path in L going from u to v. Let us denote $u \ll v$ this fact ($\ll$ is the transitive closure of $\langle$ ). Thus relating 2 points requires a search of a path in L . How can we speed-up such a search?

### 3.1. Representation

To speed-up this search we use the fact that ancestral information can be computed in constant time for a tree correctly ordered. Two problems should be addressed: (1) classical tree ordering is not easily updated and maintained for a dynamically growing structure, and (2) how to map a time-lattice to a tree.

We solve this last problem by extracting from L a maximum spanning tree T defined as follow:
- T is rooted at $t_0$ (it is not a free tree as is usually the case for a spanning tree), and covers all nodes of L;
- the number of arcs in T is maximal.

Let us denote by r(u) the rank of u in L, *i.e.* the length of the longest path in L from $t_0$ to*u*:

$r(u)= 1 + \max\{r(v)/ \forall(v,u) \in A\}$, with $r(t_0)=0$. To compute T from L we first order the nodes in L according to their rank. This can be achieved by an $O(|A|)$ breadth first search in L: all successors of nodes of rank k have their rank set to k+1, which may change previously computed ranks if longer paths are found (some simple additional tests speed-up the procedure).

Let M be the maximal rank found in L. We start from any node z of rank M, put it in T, choose among its predecessors in L any node y of rank M-1 and put in T the arc (y,z) and the node y as the parent node of z: p(z)=y. This is repeated for a predecessor x of y such as r(x)=r(y)-1; arc (x,y) and node x are added to T. We keep on moving up along a path of maximal length until the root $t_0$ is put in T. The procedure is repeated starting from a node of maximal rank among those not already in T. While processing node u if there is a choice between several of its predecessors, all at rank r(u)-1, we choose one not already in T and add it to T. If none remains we choose among such predecessors one in T which has the least number of children in T, and attach a new path to the spanning tree. The procedure is repeated until all nodes of L are put in T.

Let s(u) be the set of children of u in T, and s*(u) the set of its descendants in T (transitive closure of s). To test whether $v \in s^*(u)$ efficiently we attach to each node u as index a sequence I(u)=(i1 i2 ... ik) of one or more integers that is defined, while generating T, as follows:
- nodes of the path ($t_0$, ..., x, y, z) that was first put in T are indexed by their rank:
I(z)=(M), I(y)=(M-1), I(x)=(M-2), ... , I($t_0$)=(0);

- if $I(u)=(i1\ i2 \ldots ik)$ and $v \in s(u)$ then:
  - if $|s(u)|=1$ (v is the first children of u in T) then $I(v)=(i1\ i2 \ldots i(k\text{-}1)\ (ik+1))$
  - if $|s(u)|=2$ then $I(v)=(i1\ i2 \ldots ik\ 1)$
  - if $|s(u)|>2$ then $I(v)=(i1\ i2 \ldots ik\ \ (3 - |s(u)|)\ \ 1)$

Nodes of the first path put in T can be indexed while they are added to T. The other nodes are not indexed until their path is attached to T: indexing proceeds by moving from the attachment parent (already indexed) down along the path. Notice that the indexing is easily maintained for a dynamically growing structure: the addition of a new node (as a leaf) in the tree does not change the indexing of the previous node.

The main property of this indexing scheme is the following:
if $I(u)=(i1\ i2 \ldots ik)$, and $I(v)=(j1\ j2 \ldots jh)$ then

$$v \in s^*(u) \ \text{ iff } \ k \leq h\ ,\ (i1\ i2 \ldots ik\text{-}1)=(j1\ j2 \ldots jk\text{-}1) \ \text{ and } \ ik \leq jk \qquad (1)$$

Thus, to relate 2 nodes u in v in T we first compare their rank
- if $r(u)=r(v)$ then u and v are not related in T;
- if $r(u) < r(v)$ : either condition (1) is satisfied: v is a descendant of u, or they are not related;
- if $r(u) > r(v)$ : u and v are permuted before checking condition (1).

Notice that the rank of a node is given by its index : $r(u)=\Sigma u_i$ ; for i=1 to k, and $u_i > 0$

A precedence relation in L, such as $u \ll v$ , can be retrieved either
- through the spanning tree T if $v \in s^*(u)$ : this is checked easily; or
- through a path using some arcs of L not belonging to T.

To take care of this last case, arcs not in T are processed by 2 operations:
(1) Eliminating redundant arcs: if $(u,v) \in A$ is an arc of L not belonging to T such as $v \in s^*(u)$ then this arc does not bring any useful information. It is redundant and can be eliminated.
(2)Propagating non redundant arcs: 2 nodes may be linked by a path that mixes in any order arcs from T and non redundant residue arcs. To avoid mixing the 2 structures and simplify the retrieval of precedence relations we propagate recursively a non redundant arc (u,v) to the parent node of u in T, unless v is already a descendant of p(u) in T, *i.e.* if $v \in s^*(p(u))$.

The procedure corresponds to the trade-off mentioned earlier between using a complete graph and keeping a minimal set of relations. Let us call *residue arcs* the set obtained after elimination and propagation.The important property here is that any residue arc (u,v) is such that $r(u) < r(v)$. This is trivially true for arcs in L not belonging to T, it is also true for added arcs since the propagation goes only upward in T.

### 3.2 Algorithms

To make clear the distinction between the part of the time-lattice L covered by the spanning tree T and the residue part:
- in T we will speak of the *parent* p(u) of a node u, its *children* s(u), its *descendants* $s^*(u)$ and its *ancestors*; I(u) is its index and r(u) its rank; only s, p and I are kept as data structures.
- in the residue part, a(u) denotes the set of nodes linked to u by residue arcs $\langle$ .We will speak of the *followers* and *foregoers* of a node for adjacency relations defined by such arcs (reserving *successors* and *predecessors* for the complete lattice).

Notice that $s(u) \cap a(u)=\varnothing$: the successors of a node are partitioned into its children and its followers.

#### 3.2.1. Retrieval

A precedence relation in L is characterized by:

$$\begin{aligned} u \ll v \Leftrightarrow \quad & (r(u) < r(v)) \\ & \wedge\ [\ (v \in s^*(u)) \vee (v \in a(u)) \vee (\exists\ w \in a(u) \,/\, w \ll v)\ ] \end{aligned}$$

The 3 first conditions come from the fact that u precedes v in L if v is a descendant of u in T or if it is a follower of u. The last one is due to the property of the upward propagation mechanism: all followers of the children of u are either the followers of u or its descendants; there cannot be a non descendant node of u linked to u through $s^*(u)$ that is not also linked to u through a(u).

The comparison algorithm between two points u and v is thus:

```
**Compare** (u, v)
    If $r(u)=r(v)$ then return(nil)
    else   if $r(u) > r(v)$ then **Relate** (v, u)
           else **Relate** (u, v)

**Relate** (u, v)
    if $v \in s^*(u)$ or $v \in a(u)$ then return $(u \ll v)$
    else   for each $w \in a(u)$
               if [$r(w) < r(v)$ and **Relate**(w, v) return $(w \ll v)$
                   then return $(u \ll v)$
           return nil
```

Notice that the recursive calls to Relate are pruned when the rank of w reaches that of v

### 3.2.2. Updating

Adding new points and relations should be done such as to keep all the properties of the representation. The addition a point *w* and 2 relations with *u* and *v* ($u \langle w \langle v$) can be decomposed into 2 steps:
- add *w* as a child of *u* and give it the right index ; and
- add an arc between *w* and *v* and update the tree and residue arc if necessary.

The first operation is straightforward. The second operation involves 3 steps: a test, and eventually a propagation and a reindexation.

The test determines whether $v \ll w$, in this case the updating is impossible ($w \langle v$ can be inserted) unless all points in every path from *v* to *u* can be collapsed.

The reindexation takes place if $r(w) \geq r(v)$. In this case, to keep *v* on the longest path in the spanning tree, we give to *v* a new parent node *w*. Node *v* is removed from the children of p(v) and put as a follower of p(v); p(v) becomes *w*; the descendants of *v* are reindexed. The reindexation procedure computes the new index of each node according to the index of *v*, it then verifies if the followers of *v* have the right rank considering the new index of *v*, and, if not, it reindexes them. This is repeated recursively.

If $r(w) < r(v)$, v is put as a follower of *w*. This residue arc is propagated to the parent of *w*, and recursively to its ancestors that are not found by procedure **Relate** linked to *v*. Notice that if there is a reindexation, there will be propagation of the arc between the old parent of *v* and *v*: all the former ancestors of *v* have to know that they are still linked to it.

## 3.3. Performances

Reindexation and backward propagation procedures are detailed in [5], together with the 2 other tasks performed by the TMM : collapsing and removal. This reference also reports on experimental results for a set of time lattices of size up to 2000 points. On 60000 runs the IxTeT Time-Map Manager exhibits a linear complexity for both operations, retrieval and updating. The linearity constant is fairly low: in a 2000 points lattice about .5 second is required to insert a new point and 2 relations and less than .05 second to compare 2 points (for a non optimized Lisp implementation on a Sun3). The space complexity of IxTeT seems also to grow linearly.

IxTeT abilities for dealing with symbolic temporal knowledge can be extended to numerical quantitative constraints and to constant points (dates). All numeric relations and dates may be given with intervals precising earliest and latest possible occurrence. It is easy to combine in IxTeT an absolute referencing system by dates with qualitative relations. A point can be attached to a date. Arcs can be labeled by durations. The direct link that exists between durations and dates allows several deductions on the order of points in the lattice. A lattice may have some points known as precise dates, others are variables linked with qualitative relations, and others have dates deduced from quantitative relations.

## 4. Planning with IxTeT

The*IxTeT*representation for planning relies on a 2-dimensional array with rows corresponding to logical assertions and columns to time points (instants). Cells in the table are temporal qualifications of the assertions. Instants are related through a time lattice that is managed as described earlier.

The user defines in a declarative way a set of decomposition operators, each one being a description of the steps needed to achieve an elementary task. This description gives recursively the actions, subtasks, conditions required, and goals achieved by the task, together with their temporal links. Those are stated as symbolic relations over *implicit intervals* using elementary relations of interval algebra (start, meet, finish, overlap, before, during equal and their inverse). Indeed, it is preferable and easier to express input temporal knowledge in terms of intervals, and to transform it for efficiency reasons into time points. Actions are defined separately from the operators through their effects, temporally linked to the action. One may specify an action with a duration and effects that take place when the action starts, when it is going on, when it finishes, or effects that are later delayed.

By actions, we mean here the elementary actions that cannot be divided into smaller parts. Of course, each action may have several arguments that define its possible parameters. So the description of an action will be:

```
(Describe (Action arg$_1$ arg$_2$ ... arg$_n$)
          (Effects (effect$_1$)
                   (effect$_2$)
                   ...
                   (effect$_k$))
          (Duration (d)))
```

Each effect in the description is, in fact, the name of one of the effects induced by the action preceded by a temporal relation defining precisely when the effect takes place *vs* the action. As we said, the temporal relation is one of the 13 relations of interval algebra. The duration may be defined by a constant number or by a function of the arguments. A **When** field may be added to the action to define the context when the action can be performed.

When elementary actions are defined, it is possible to combine them into operators that achieve a task. A task will be a combination of several actions linked by temporal relations and some time constraints. Time constraints are here to define the environment in which the task can be realized. All this is defined as follows:

```
(To Achieve (Task arg$_1$ arg$_2$ ... arg$_n$)
          (TimeCond (TC$_1$)
                    (TC$_2$)
                    ...
                    (TC$_k$))
          (Do Steps
                    s$_1$ (Action$_1$)
                    s$_2$ (Action$_2$)
                      ...
                    sp (Actionp))
          (Such That (TempRel$_1$)
                    (TempRel$_2$)
                    ...
                    (TempRel$_q$)))
```

The **TimeCond** defines all the temporal constraints of the task: it cannot begin before certain effects are verified, or it has to keep something true during the whole executiön, etc... The constraints are defined by one or more of the 13 interval relations followed by an effect. If there is more than one relation, the set of relations is disjunctive and taken in the subset of restricted interval algebra that is compatible with the time points algebra.

The **Such That** field defines the temporal relations that link the actions performed in the task. With this, the task is totally defined with its constraints and the actions it takes (and so, the effects it induces).

The implicit intervals in this user-defined knowledge and their relations are automatically translated into a lattice of time points. This is done by *compiling* each task operator into a general *IxTeT* (*i.e.* assertions have quantified variables that are consistently related). A plan with such representation is an instantiated *IxTeT*.

The *compilation* is an off-line preparation for the planner: this operation creates the time lattice corresponding to each task, attaching to the beginning or the end of effects or actions a time point. The lattice is easily created by the temporal constraints contained in the task, and its coherence is tested (for instance, an action cannot have two beginnings and no end). So the result is in two parts: a table giving for each action and effect its beginning(s) and end(s), and a time lattice relating those points.

This preprocessing work simplifies the work of the planner: it has not to build all the lattices and to test their coherence, it has only to work with ready parts to build the plan. The *compiled* task can be inserted in any plan (represented itself by a lattice and a table) just like a brick in a wall. This is very important because it allows fast insertion of tasks in a reactive system: if the system is fast enough, it is possible to change the plan during its execution according to the changes of external world.

Planning, along the hierarchical goal decomposition approach proceeds by attaching tasks to goals and subgoals. This corresponds here to the insertion (and instanciation) of new ***IxTeT***s into the current plan. Controlling the plan execution while acting is performed by progressively reducing the time-lattice to an ordered sequence as time advances. This results either from an opportunistic choice made at execution time, or through the observed occurrence of an expected event. Reacting to unexpected events on the other hand involves considering new subgoals, *i.e.* inserting new ***IxTeT***s into the projected future, and reconsidering the need of previously planned actions an goals.

Let us illustrate some of the planning processes with IxTeT through the example of managing a simplified earth observation satellite similar to SPOT[8]. There are two cameras on board that may be used either each one alone or in conjunction. Each camera may have its mode (panchromatic or multispectral) and its focal distance changed and a mirror in front of it moved such as to aim at a particular direction.

On board, there are two recorders that allow images to be kept if necessary until they are sent to earth. Each recorder may be freely used by any of the two cameras. And there is also a radio system allowing reception of requests from earth and sending of images.

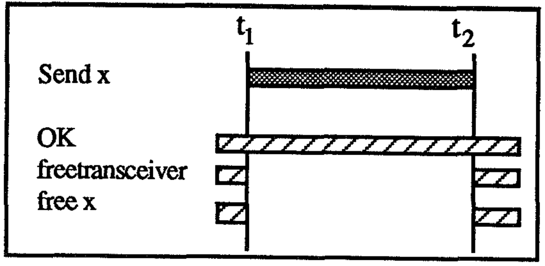
Images of a region on earth can be taken only under certain conditions. These conditions can be summarized by parts of the trajectory in which photographs of the region can be taken. So the whole trajectory of the satellite will be cut into numbered parts.

In the IxTeT formalism, we can attempt now to represent some of the most important actions and tasks of satellite:

**Elementary Actions**

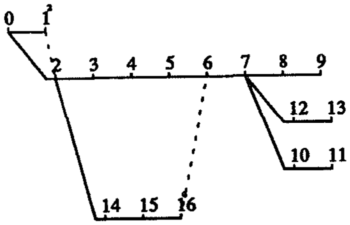
```
(Describe (ChangeFoc x a b)             ; Change focal dist. of camera x
        (Effects                        ; from an initial value a to b
                (mi foc x a)
                (m foc x b))
        (Duration (+ 10 (* 5 (abs (- b a)))))       ; Notice that duration is a function here

(Describe (LockFoc x)                   ; Lock focal distance of camera x
        (Effects  (m lockedfoc x))      ; to its current value
        (Duration 1))                   ; Here, duration is a constant

(Describe (UnlockFoc x)                 ; Unlock focal of camera x
        (Effects
                (mi lockedfoc x))
        (Duration 1))

(Describe (ChangePos x a b)             ; Change position of mirror x
        (Effects                        ; from an initial position a to b
                (mi pos x a)
                (m pos x b))
        (Duration (+ 10 (* 5 (abs (- b a)))))

(Describe (LockPos x)                   ; Lock position of mirror of camera x
        (Effects  (m lockedpos x))      ; to its current value
        (Duration 1))
```

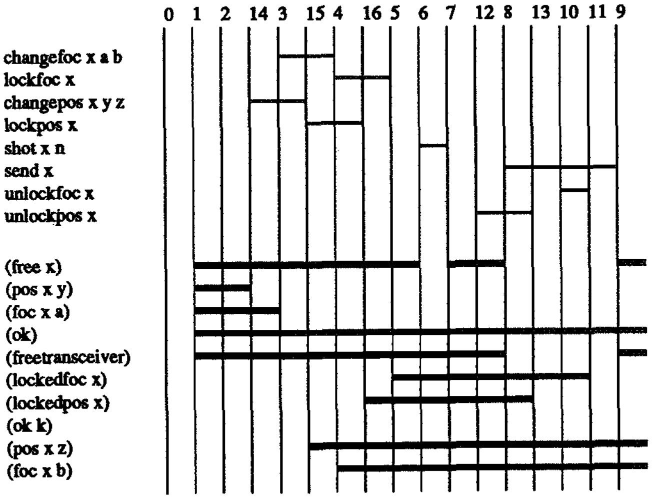
```
(Describe (UnlockPos x)           ; Unlock position of mirror of camera x
        (Effects
                (mi lockedpos x))
        (Duration 1))

(Describe (Record x y)            ; Record image in camera x in
        (Effects                  ; recorder y
                (mi free x)
                (mi free y)
                (m free y)
                (m free x))
        (Duration 10))

(Describe (Shot x n)              ; Get images from camera x during
        (Effects                  ; a period n (the mode may be added)
                (mi free x)
                (m free x))
        (Duration (+ 15 (* n 5))))

 (Describe (Send x)      ; Send image from recorder or camera x
    (Effects
       (d OK)
       (mi freetransceiver)
       (mi free x)
       (m free x)
       (m freetransceiver))
    (Duration 100))
```



**Tasks performed by the satellite**

```
(To achieve (ShotAndSend k x n a b y z)   ; Get images with camera x during n
        (TimeCond                         ; and from the trajectory part k
                (When free x)             ; with focus b and mirror position z
                (After lockedfoc x)       ; Then send the image
                (After lockedpos x)
                (While OK k))
        (Do Steps
                s1 (changefoc x a b)
                s2 (lockfoc x)
                s3 (changepos x y z)
                s4 (lockpos x)
                s5 (shot x n)
                s6 (send x)
                s7 (unlockfoc x)
                s8 (unlockpos x))
        (Such That
                (m s1 s2)
                (m s3 s4)
                (< s4 s5)
                (< s2 s5)
                (< s5 s6)
                (< s5 s7)
                (< s5 s8)))
```

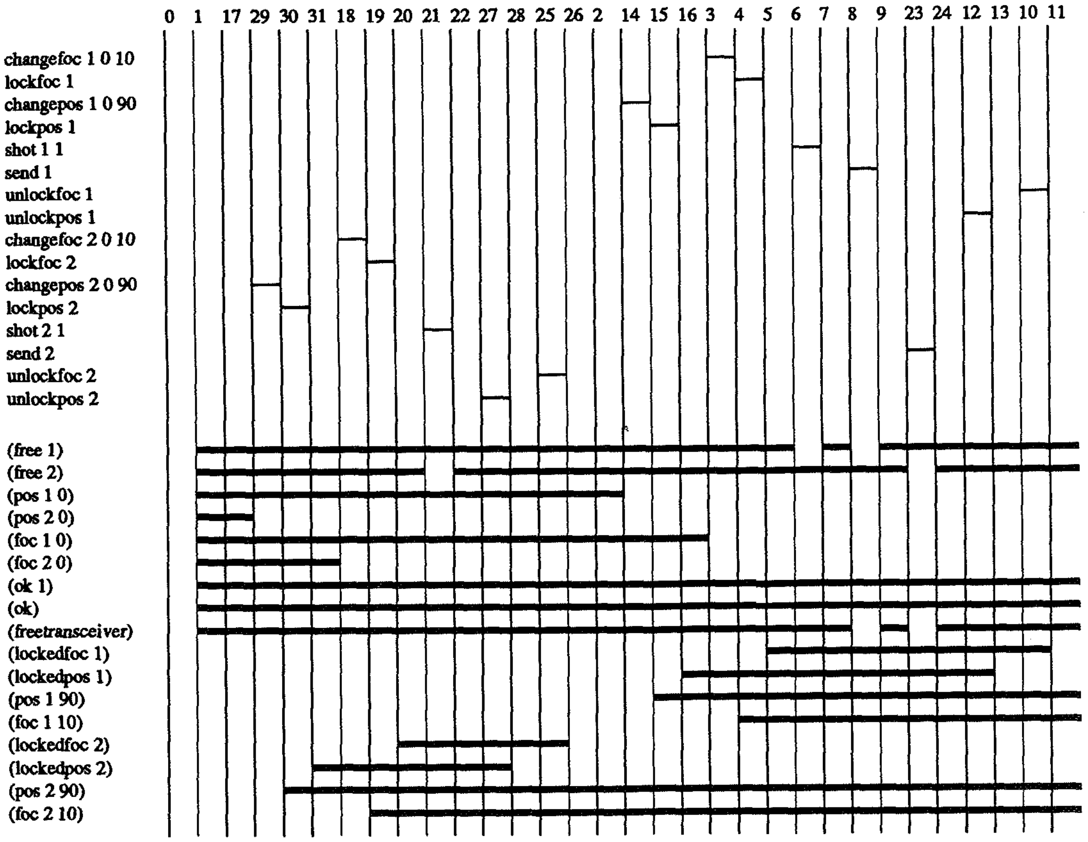


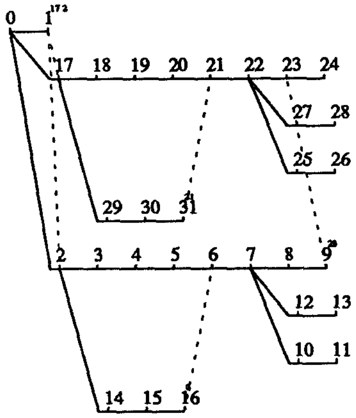


Notice that the **TimeCond** field contains time constraints other than those of interval algebra. This is because it is much more simple to have the single word **When** than (mi oi f d). The disjunction (mi oi f d) is a constraint which is compatible with the time points algebra [6].

Notice that all the constraints are relative, but durations and some absolute dates may give to the time lattice other constraints that are not incompatible with the first ones. In fact, a mixed system with absolute and relative constraints is completely representable with this formalism.

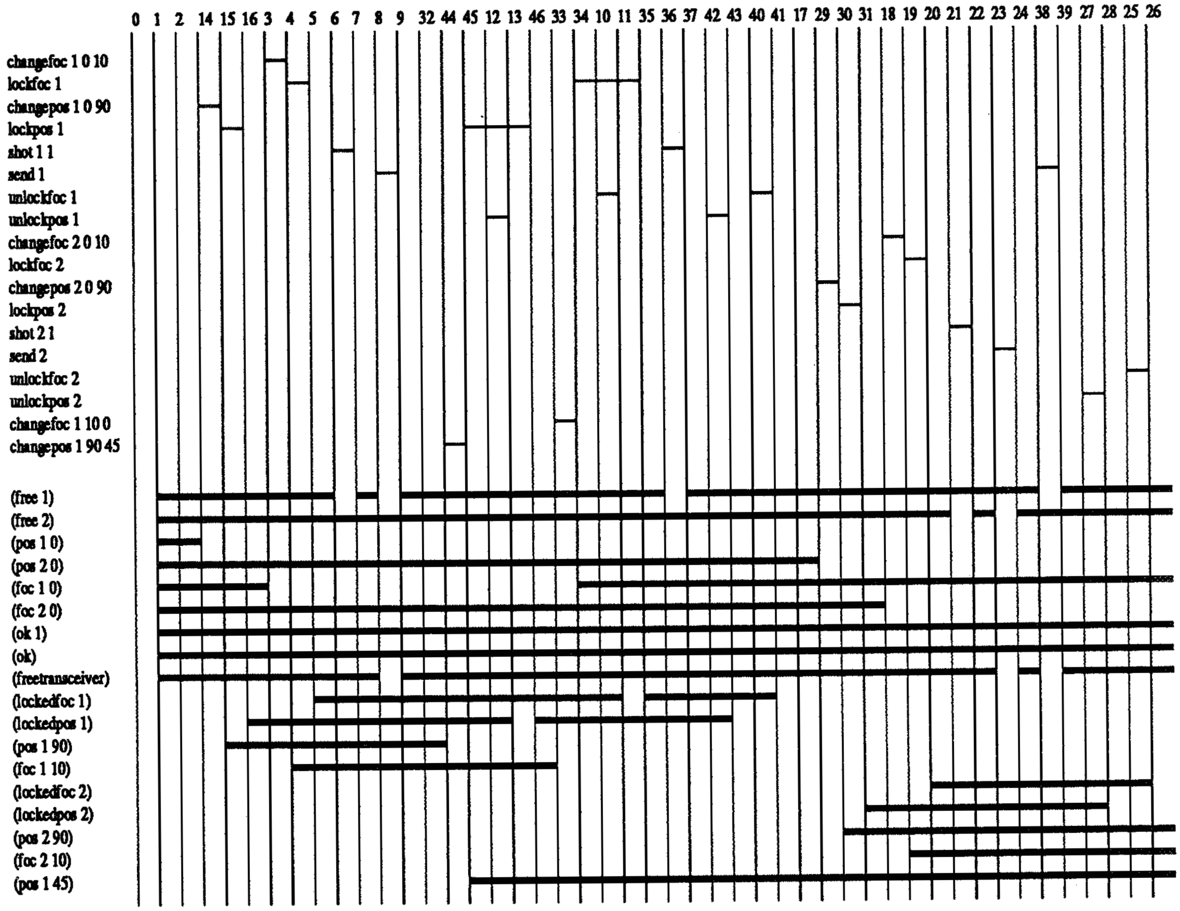
Now, it is possible from this example to build some planned actions. For instance 2 tasks **ShotAndSend** may occur one after the other with different arguments, and then we want to insert a new one. We see in the generated lattice (figure below) all the constraints that the planner has to deal with; but we see also all the time points that are independent and that represent the possible parallelism between actions.

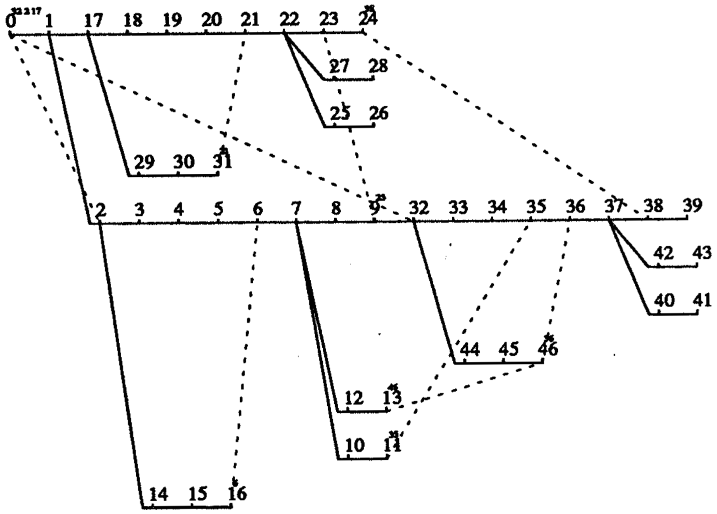




The previous figure shows the combined actions (Shotandsend 1 1 1 0 10 0 90) and (Shotandsend 1 2 1 0 10 0 90). The two tasks are almost completely independent: the only constraint between them is the use of the transceiver to transmit data to earth. This is represented by the constraint t9 ⟨ t23.

***Insertion***





Let us now add a new task with camera 1: (Shotandsend 1 1 1 10 0 90 45). This new task will be placed opportunistically with respect to the constraints induced by each action and the already computed IxTeT. We see clearly that this task is very constrained by the former task with the same camera, but only one constraint attaches it to the task with the other camera: the use of the transceiver.

This example shows clearly the IxTeT ability for dealing with temporal constraints, placing them opportunistically at the right place without any other information than the temporal constraints given to it. Another point is very important: this is done quickly. The compilation of the set of tasks took 1.8 second and the insertion of the third task in the final plan about .16 second (on a Sun 3). The response time of IxTeT is low enough to permit its use in real-time reactive systems.

This simple example did not illustrate all aspects of the IxTeT planning system. The actions and tasks may affect more than the simple state of the world: they may change the reasoning process by some side effects that are not directly induced by the tasks. Sometime, it is necessary to guide the planner in the choice of the task to be performed. All this is done by rules. There are two types of rules: temporal and non-temporal rules.

The non-temporal rules describe the logical relations that may exist between effects. For example, if a camera is getting an image, it is not sending images to the recorders. These rules describe essentially all the exclusions between actions and effects, and so, the side effects of actions and tasks. They enable to compute the qualification and ramifications of tasks.

Temporal rules are essentially rules to induce tasks in certain temporal situations. For example, if the satellite will be over a certain region for more than 1 minute, photos have to be taken. Such rules are slightly different from the ones we saw before: their conditions are temporal relations, and their second part contains tasks and not effects. The use of this second type of rule is essentially to give the planner a priority in its choice of a task.

## 5. Conclusion

This paper has argued that teleoperation for a space robot is complementary to the autonomy needed in tasks ranging from sensing and environment interpretation to planning, acting and reacting to events. Those tasks involve time as a constraint and as a concept to be explicitly represented and reasoned with. One thus needs a representation of time and temporal relations that is both powerful from the reasoning point of view and efficient from the computational point of view.

A representation aiming at such goals has been proposed. It relies on the time point algebra for expressing symbolic temporal relations. A data base of such relations is managed through an efficient data structure, an indexed maximal spanning tree of the time lattice. We have shown how this structure is built, maintained and used by the Time Map Manager developed. Empirical evidences have been reported that support the linear complexity of the algorithms involved and the overall efficiency of the proposed representation.

The use of the proposed representation in planning was thus considered. A formalism for describing elementary actions and tasks in terms of symbolic temporal relations was developed. A preprocessing transforms such description of tasks into particular arrays and time lattices. Planning proceeds along the hierarchical goal decomposition approach by inserting such arrays and time lattices into a larger similar structure of goals, subgoals and expected events. Plan refinement and modification are carried out by the same process.

This planning system has been exemplified through the task of generating and changing the mission plan of a simple surveying satellite.

## References

[1] Allen, J. F. Maintaining knowledge about temporal intervals Communications of the ACM 26(11):832-843, November 1983

[2] Allen J.F and Koomen J.A. Planning using a Temporal World Model. Proc. 8th IJCAI, Aug. 1983

[3] Allen, J. F. Towards a general theory of action and time. Artificial intelligence 23: 123-154, 1984

[4] Ghallab, M., Alami, R. and Chatila, R. Dealing with time in planning and execution monitoring Robotics Research 4, R. Bolles, MIT Press, 1988

[5] Ghallab M. and Mounir Alaoui A. Managing efficiently temporal relations through indexed spanning trees. Rappot LAAS 88.360, Dec. 1988, 13p.

[6] Granier, T. Contribution a l'etude du temps objectif dans le raisonnement Rapport LIFIA RR 716-I-73, Grenoble, February 1988

[7] Koomen, J. A. G. M. The TIMELOGIC temporal reasoning system in common lisp Technical report TR 231, Rochester University, November 1987

[8] Maarek, G., Gateau, D., Hua, L. and Proth, J.M. Projet SPOT, Rapport final CNES/INRIA/SEMA-METRA, September 1987

[9] McDermott, D. V. A temporal logic for reasoning about processes and plans Cognitive Sci. 6:101-155, 1982

[10] Mounir Alaoui, A. IxTeT: un systeme de gestion de treillis d'instants Rapport LAAS 88.154, Laboratoire d'Automatique et d'Analyse des Systemes / CNRS, Toulouse, June 1988

[11] Sacerdoti E.D. A Structure for Plan and Behavior. Elsevier, 1977

[12] Shoham, Y. Temporal logics in AI: semantical and ontological considerations Artificial intelligence 33: 89-104, 1987

[13] Tate A. Generating Project Network. Proc. 5th IJCAI, Aug. 1977

[14] Tsang, E. Time structure for AI in Proceedings of the tenth IJCAI: 456-461, 1987

[15] Vere S.A. Planning in Time: Windows and Durations for Activities and Goals. IEEE Trans. PAMI, 5, 3, May 1983

[16] Vilain, M. and Kautz, H Constraint propagation algorithms for temporal reasoning in Proceedings of the fifth national conference on artificial intelligence (AAAI-86):377-382, August 1986

**Acknowledgement**: This work was supported by the EEC under the ESPRIT project P1560 (SKIDS), and by the French National Research Program GRECO/PRC-IA.

# Reactive Behavior, Learning, and Anticipation

Steven D. Whitehead and Dana H. Ballard
The University of Rochester
Computer Science Department
Rochester, New York 14627

**Abstract**

When should a robot act and when should it think? Reactive systems always act, thinking only long enough to "look up" the action to execute. Traditional planning systems think a lot, and act only after generating fairly precise plans. Each represents an endpoint on a spectrum. When should a robot act and when should it think?

In this paper, it is suggested that this question is best addressed in the context of systems that learn. It is argued that primitive forms of reasoning, like anticipation, play an important role in reducing the cost of learning and that the decision to act or think should be based on the uncertainty associated with the utility of executing an action in a particular situation.

We present an architecture for an adaptable reactive system and show how it can be augmented with a simple anticipation mechanism that can substantially reduce the cost and time of learning.

## 1 Introduction

Much of the earliest research on intelligent robotics viewed robot control primarily as a two stage process of plan generation and execution. During plan generation the system reasons about the world in order to construct a plan that is followed during execution. Plan generation was considered the difficult, intelligent part of the problem; and received the most attention [13, 7, 15, 16, 18, 19, 5]. Plan execution, on the other hand, was considered more straightforward, less intelligent; and until recently received much less attention [6, 20]. Unfortunately, this two stage *reason then act* model for robot control has been shown to have several limitations, including: an inability to adequately model the effects of actions (variants of the frame problem)[10], an inability to adapt plans based on unexpected opportunities and contingencies (lack of flexibility)[1], and an inability to generate plans quickly (computational intractability) [5].

These problems have recently led to the emergence of *Reactive Systems.* These systems are primarily concerned with responsiveness and competence, emphasizing the *timely* generation of *appropriate* behavior in a wide variety of situations. Instead of relying on costly symbolic reasoning, reactive systems depend on precompiled knowledge about how to behave given a particular situation. Instead of following an explicit plan, generated by a planner, reactive systems use information present in the world (and sometimes a small amount of internal state), as an index to "look up" the action to execute next [1, 9, 8, 17, 2, 4, 12]. Another interesting property of *some* of these systems is that they contain no centralized, sequential controller. Instead, the parallel interaction

*This work was supported in part by NSF research grant no. DCR-8602958 and in part by NIH Public Health Service research grant no. R01 NS22407-01.

of a number of relatively simple, semi-independent subsystems leads to the emergence of intelligent behavior (Especially see [1, 3, 2]).

Reactive systems raise a number of important new issues. One issue is adaptability, which current reactive systems lack. Currently, knowledge about decision making is built into the system *a priori.* This precludes the system from exploiting unanticipated structure in the problem domain to improve its performance. Although machine learning has received considerable attention for quite some time, little work has been done to integrate machine learning results with reactive systems. A second issue is the relationship between reasoning and reacting[14]. From the standpoint of a cognitive model, reasoning is important. Introspectively, it is clear that humans contemplate the effects of actions, anticipate the future, and build and execute plans. It seems inevitable that highly intelligent robots must eventually incorporate mechanisms for anticipation and planning; but the role of reasoning in reactive systems is unclear. Current reactive systems do not include mechanisms for reasoning.

This paper claims that the role of reasoning in reactive systems is best addressed in the context of systems that learn. We argue that primitive forms of projection and anticipation may have evolved out of the need to minimize the cost and time associated with learning. Also, we define an adaptable reactive system that is based on the classifier systems described by Holland *et. al.*[11]. The system is massively parallel and relies on rule priorities to make decisions. Priorities are approximations of the expected utility of executing a rule's action. The system learns by adding and deleting rules and by adjusting rule priorities. Although the system may eventually learn to perform optimally, the time and cost of learning optimal behavior may be prohibitively expensive. To reduce this cost a simple projection mechanism is introduced. Projection allows the system to predict the effects of actions and to use those predictions to reduce uncertainty about the utility of executing an action. In familiar situations, projection is almost useless, but in novel situations it can greatly aid decision making. It is suggested that although primitive, this projection mechanism may be a precursor to more complex forms of reasoning.

# 2 Adaptable Reactive Systems

## 2.1 The Importance of Adaptability

Most reactive systems reported to date are static. The knowledge used for decision making is determined at design time and the performance of the system is the same after ten years of operation as it is on the first day. Although the current interest in reactive systems is important to the development of intelligent robotics, equally vital is that these systems be adaptive. There are several reasons why adaptability is so important. First, the world is extremely complex. Even in limited domains it is unlikely that a robot designer can anticipate all the subtleties that affect behavior. Second, the world is idiosyncratic. The world of each individual robot contains important features that can be exploited to optimize performance, but which cannot be anticipated in advance. For example, the spatial layout of a Mars rover's local environment cannot be anticipated, but can be learned and used to increase the system's performance. Third, the world is dynamic; components fail, objects move, objectives change. Without an ability to adapt, a once optimal system can become useless. Evidently, non-adaptive systems are doomed to be useful only in small, static domains.

## 2.2 Adaptable Reactive Systems

Recently, Holland *et al.* have proposed inductive mechanisms for adapting massively parallel rule based systems. They have used these mechanisms to model various forms of learning, from classical conditioning to scientific discovery[11]. One application of these elegant ideas is to adaptable reactive systems. By an adaptive reactive system we mean a massively parallel, adaptive rule based system which controls a robot by repeatedly examining the current sensory inputs, and using its rules to reactively pick and execute an action. Formally, an adaptable reactive system (or ARS) is defined by the tuple $< S, C, E, D >$. $S = \{s_1, s_2, ..., s_n\}$ is a set of atomic propositions describing the current state of the world and $C = \{c_1, c_2, ..., c_l\}$ is the set of actions the system can execute. $E = \{e_1, e_2, ..., e_{2^n}\}$ is a set of state utility estimates. There is one utility estimate, $e_m$, for each world state, $m \in W$, where $W$ is the set of world states. Intuitively, the utility of a state is a measure of the value of being in that state. Since in general the robot does not know the utility of a state *a priori*, the utility estimates are approximations to the actual state utilities. The estimates are incrementally improved using a technique known as the *bucket brigade* algorithm. $D = \{d_1, d_2, ..., d_k\}$ is a set of production rules of the form: $d_i : P_i \xrightarrow{\rho_i} G_i$, where $P_i$ is a sentence constructed out of the atomic propositions $S$ and the connectives $\wedge, \vee, \neg$ in the usual way, the action $G_i \in C$ is a single executable action, and $\rho_i$ is a priority value used to guide decision making.

An ARS operates as follows. At each time step (or trial), the system is presented with an input pattern describing the state of the world. The pattern is used to evaluate the conditions (LHS) of the production rules. The set of rules whose conditions are satisfied on a given trial are said to be *active*. An active rule represents a tendency for the system to execute the action associated with that rule. For example, the rule: $s_1 \wedge s_2 \rightarrow c_1$ says "If $s_1 \wedge s_2$ is true in the current situation then *try* to execute action $c_1$." On each trial, multiple rules are likely to become active and several actions may be suggested for execution. The system is allowed to only execute one action per trial and uses a priority based bidding mechanism to choose the action to execute. The bidding mechanism works as follows: on each trial, an active rule is a bidder if there is no other active rule whose condition is more specific. Each bidder makes a bid for its associated action. A bid contains *an action* and *a priority*. After all the bids are in, the bids are tallied and a total bid for each action is determined. The total bids are then used to probabilistically choose the action to be executed. For example, one means of determining the action to execute is to assign a probability to each action on a per trial basis that equals the normalized sum of the bids received for that action. That is,

$$pr(a_i) = \frac{\sum_{j \in B_{a_i}} \rho_j}{\sum_{k \in B} \rho_k} \tag{1}$$

where $pr(a_i)$ is the probability that the system chooses to execute action $a_i$, $B_{a_i}$ is the set of bidders who bid for action $a_i$, $B$ is the set of all bidders, and $\rho_j$ is the priority of bid $j$. A rule is said to have *executed* in a trial if it is active and its action is executed.

Clearly, rule priorities play a crucial role in determining system performance. How are priorities assigned? Utility estimates are used to determine rule priorities. The priority of a rule is adjusted to approximate the expected utility estimate of the state that results from *executing* the rule. That is,

$$\rho_i \approx \sum_{m \in W} e_m \cdot pr(m \mid d_i) \tag{2}$$

where $pr(m \mid d_i)$ is the probability that the system will be in state $m$ at time $t+1$ given that it executes rule $d_i$ at time $t$. A simple algorithm for computing a rule's priority is to think of the utility estimates as supplying a form of *internal reenforcement*. That is, in each trial the system executes an action and gives itself an internal reenforcement equal to the utility estimate of the resulting state. This reenforcement is used to incrementally adjust the priority according to the following rule:

$$\rho_i(t+1) = \begin{cases} \gamma e(t+1) + (1-\gamma)\rho_i(t) & \text{if rule } d_i \text{ is executed in trial } t \\ \rho_i(t) & \text{otherwise} \end{cases} \tag{3}$$

where $e(t+1)$ is the utility estimate of the resulting world state at time $t+1$ and $\gamma$ is a constant that geometrically decreases the weight given to trials that occurred long ago. The basic architecture for an ARS that uses utility estimates for internal reenforcement is shown in Figure 1. Using this

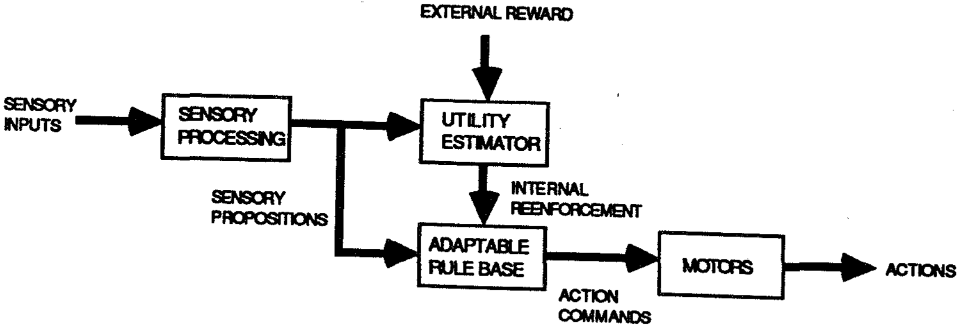


Figure 1: The basic architecture of an adaptable reactive system, (ARS).

priority assignment scheme, it is easy to see that if the system has good utility estimates it will tend to execute those actions that maximize the expected utility. The trick then is to define an appropriate utility function and a mechanism for learning it.

Although any number of utility functions can be defined, a good approach is to define the utility of a state to be the expected reward of possible futures that begin in that state. That is, suppose that an *external reward* is associated with each world state, and that the robot's objective is to obtain as much reward as possible. Intuitively, the robot wants to be in or "near" states that have a high reward associated with them, and the utility of a state should reflect this preference.

Formally, a *possible future* for the robot, $p$, is a sequence of states and actions, $m_0, a_0, m_1, a_1, ...$, where $m_i$ is the state of the environment at time $t_i$ and $a_i$ is the action executed by the system at time $t_i$, respectively. Further the *utility of a possible future* is a weighted sum of the reward received by the system upon following that possible future, namely:

$$U_p = \sum_{i=0}^{\infty} \beta^i r(m_i) \tag{4}$$

where, $U_p$ is the utility of possible future $p$, $\beta$ is a constant used to geometrically diminish the importance of states on the path that are far into the future, and $r(m_i)$ is the external reward received by the system in state $m_i$.

The *utility of a state* is then defined as the expected utility of possible futures that begin in that state. That is,

$$U_m = \sum_{p \in F_m} U_p pr(p \mid m) \tag{5}$$

where $U_m$ is the utility of state $m$, $F_m$ is the set of possible futures that begins in state $m$, and $pr(p \mid m)$ is the probability that possible future $p$ is followed given that the system starts in state $m$. Given this definition of utility, it can be seen that by maximizing its expected utility, a system also approximately maximizes its expected reward.

As mentioned above the system doesn't know the state utilities *a priori*. Instead utilities are learned as the system experiences the world and receives external rewards. At any given time, the utility estimates, $E$, represent the system's best approximation of the state utilities. Utility estimates are incrementally adapted using the ***bucket brigade*** algorithm described by Holland *et. al.*[11]. The basic idea behind the bucket brigade algorithm is that state utilities are related by the following system of equations:

$$\forall_{m \in W}[u_m = r(m) + \sum_{m' \in W} u'_m pr(m' \mid m)] \tag{6}$$

where $u_m$ is the utility of state $m$, and $pr(m' \mid m)$ is the probability that state $m'$ is obtained at time $t+1$ given the system is in state $m$ at time $t$. The bucket brigade algorithm makes use of these equations to incrementally build up approximations of a state's utility based on approximations of other states' utilities. Although the details of the bucket brigade algorithm are extremely interesting, for the purposes of this paper, it is sufficient to know that such an algorithm exists and that a massively parallel implementation is realizable.

In addition to estimating state utilities and modifying rule priorities, the system can adapt by generating new rules and replacing old "less useful" ones. Although the algorithms for this form of learning are important, they are not central to this paper and therefore will not be discussed. It is sufficient to say, that at any given time only a subset of possible rules are actually in the system (due to space constraints) and that as the system learns, rules are introduced and deleted in way that tends to keep around those rules that contribute most to the system's performance.

### 2.3 Stacking Blocks

To illustrate the ideas presented above, consider a robot that plays the following game. The robot sits in front of a table that has three blocks on it: A, B, and C. The robot can manipulate the blocks and arrange them in any configuration of stacks it likes. When the robot is happy with the arrangement of blocks it presses a small button near the edge of the table and receives a reward that is commensurate with the arrangement of blocks on the table. After the robot receives a reward, the blocks are randomly "scrambled" for another round. The robot's objective is to maximize its overall reward.

To formalize this game the sensory propositions $S$, the available actions $A$, and the reward function $r$ must be defined. Suppose the robot has the following sensory propositions: $S = ON \cup CLEAR \cup LEVER$. Here $ON$ is a set of propositions describing whether one block is on top of another or not:

$$ON = \{ON(A,B), ON(A,C), ON(A,Table), ON(B,A)...\}$$

where $ON(i,j)$ is true iff block $i$ is on block $j$; $CLEAR$ is a set of propositions that describe whether or not a block has another block on top of it:

$$CLEAR = \{CLEAR(A), CLEAR(B), CLEAR(C)\};$$

and LEVER is a proposition that is true just when the robot has pushed the lever to receive the reward. Suppose further that the robot can execute the following actions: $A = PUSH\_LEVER \cup MOVE$, where $PUSH\_LEVER$ is an action for pushing the reward lever and $MOVE$ is a set of actions for placing blocks on top of each other and on the table:

$$MOVE = \{MOVE(A,B), MOVE(A,C), MOVE(A,Table), MOVE(B,A), ...\}$$

Here $MOVE(i,j)$ represents the action of placing block $i$ on top of object $j$. Assume that both the block being moved and the target object must be clear for the action to successfully complete. Otherwise, assume the action fails and the configuration of blocks remains unchanged. Finally, suppose the reward function is defined as follows: If the reward lever is up (ie $LEVER = T$) then the reward is zero for all block arrangements. If the reward lever is down, then the reward value is

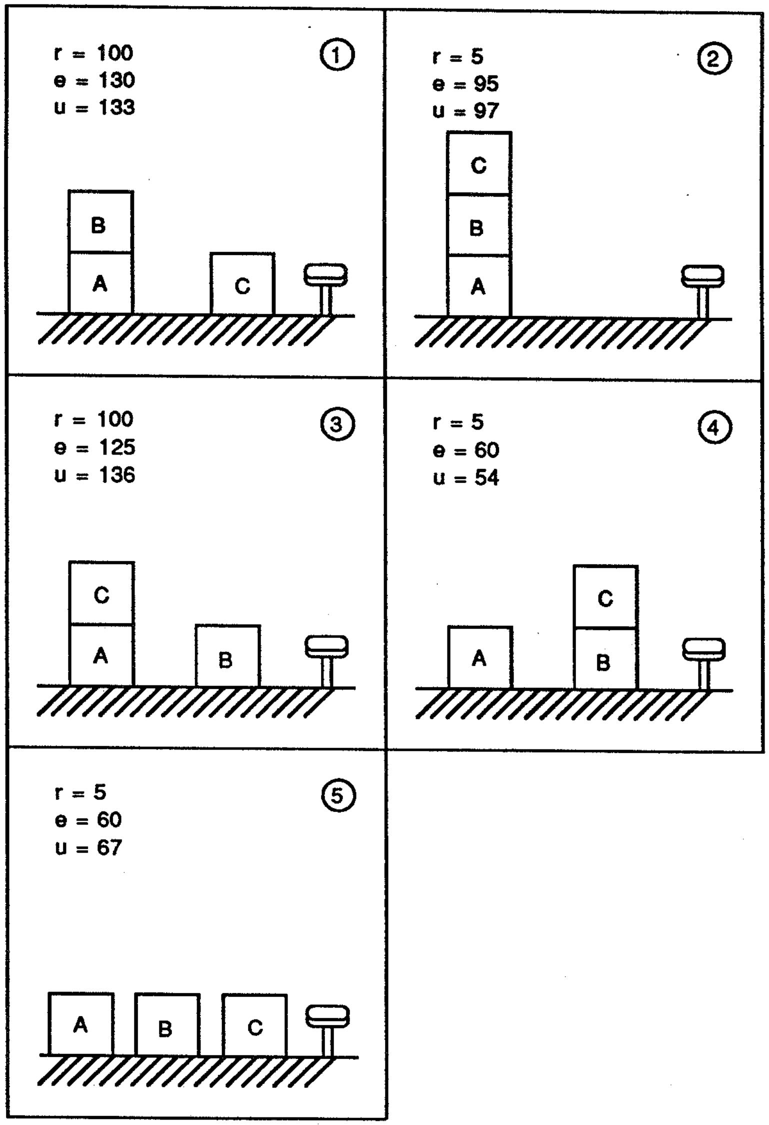


Figure 2: Five states of the block stacking game. In the upper left hand corner of each frame is the utility value and the utility estimate associated with the state. Also shown is the reward value associated with the state with the block configuration shown but with the reward lever depressed.

| *No.* | *Rule* | $\rho$ | *States Active* |
|---|---|---|---|
| 1 | $ON(B,A) \land ON(C,Table) \land ON(A,Table) \land \neg LEVER\_DOWN \rightarrow PUSH$ | 129 | 1 |
| 2 | $ON(A,Table) \land ON(B,A) \land ON(C,B) \land \neg LEVER\_DOWN \rightarrow PUSH$ | 37 | 2 |
| 3 | $ON(A,Table) \land ON(B,A) \land CLEAR(C) \land \neg ON(C,Table) \land \neg ON(C,A)$ $\rightarrow MOVE(C,Table)$ | 109 | 2 |
| 4 | $ON(B,Table) \land ON(A,Table) \land CLEAR(B) \land ON(C,A) \rightarrow PUSH$ | 132 | 3 |
| 5 | $\neg ON(C,Table) \land CLEAR(C) \rightarrow MOVE(C,Table)$ | 75 | $2,3,4,+$ |
| 6 | $ON(A,Table) \land ON(B,Table) \rightarrow PUSH$ | 65 | $3,4,5,+$ |
| 7 | $CLEAR(A) \land CLEAR(C) \rightarrow MOVE(C,A)$ | 72 | $4,5,+$ |

Table 1: Typical rules from an ARS robot playing the block stacking game. Rules are numbered in column 1 and and shown in column 2. Rule priorities are shown in column 3 and the states in Figure 2 in which the rule is active are listed in column 4. The first four rules are specific, the last three are general.

100 if block A is on the table with another block on top of it, and the third block is on the table. Otherwise the reward is 5.

Figure 2 shows five frames each representing a possible world state. The utility value $u$, and the utility estimate $e$, for each state is shown in the upper left hand corner of each frame. The utility value is fixed when the game is defined but is unknown to the robot. The utility estimate is the robot's model of the utility function and changes as the robot plays the game, based on the bucket brigade algorithm. Also shown in the upper left hand corner is the reward value associated with the state with the block configuration shown but with the reward lever in the down position. The two states with the reward lever down and the block configurations shown in frames 1 and 3 are the only states where the robot receives a reward of 100.

Of the five states shown, states 1 and 3 have the highest utilities. Intuitively this follows since in these states all the robot has to do to receive a large reward is press the reward lever. The utilities of the states 2, 4, and 5 are somewhat less since the robot has to go further to receive a large reward. The utility estimates shown in the figure indicate that the system has a fairly good model of the utility function.

A partial list of rules for a robot playing the game is shown in Table 1. Although this list is not complete it is representative of the kinds of rules the system will generate. The first four rules are very specific. In each of these rules, the condition is satisfied in only one state (rule 1 is satisfied in state 1, rules 2 and 3 in state 2, and rule 3 in state 3). Rules 5 - 7 are more general and are satisfied in several states. Specific rules are important for encoding knowledge about the correct action to take in familiar situations, because they are satisfied only in very specific situations and are likely to have good utility estimates. Specific rules, however, are less useful for dealing with novel situations since they are less likely to be become active. General rules are the opposite; they are active more often but are less useful in familiar situations. The reason they are less useful in familiar situations is that the internal reenforcement they receive tends to be highly variable and their priorities are less likely to reflect the actual utility of executing an action *in a particular situation.* Figure 3 illustrates this point by showing that the priority of a specific rule is an average of utilities over a very small number of states while the priority of a general rule averages utilities over many states. In general, an ideal rule is one that 1) consistently receives the same reenforcement (low variability) and 2) is applicable in many states (abstract). Specific and general rules tend to have one or the other of these properties. As the robot plays the game it will tend to generate more and more specific rules as it learns about the state space. Initially, general rules are good since they represent heuristics or tendencies that lead to good payoffs. However, they are eventually replaced by more specific rules (with less variable reenforcement) that are particular to the idiosyncrasies of particular states.

To illustrate the basic operation of the system, suppose the world is currently in state 2. At the

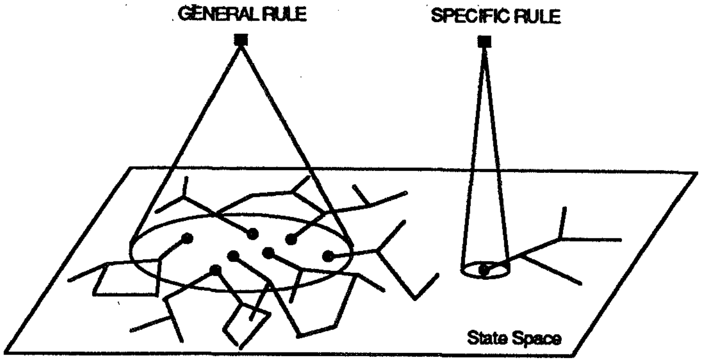


Figure 3: The priority of a general rule is the average utility of many possible futures since the rule is satisfied in may states. The priority of a specific rule, however, is an average over many fewer states and therefore is likely to have a smaller variance.

beginning of the trial, the sensory propositions are set and used to evaluation rule conditions. In state 2 rules 2, 3, and 5 are satisfied and become active. Of these rules, 2 and 3 are *most specific* and bid for control. Rule 5 is a generalization of rule 3 and therefore is a non-bidder. Since rules 2 and 3 call for different actions, the system probabilistically decides whether to execute $MOVE(C, Table)$ or $PUSH$. That is, according to Equation (1):

$$pr(MOVE(C, Table)) = \frac{\rho_3}{\rho_3 + \rho_2} = \frac{109}{109 + 37} = 0.75 \tag{7}$$

and

$$pr(PUSH) = \frac{\rho_2}{\rho_3 + \rho_2} = \frac{37}{109 + 37} = 0.25 \tag{8}$$

where $pr(MOVE(C, Table)$ and $pr(PUSH)$ are the probabilities of executing the move and push actions respectively.

Suppose, in this case that the system decides to execute $MOVE(C, Table)$. Then after moving $C$, the world is in state 1 and the priorities of rules 3 and 5 are updated based on the utility estimate for state 1, namely $e_1 = 130$. The priority of rule 5 is updated because its condition is satisfied and the action chosen was consistent with the action specified by the rule. Once the system is in state 1, based on rule 1, it is likely to push the reward lever and receive a sizable reenforcement.

# 3 Anticipation in Systems that Learn

## 3.1 Why anticipation is good

Almost all learning systems repeat the following cycle again and again: The system senses the input, decides on and executes an action, receives feedback based on the action, and adjusts itself to better its performance. The problem is that if the system learns slowly it must endure a lot of negative reenforcement. Learning is expensive. Using a faster learning algorithm is one way to reduce this cost, *anticipation* is another. Here, anticipation means the ability to predict future states by *projecting* the effects of actions. When a system without anticipation encounters a novel situation, where it doesn't have a good estimate of the utility of possible actions, it must randomly choose an action, suffer the consequences, learn and go on. In environments where wrong moves can be catastrophic, systems without anticipation are doomed to suffer. If a system can recognize that

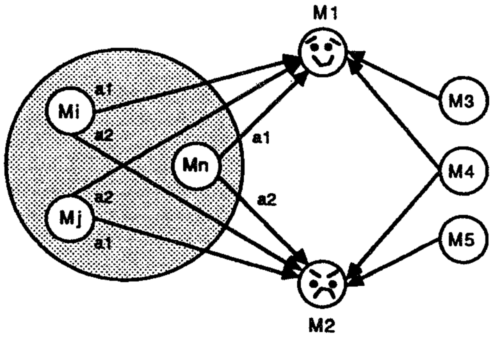


Figure 4: By entering states $m_1$ and $m_2$ from states $m_3, m_4$, and $m_5$ on previous trials, the system has learned that $m_1$ has high utility and that $m_2$ has low. Entering the new situation, $m_n$, a system without anticipation must rely on general rules which may be unreliable. A system with anticipation, however, can project the effects of possible actions and obtain better utility estimates, which result in better decisions.

it is in an unknown situation and project the effects of its actions, it may be able to obtain a better estimate of the utility of executing an action in that particular situation.

Figure 4 illustrates the idea. Suppose that it has been determined with good certainty that states $m_1$ and $m_2$ have very high and very low utilities respectively. The certainty of these estimates could be based on experience gained while visiting $m_1$ and $m_2$ from states $m_3, m_4$, and $m_5$. Further suppose the system has found itself in the novel state, $m_n$, for which it has only a rough estimates of the utilities of executing actions. These rough estimates come from "general" rules that are developed while executing in similar situations, like states $m_i$ and $m_j$. Without anticipation, the system must rely on its general rules to guide its behavior. In this case, there is a good chance that the system will make a poor decision and suffer. If the system is capable of projecting the effects of actions, then it can "internally" explore its options by looking at the utility estimates of projected states, and feed back those estimates to *temporarily* adapt rule priorities. In the example above, the system might simulate executing action $a_2$ to find it yields state $m_2$; use the low utility estimate of $m_2$ to temporarily degrade the priority of the rules that caused $a_2$ to be executed; evaluate the rules again; simulate action $a_1$; find it leads to a high utility state; temporarily upgrade the priority of those rules; and finally execute $a_1$. Of course, a projection can be more than one step into the future. If the utilities of the projected states are themselves uncertain, the system can continue to project until it reaches a state that has a stable estimate.

In the context of the block stacking example given above, state 4 is a relatively novel state since only general rules are active in this state; in this case rules 5, 6 and 7. Notice how the priorities associated with each of these rules is roughly an average of the utility estimates of the states that are obtained after executing the rule any time its condition is satisfied. For any given activation of these rules, the actual internal reenforcement received may vary wildly. For example executing rule 5 in state 2 yields a reenforcement of 130, while executing rule 5 in state 4 yields a reenforcement of only 60. Using anticipation, the system can predict the low yield obtained by executing rule 5 in state 4 and can temporarily lower its priority. This in turn leads to the anticipation of the effect of rule 7, which yields a high internal reenforcement when executed in state 4. This finally leads to a temporary increase in the priority of rule 7, and eventually causes it to be executed.

As can be seen from the above examples, the variance in the internal reenforcement received by a rule is crucial for determining whether to act or anticipate. If the variance is low, the priority is a good approximation of the internal reenforcement the system is likely to receive when it executes the

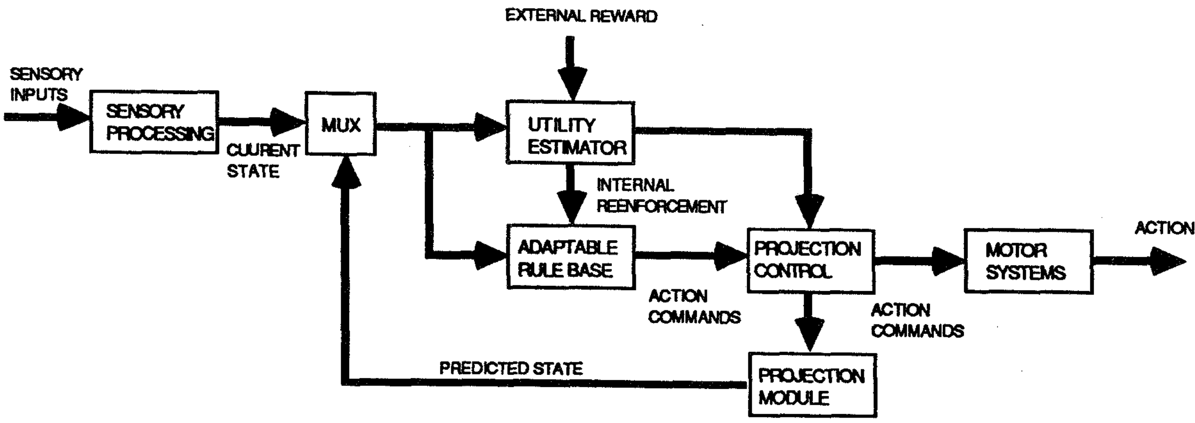


Figure 5: The ARS architecture augmented with a projection mechanism. In unfamiliar situations, where the uncertainty about the utility of an action is high, the system projects the effects of its actions to reduce uncertainty.

action. In this case, anticipation isn't very useful because it will just yield the same result. However if the variance is high, the actual reenforcement received may be very different than the average. In this case, anticipation is useful since it can lead to a much better approximation of the utility of executing the action in a particular situation.

## 3.2 Adding Anticipation to the ARS architecture

To add an anticipation mechanism to the ARS architecture, rule priorities are augmented with a variance value, $\sigma_i$, which is an estimate of the variance in the internal reenforcement received when the rule executes. Similarly, a variance value is also maintained for the utility estimates associated with each state.[1] Finally, a prediction mechanism is added that maps states and actions into predicted states.[2] The resulting architecture is shown in Figure 5.

The basic algorithms implemented by the architecture are given below:

Action Selection Algorithm

For each trial:

1. determine the active rules;
2. determine the bidders (most specific rules);
3. probabilistically pick an action to execute (based on current priorities and priority variances);
4. If the variance associated with the chosen action is low enough, say below $V\_thres$ then execute the action; otherwise:
    (a) project the effects of executing the action;

[1] The algorithm for determining utility variances is simple, and based on a slight modification of the Bucket Brigade algorithm.

[2] In general predictions must be learned as well. However for the purposes of this paper, we assume prediction knowledge is known *a priori*.

(b) use the results of the projection to temporarily adjust the priorities and variances of active rules, and

(c) go to Step 3;

Projection Algorithm

For each projection:

1. Based on the current world state and knowledge about the effects of actions, predict the state of the world after executing the action;

2. If the uncertainty about the prediction is too high, say above *P_thres*, then quit the projection and return a degraded priority value; otherwise,

3. If the variance of the utility of the predicted state is low enough, say below *V_thres*, then quit the projection and return the utility of the predicted state (degraded by $\beta$) and its variance as new estimates of the action's priority and variance respectively; otherwise,

4. Use the current predicted state as the basis for another projection.

There are two reasons to terminate a projection. First, if there is a large uncertainty about the results of the projection, then the projection may as well terminate because further projection probably won't help reduce uncertainty. Second, if the projection has lead to a state with a stable utility estimate (low variance), then the projection should terminate because this information can be used to lower the uncertainty and further projection is not likely to be more useful.

# 4 Summary and Future Work

In this paper, an adaptable reactive system was presented. It was shown that by adding a simple anticipation mechanism the cost of learning could be reduced by allowing the system to avoid undesirable situations. It was argued that the decision to act or project should be based on the uncertainty in the utility of executing an action in a particular state. Using anticipation this uncertainty is reduced by projecting into the future until a state with a stable utility is obtained. This utility is then used to derive a more certain estimate.

As of this writing, empirical results regarding the savings gained by adding projection are not available. We are currently in the process of implementing a system that plays the block stacking game. There are also several other directions for future work. One limitation of the current system is that the sensory and motor systems are binary. We hope to replace sensory propositions with semi-continuous variables, and augment the motor system to allow multiple simultaneous actions.

Abstraction is another area that needs attention. In the current system projections are one "primitive" action at a time; it would be interesting to develop an abstraction mechanism that allowed projection over "abstract" actions. Finally, short term memory has been ignored. It was assumed that the sensory information available to the system was sufficient to define the state of the world. This is fallacious in general and memory is necessary to overcome this simplification. It is also interesting to speculate about how memory could be coupled with projection to lead to more complex forms of reasoning.

# References

[1] Philip E. Agre and David Chapman. Pengi: An implementation of a theory of activity. In *AAAI*, pages 268–272, 1987.

[2] James S. Albus. ***Brains, behavior, and robotics.*** BYTE Books, Petersborough NH, 1981.

[3] Rodney A. Brooks. Achieving artificial intelligence through building robots. AI Memo 899, MIT, 1986.

[4] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, pages 14–22, April 1986.

[5] David Chapman. Planning for conjunctive goals. *Artifical Intelligence*, 32:333–377, 1987. also MIT AI Technical Report 802 November, 1985.

[6] Richard E. Fikes, Paul E. Hart, and Nils J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3(4):251–288, 1972. Also in Readings in Artificial Intelligence, 1980.

[7] Richard E. Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[8] R. James Firby. An investigation into reactive planning in complex domains. In *AAAI*, pages 202–206, 1987.

[9] Michael P. Georgeff and Amy L. Lansky. Reactive reasoning and planning. In *AAAI*, pages 677–682, 1987.

[10] Patrick Hayes. The frame problem and related problems in artificial intelligence. In A. Elithorn and D. Jones, editors, *Artificial and Human Thinking*, pages 45–49. San Francisco: Jossey-Bass, 1973. Also in Readings In Artificial Intelligence, 1980.

[11] John H. Holland, Keith F. Holyoak, Richard E. Nisbett, and Paul R. Thagard. *Induction: processes of inference, learning, and discovery.* MIT Press, 1986.

[12] L. P. Kaelbling. Goals as parallel program specifications. In *Proceedings of AAAI-88*, page 60, 1988.

[13] Nils J. Nilsson. Shakey the robot. Technical Note 323, SRI AI Center, 1984.

[14] Panel on Planning and Execution. Rochester planning workshop, 1988.

[15] E. D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5(2):115–135, 1974.

[16] E. D. Sacerdoti. *A Structure for Plans and Behavior.* New York: Elsevier, 1977.

[17] M. J. Schoppers. Universal plans for reactive robots in unpredictable domains. In *Proceedings of IJCAI-87*, 1987.

[18] Austin Tate. Interacting goals and their use. In *Advance Papers of the 4th IJCAI*, 1975.

[19] D. E. Wilkins. Domain independent planning: representation and plan generation. *Artificial Intelligence*, 22:269–301, 1984.

[20] D. E. Wilkins. Recovering from execution errors in SIPE. *Computational Intelligence*, 1:33–45, 1985.

# NASA JOHNSON SPACE CENTER

# SHUTTLE REMOTE MANIPULATOR SYSTEM MISSION PREPARATION AND OPERATIONS

Ernest E. Smith, Jr.

Systems Division/Mechanical and Crew Systems Branch
Johnson Space Center
Houston, Texas 77058

## Abstract

The pre-flight planning, analysis, procedures development, and operations support for the Space Transportation System (STS) payload deployment/retrieval missions utilizing the Shuttle Remote Manipulator System are summarized. Analysis of the normal operational loads and failure induced loads and motion are factored into all procedures. Both the astronaut flight crews and the Mission Control Center (MCC) flight control teams receive considerable training for standard and mission specific operations. The real-time flight control team activities are described.

## 1. Introduction

The Payload Deployment and Retrieval System (PDRS) is the 50 foot, 6 jointed robotic arm that is used during Shuttle missions primarily to deploy and retrieve payloads. The PDRS was developed by Canada (SPAR Aerospace Limited under contract to the National Research Council of Canada) through an international partnership agreement between our two countries. Three flight units are currently available and are manifested as required on Shuttle flights. NASA has flown the PDRS on 18 Shuttle flights to date, and the PDRS has been used to deploy or retrieve 14 payloads. Payload retrieval and payload servicing has also been accomplished by attaching astronauts to the end of the PDRS via a manipulator foot restraint. This paper first describes the Shuttle PDRS, then describes the mission preparation and mission operations related to the PDRS.

## 2. PDRS Description

The PDRS is a compilation of several specialized systems including the following:

- Remote Manipulator System (RMS) - The mechanical arm assembly itself, the end effector (the "hand" of the arm), and the crew interfaces (Display and Control Panel, hand controllers, and interfacing electronics). The mechanical arm has six joints, a pitch and yaw joint at the shoulder, an elbow pitch joint, and a pitch, yaw and roll joint at the wrist.

- Manipulator Positioning Mechanisms/Manipulator Retention Latches (MPM's/MRL's) - Attaches the RMS to the Orbiter longeron. Mechanism (MPM) "rolls" the RMS outboard after the payload bay doors are open, and releases (MRL) the arm prior to RMS activity. These same mechanisms also "roll" the RMS back in and latch it down for re-entry.

- Grapple Fixture - Attaches directly to the payload. Includes a grapple shaft which is captured or snared by the end effector, and a target to align the end effector to the grapple shaft during payload capture.

- Closed Circuit TV - Two cameras mounted on the RMS at the elbow and wrist joints. Monitors are located near the hand controllers.

- Software Control - Control loop algorithms, system health monitoring routines, data transfer routines, and caution and warning alarm generation resident in the on-board systems management orbiter general purpose computer. (The RMS is capable of operating without these software controls in a contingency mode.)

The RMS plus the MPM's/MRL's weighs approximately 1000 lbs. and is occasionally not flown on Shuttle missions for weight savings purposes.

There are several operating modes available for control of the RMS as follows:

- Software controlled modes:

  - Orbiter Unloaded - Multi-joint control via two 3-degrees-of-freedom hand controllers. The software joint rate limiting algorithm assumes an unloaded arm.

  - Orbiter Loaded - Similar to Orbiter Unloaded, but joint rate limiting algorithm uses payload specific mass properties constants.

  - End Effector - Similar to Orbiter Unloaded, but with a different coordinate system for control.

  - Payload - Similar to Orbiter Loaded, but with a different coordinate system for control.

  - Automatic Sequences - Complex pre-flight planned sequences and simple operator "fly-to" point designation are supported.

  - Single Joint Mode - Individual joints are driven by operator command through the Display and Control panel and routed through the RMS software.

- Hardware controlled modes:

  - Direct Drive - A contingency mode used to drive individual joints directly via hardwire control, bypassing software control. The hardware in the RMS utilized by this mode is the same as used when under software control.

  - Backup Drive - Another contingency mode used to drive individual joints directly via hardwire control, bypassing software control. Some of the hardware in the RMS utilized by this mode is different than that used by the Direct Drive or software control. Also, an alternate power source is used to power the RMS in this mode.

### 3. PDRS Operational Limitations

Payloads weighing up to 65,000 lbs. can be deployed by the RMS in a non-time-constrained operation. Most payloads to date have weighed less than 2000 lbs, with the Long Duration Exposure Facility (LDEF) being the heaviest at 21,000 lbs. Payloads weighing up to 32,000 lbs can be retrieved by the RMS. Again, most payloads have been much lighter than this. LDEF will be our heaviest payload to date to retrieve when we fly the STS-32 mission (November, 1989).

The RMS operator must have good visual cues both by observing the payload/RMS out the aft flight deck windows and via cameras. Often visual cues must be placed on the payload to enhance operations (such as berthing whiskers for LDEF). There must be sufficient clearance between all points on the payload and the Orbiter structure (at least 2 feet) during maneuvering by the RMS. The operator slows the rates at which the payload/RMS is maneuvered anytime the payload is within 5 feet of structure. Loads which may be induced on the payload or the RMS (RMS runaway failures, reaction control system jet firings, etc.) must be within the structural limitations of these systems.

### 4. Mission Preparation

The detailed deploy/retrieve procedures and analyses supporting those procedures begin in earnest approximately 2 years before the flight of a payload. Preliminary mission design and payload integration begins many years earlier. This mission design process ensures that payload unique constraints are accommodated while ensuring safe operational use of the RMS. Every payload which is to be deployed or retrieved has its own unique set of constraints which must be accommodated while on the RMS. For example, some payloads must be pointed away from the sun to prevent over-temperature, some must have their antennas pointed at communications satellites, special orientations must be developed to allow solar array/antenna deployment, etc.

The mission procedures must also consider clearances during normal RMS operations (2 foot minimum), and the Orbiter reaction control system and runaway induced loads and motion must be analyzed to ensure structural compatibility (and to avoid contact between the payload and the Orbiter). If excessive loads or motion are determined, the deploy/retrieve operations can usually be modified to eliminate the problem.

Flight control systems stability must also be analyzed, and the Orbiter/RMS/payload configuration must be controllable via the digital auto pilot software and reaction control system jets.

Mission design and procedures development is accomplished by an assigned Mission Operations Directorate (MOD) mission designer, who is supported by a team which grows as the flight nears. Payload RMS reach and visibility studies are performed on graphics supported kinematic simulators executing on office workstations, and a preliminary deploy/retrieve scenario is developed. Dynamic loads and motion analyses are performed on non-graphics simulators, and the results used to iterate on the proposed operations.

As procedures mature, they are reviewed in higher fidelity simulators which include Orbiter payload bay plus RMS scene generation and crew interfaces. Often, the astronauts become involved at this stage of procedures development. Both normal and multiple contingency procedures are developed to work around any problems which may occur during the flight. The multiple control modes of the RMS are very useful for these contingency procedures. Validated procedures are documented in the flight data file checklists, and the appropriate RMS-trained crewmen are further trained for flight specific operations. Many simulations are conducted using the developed procedures. These include integrated simulations in which both the crewmen and the flight control team are trained and tested in the use of the procedures and in the systems capabilities and malfunction recognition/analysis/workaround.

## 5. Mission Operations

The astronauts are the operators of the RMS (no control from the ground) and are trained to follow the normal procedures, plus the multiple contingency procedures that are developed for every flight. Three PDRS Mission Control Center (MCC) operators man the consoles during all PDRS operations, monitoring the health and status of the systems. If PDRS failures occur during the flight, the crew performs the pre-flight developed malfunction procedures. Between these procedures, the crew observations, and analysis of the telemetry data by the PDRS console operators, the source and operational effect of the problem is determined. Additional detailed engineering support for the PDRS systems are provided by the NASA engineering community monitoring the flight from the Mission Evaluation Room (MER), and is available as needed.

Pre-flight developed operational workaround procedures for problems are utilized if appropriate, otherwise, real-time procedures are developed. Often, the RMS is used in a manner not predicted pre-flight, and real-time procedures must be developed. For example, on STS 41-D, the RMS was used to knock off a large mass of ice that had formed on the side of the Orbiter when an overboard water vent became clogged and ice formed. All real-time developed procedures must be validated by compressing the mission design process (simulator and required support personnel are immediately available).

## 6. Closing Remarks

The PDRS is an extension of the crewman in the payload bay. It is an extremely flexible and valuable tool to the NASA space program. Operational procedures for using the RMS are either developed and validated pre-flight (for 2 years), or are developed and validated in real-time (real-time developed procedures are typically less complicated). Trained crewmen and trained console operators ensure that procedures are executed properly, and that problems are assessed and worked expeditiously. The PDRS will continue to support the deployment and retrieval of payloads, and will play a major role in the assembly and operations of Space Station Freedom.

# A COMPARISON OF THE SHUTTLE REMOTE MANIPULATOR SYSTEM AND THE SPACE STATION FREEDOM MOBILE SERVICING CENTER

Edith C. Taylor
Lyndon B. Johnson Space Center
Houston, Texas 77058

Michael Ross
Lockheed Engineering & Sciences Co.
Houston, Texas 77258

## ABSTRACT

The Shuttle Remote Manipulator System is a mature system which has successfully completed 18 flights. Its primary functional design driver was the capability to deploy and retrieve payloads from the Orbiter cargo bay. The Space Station Freedom Mobile Servicing Center is still in the requirements definition and early design stage. Its primary function design drivers are the capabilities to support Space Station construction and assembly tasks; to provide external transportation about the Space Station; to provide handling capabilities for the Orbiter, free flyers, and payloads; to support attached payload servicing in the extravehicular environment; and to perform scheduled and un-scheduled maintenance on the Space Station. This paper discusses the differences between the two systems in the areas of geometric configuration, mobility, sensor capabilities, control stations, control algorithms, handling performance, end- effector dexterity, and fault tolerance.

## 1. INTRODUCTION

The Shuttle Remote Manipulator System (RMS) was the first generation space manipulator. It was designed in the late 1970's and had its first flight on STS 2 in November 1981. It is now a mature system which has successfully completed 18 flights. The Shuttle RMS was developed by the National Research Council of Canada with their prime contractor, SPAR Aerospace of Toronto, Canada.

The Space Station Freedom Mobile Servicing Center (MSC) will be the second generation space manipulator. Currently, it is still in the requirements definition and early design stage. The MSC consists of two flight elements : the Mobile Transporter and the Mobile Remote Servicer. The Mobile Transporter is being developed by NASA/Johnson Space Center with its Space Station contractor McDonnell Douglas in Huntington Beach, California. The Mobile Remote Servicer is being developed by the National Research Council of Canada and they are again using SPAR Aerospace as their prime contractor.

This paper will begin by discussing the functions for which the Shuttle RMS has been used and then the functional requirements for the Station MSC. The paper will then discuss the differences between the two systems in the areas of geometric configuration, mobility, sensor capabilities, control stations, control algorithms, handling performance, end-effector dexterity, and fault tolerance.

## 2. MANIPULATOR FUNCTIONS

The primary functional design driver for the Shuttle RMS was the capability to retrieve and deploy payloads to and from the Orbiter cargo bay. The Shuttle RMS has successfully deployed two payloads to low earth orbit : the Long Duration Exposure Facility (LDEF) on Shuttle Flight STS 41-C and the Earth Radiation Budget Satellite (ERBS) on STS 41-G. And the Shuttle RMS has successfully retrieved two payloads and returned them to earth : the PALAPA and the WESTAR, two communication satellites which failed to function on their original deploy missions, on STS 51-A.

An equally important utilization of the Shuttle RMS has been in the assistance of payload flight experiments. The Shuttle RMS has been used to position payloads at specific data collection points. The Plasma Diagnostics Package (PDP) on STS 3 and the Induced Environmental Contamination Monitor (IECM) on STS 4 were positioned by the arm to take measurements of electric and magnetic fields and plasma characteristics in the environment of the Orbiter (PDP) and the concentration of particles and gases emitted by the Space Shuttle (IECM). In both cases the payload was maneuvered through a sequence of preprogrammed positions without releasing the payload from the arm. For other payload flight experiments, the Shuttle RMS was used to deploy and then later retrieve the payload experiment on the same mission. The SPAS (STS 7) and the SPARTAN (STS 51-G) fall into this later category.

The Shuttle RMS has also proved to be a valuable general purpose tool for observation, positioning astronauts, and applying a little shove at a critical point. The end-effector camera is routinely used for inflight visual inspections of payloads, Orbiter thermal tiles and second stage motor burns. A Manipulator Foot Restraint (MFR) has been attached to the arm to provide a stable platform for astronauts to repair satellites (Solar Maximum Satellite on STS 13, WESTAR/PALAPA on STS 51-A, and SYNCOM on 51-I) and to construct prototype Space Station truss structures (EASE/ACCESS experiment on ST 61-B). Finally, the Shuttle RMS has been used to push a stuck SIR-B antenna closed (STS 41-G), knock off an ice chunk which had formed coming out of the waste water dump nozzle (STS 41-D), and to hit a switch on the SYNCOM satellite (STS 51-D).

The Station MSC has been assigned the responsibility of playing the predominant role in the following Space Station functions: attached payload servicing (external), Space Station assembly, Space Station maintenance (external), transportation on the Space Station, deployment and retrieval, and EVA support.

To satisfy these responsibilities, the MSC must provide two new functions, which are not provided by the Shuttle RMS. The first is transportation. The MSC is required to transport payloads, EVA crew members, Space Station Program Elements and systems, Orbital Maneuvering Vehicles, and Orbital

Replacement Units (ORU's) to all locations as required to support Space Station and payload operations. This requirement drives the design to a manipulator with a base which can move up and down the truss. The Mobile Transporter is this moving base. The second requirement is dexterity. To play the predominant role in assembly, servicing and maintenance requires a device with much more finesse and dexterity than the Shuttle RMS. The problems with fine motions arise from the lightweight long flexible links comprising these manipulators. The design solution to meet this requirement is to develop a separate robotic device which will operate from the end of the manipulator arm. The Special Purpose Dexterous Manipulator (SPDM) is this robotic device. Although programmatically the SPDM is a separate flight element from the MSC, for the purposes of this paper, it will be included with the MSC system.

## 3. DESIGN CHARACTERISTICS

This section discusses the differences between the Shuttle RMS and the Station MSC in the areas of geometric configuration, mobility, sensor capabilities, control stations, control algorithms, handling performance, end-effector dexterity, and fault tolerance.

### 3.1 GEOMETRIC CONFIGURATION AND MOBILITY

The geometric configuration of the Shuttle RMS is shown in Figure 1. The arm is approximately 50 feet long with six in-line joints (shoulder yaw, shoulder pitch, elbow pitch, wrist pitch, wrist yaw, and wrist roll). All joints except the wrist roll have travel limits less than +/- 180 degrees. The effective reach envelope of the Shuttle RMS is approximately 35 feet from the base of the arm. The long booms are 12" diameter thin-walled tubes of composite material with internal stiffeners. The shoulder end of the Shuttle RMS arm is bolted to the Orbiter longeron and an end-effector is mounted onto the other end. The end-effector is the device that attaches (grapples) to the object to be manipulated, and uses a grapple fixture and an ingenious snare-wire device to rigidly attach to the grapple fixture. The grapple fixture is an 11" long pin with a nob on the end, attached to and protruding out from the payload. Affixed to the grapple fixture is a target pin, which lines up with the end-effector camera when the end-effector is properly aligned for capture. The grappling mechanism is also shown in Figure 1.

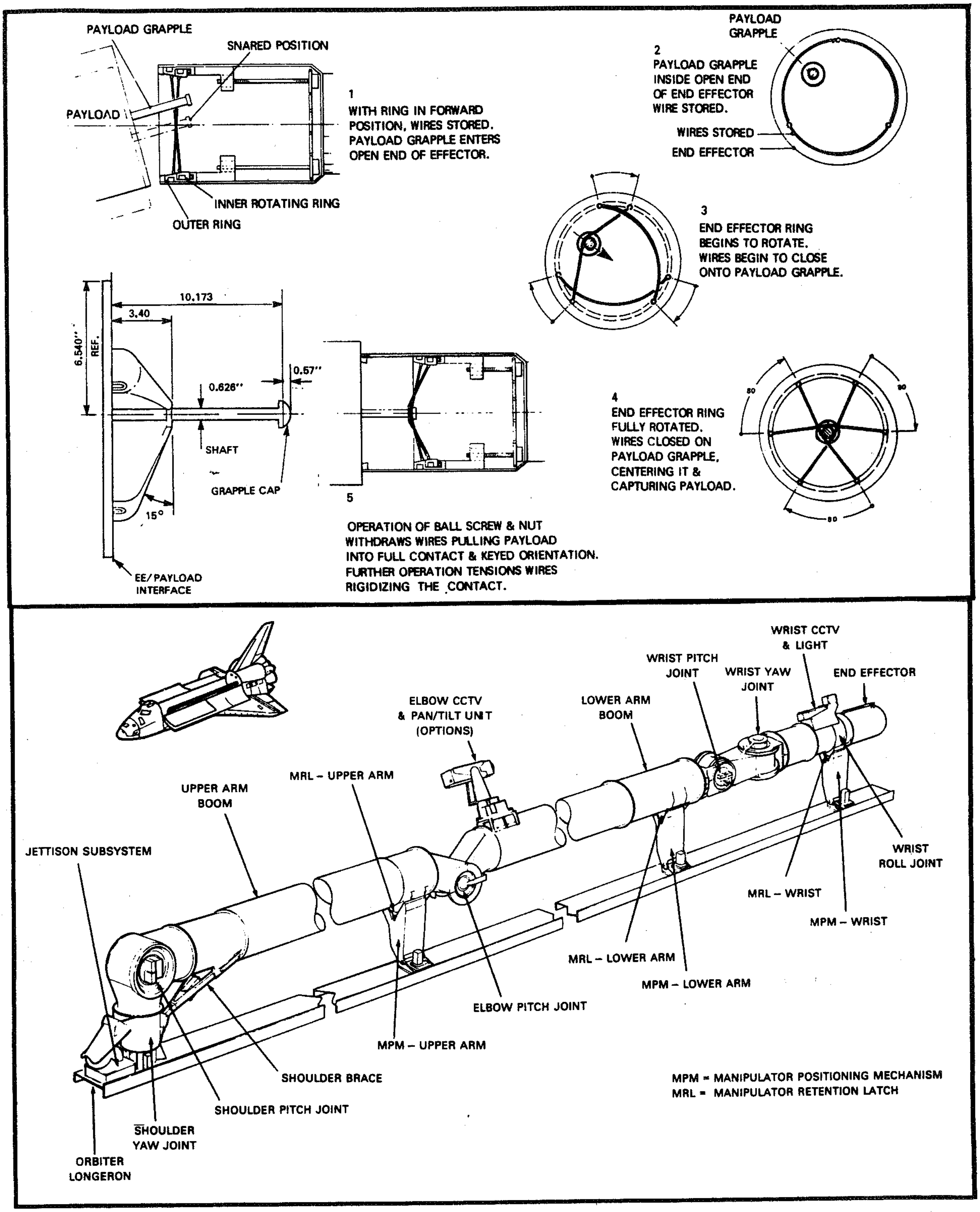

PAYLOAD GRAPPLE
SNARED POSITION
PAYLOAD
INNER ROTATING RING
OUTER RING
1
WITH RING IN FORWARD POSITION, WIRES STORED. PAYLOAD GRAPPLE ENTERS OPEN END OF EFFECTOR.
PAYLOAD GRAPPLE
2
PAYLOAD GRAPPLE INSIDE OPEN END OF END EFFECTOR WIRE STORED.
WIRES STORED
END EFFECTOR
3
END EFFECTOR RING BEGINS TO ROTATE. WIRES BEGIN TO CLOSE ONTO PAYLOAD GRAPPLE.
10.173
3.40
6.540" REF.
0.67"
0.626"
SHAFT
GRAPPLE CAP
15°
EE/PAYLOAD INTERFACE
5
OPERATION OF BALL SCREW & NUT WITHDRAWS WIRES PULLING PAYLOAD INTO FULL CONTACT & KEYED ORIENTATION. FURTHER OPERATION TENSIONS WIRES RIGIDIZING THE CONTACT.
4
END EFFECTOR RING FULLY ROTATED. WIRES CLOSED ON PAYLOAD GRAPPLE, CENTERING IT & CAPTURING PAYLOAD.
80
80
80
WRIST CCTV & LIGHT
WRIST PITCH JOINT
WRIST YAW JOINT
END EFFECTOR
ELBOW CCTV & PAN/TILT UNIT (OPTIONS)
LOWER ARM BOOM
MRL – UPPER ARM
UPPER ARM BOOM
JETTISON SUBSYSTEM
WRIST ROLL JOINT
MRL – WRIST
MPM – WRIST
MRL – LOWER ARM
MPM – LOWER ARM
ELBOW PITCH JOINT
MPM – UPPER ARM
SHOULDER BRACE
SHOULDER PITCH JOINT
SHOULDER YAW JOINT
ORBITER LONGERON
MPM = MANIPULATOR POSITIONING MECHANISM
MRL = MANIPULATOR RETENTION LATCH

Figure 1. Shuttle RMS Configuration

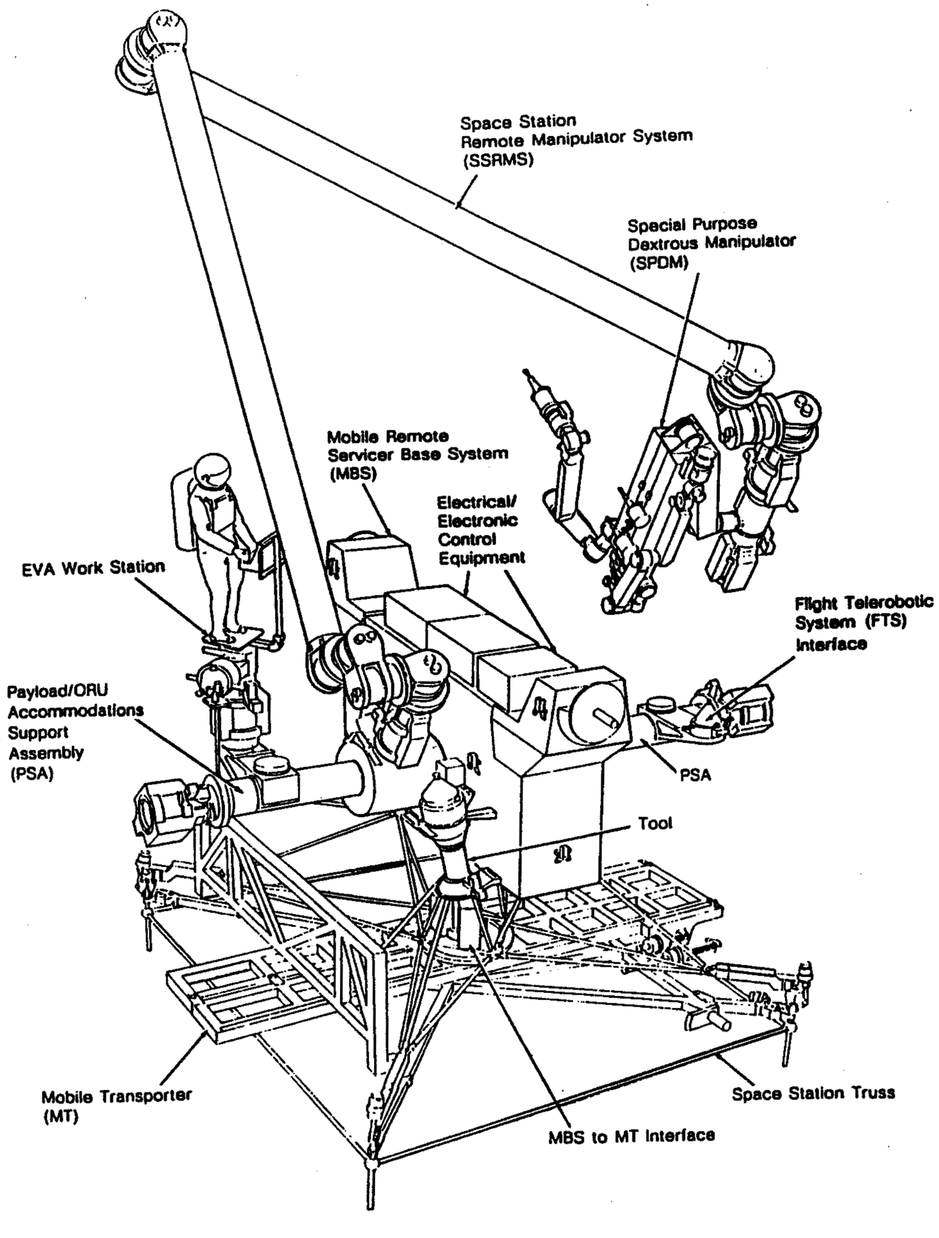


Figure 2. Space Station MSC Configuration

The Mobile Servicing Center is physically composed of four parts as shown above in Figure 2. At the bottom riding on the Station truss is the Mobile Transporter (MT). Sitting on top of the MT is the Mobile Remote Servicer (MRS) base. Attached to the MRS base through a power data grapple fixture (PDGF) is the MSC manipulator, the Space Station Remote Manipulator System (SSRMS). And finally, attached to the end-effector of the SSRMS is the Special Purpose Dexterous Manipulator (SPDM), the robotic end-effector for the MSC. The MSC can be operated with or without the SPDM.

The Mobile Transporter (with or without the MRS base on top) will be able to translate up and down, turn corners and change planes on the station truss. The MT itself will have an early role in the assembly of the Space Station Freedom, as it will be mounted in the Shuttle bay to hold and extend the truss assembly as each 5 meter bay is assembled.

The MSC manipulator is approximately 55 feet long with seven offset joints. It is symmetrical about the middle (elbow) joint with an end-effector on each end. These end-effectors will be similar to the current design, but will have additional structural latches and will transfer power and data across the interface. Either end-effector can attach itself to the power data grapple fixture on the MRS base. The offset joints will allow each joint to have +/- 270 degrees of travel.

The SSRMS has a relocatability feature. Either end-effector can attach itself to and operate from a power data grapple fixture located anywhere on the station. Current plans call for power data grapple fixtures mounted along the truss and perhaps also one mounted on one of the Space Station modules. This gives the MSC manipulator the ability to walk end over end down the truss in an "inchworm" motion. This type of mobility would preclude the carrying of payloads, however.

With the combined mobility of the Mobile Transporter and the SSRMS relocatability feature, the MSC reach envelope covers most of the Space Station external structure.

### 3.2 SENSORS

The Shuttle RMS has limited sensor data available. The motor tachometer rates and joint encoder angles are measured and fed back into the control algorithms. In the early Shuttle test flights, the Shuttle RMS was instrumented with strain gauges to provide load data at the shoulder and wrist, but the instrumentation was removed after the arm become operational. There are two cameras located on the Shuttle RMS, one at the elbow (with pan and tilt controls) and the other fixed at the wrist. Their views are displayed back to the Shuttle RMS operator at the Orbiter aft flight deck crew workstation, but there is no integration of the video data into the control loop except through the human operator.

The MSC manipulator will add force/moment sensing, vision sensing and vision data processing to the control loop. The force/moment sensor will be mounted at each end of the arm. Vision data will come from two cameras fixed at either end, with two more pan-and-tilt cameras mounted on the booms at the elbow joint, looking in opposite directions. The vision data will yield the state of the end-effector (position, attitude, rate). These additional sensors will reflect the advance of the state-of-the-art in manipulator control.

### 3.3 OPERATOR WORKSTATIONS

The Shuttle RMS is operated by a crewman at a station in the aft of the cabin, looking out through the rear windows onto the cargo bay. Control of the RMS is effected through two hand-controllers, one for translational motion and the other for rotational motion. The operator has control (pan, tilt, and zoom) of the closed-circuit television cameras (CCTV) located at each corner of the cargo bay (as well as special mission-specific locations). The operator also controls the lights illuminating the bay.

Another workstation is the Manipulator Foot Restraint (MFR), which is a platform grappled by the arm for suited crewpersons to "stand" on and work from while the RMS provides any necessary mobility. No RMS controls exist at this station, however, so all motion must be commanded by the operator in the Shuttle at the request of the crewperson on the MFR.

In contrast, the MSC will have numerous workstations. The operator will have a choice of controlling all the MSC's functions from the base of the arm, the tip of the arm, inside the station module, and from inside the Shuttle. The stations outside on the MSC will be referred to as extravehicular activity (EVA) workstations, while those used inside the pressurized environment are intravehicular activity (IVA) workstations. Of the IVA workstations, some are to be portable, enabling control of the MSC from the Shuttle (for possible berthing operations in which the MSC captures the Shuttle) or elsewhere.

### 3.4 CONTROL ALGORITHMS

The Shuttle RMS's design was mostly finalized in the late 1970's, involving more effort than any manipulator or robot built prior to that time. The RMS was also the first dexterous "space robot", and its control algorithms, while state-of-the-art in 1978, have become dated in the decade since. The main control algorithm involves commanding a desired rate for the payload. The joint rates necessary for this command then become the input rate command to each individual joint servo. Since the RMS has six joints to achieve 6 degrees of freedom at the end-effector, conversion from end-effector states to joint states is a simple matter. No true end-effector position control is possible, although a rate command designed to hold a joint position constant can be generated by the flight software. Holding joints still does not guarantee the end-effector will not move, however, since the joint encoders do not sense changes in the booms (like flexure or thermal warpage).

The MSC will have seven joints, which gives much more freedom of control while making that control much more complex. The method of converting from six end-effector degrees of freedom to the seven joint states requires input of some other constraint, usually minimization of some quantity like total kinetic energy in the arm.

Along with having seven joints, the MSC is expected to be able to sense and control end-effector position, rates, and forces. This will add greatly to the types of tasks which can be feasibly accomplished.

### 3.5 HANDLING PERFORMANCE

The Shuttle RMS was designed to handle payloads up to 65,000 lbm., while maintaining a tip positioning accuracy of 2 inches and one degree. A recent study sponsored by Johnson Space Center involving SPAR Aerospace and three independent contractors was conducted to determine the ability of the current RMS design to handle payloads outside of its design range (up to 250,000 lbm.). The results of this study, yet to be released, indicate the Shuttle RMS's performance is degraded by the increased mass, but not greatly. The greatest problems foreseen in such a situation stem from potential control system instabilities, where the position or rate of the payload oscillates about a desired point. All other handling characteristics were not seriously impacted, however.

The MSC's design range will extend to the massive payloads expected for the Space Station Freedom. The handling performance specifications are not known yet, but are expected to be more stringent since the improved sensors will enable higher-resolution control. The seven-joint arrangement will aid in obstacle avoidance, since an infinite number of arm configurations is available for a particular end-effector position and attitude.

Another improvement in the performance will come from the collision avoidance algorithms expected to be implemented in the control scheme. This should allow the operator to concentrate on the task, and pay less attention to the arm's configuration. The collision avoidance algorithm is expected to be based on a world model of the environment, rather than proximity sensors.

### 3.6 END-EFFECTOR DEXTERITY

The RMS's end-effector position and rate are determined by the joint encoders and tachometers. This causes problems, as stated previously, because using joint data assumes perfectly constant links connecting the joints, i.e. no change from the ideal. This is not generally true for space manipulators, because of their relatively flexible lightweight links.

The MSC manipulator control scheme will utilize vision data to determine the actual position and rates of the end-effector, and will close the loop around this information, rather than around the individual joints. This method will lead to an improvement in the accuracy of the end-effector's trajectory, and therefore its dexterity.

In addition to this improved positioning accuracy, even more dexterity will be obtained using the Special Purpose Dexterous Manipulator (SPDM). This item will be a tool attached to the end-effector (or elsewhere) which will use two small seven-joint manipulators with cameras to perform tasks which require highly precise motion, like module changeouts, adjustments, testing, and cleaning.

### 3.7 FAULT TOLERANCE

Fault tolerance is a measure of the ability of a system or component to function despite failures of subsystems. Redundant and/or backup subsystems are used to ensure this tolerance. A system may be fail-safe, which means a safe shutdown (not an out-of-control machine), or it could be fail-operational, which means operation of the device may continue (perhaps in a degraded mode) despite that failure.

A weak point in the design of the Shuttle's RMS lies in its fault tolerance. The RMS has some redundant equipment and backup control modes which can be used in event of certain failures, however unrecoverable failure scenarios do exist. The end-effector has several failure points which can completely disable it, effectively disabling the arm. This is because the RMS was designed to be fail-safe, instead of fail-operational.

The MSC, however, is expected to be designed to a philosophy of one-fault tolerance as a minimum, which implies a functional redundancy of two (two redundant strings of functional elements). Any functional capabilities of the MSC which may be essential to crew safety or Space Station survival shall be two-failure tolerant, i.e., the system will be able to continue operation after two failures, and the third failure will cause a safe shutdown.

## 4. CONCLUSIONS

The Shuttle RMS was the world's first manipulator designed and tested almost completely by computer. Its first flight on STS-2 was the first time it was able to operate, since it was too weak to operate under gravity. While the technology of robotics today has surpassed the RMS's mid-seventies design, much has been learned from this first manipulator. Armed with this experience, the United States and Canada are expected to jointly produce a final design which will serve the needs of the Space Station Freedom for years after its construction. The design, construction and operation of the MSC will yield valuable experience in the field of space robotics and control. Perhaps even more important are the potential benefits to ground-based robotic technology, artificial intelligence and information processing which are bound to result from the research required for this ambitious project.

# DEXTEROUS MANIPULATOR FLIGHT DEMONSTRATION

Edward L. Carter
Lockheed Engineering & Sciences Company
Houston, Texas

## ABSTRACT

The Dexterous Manipulator Flight Experiment, an outgrowth of the Dexterous End Effector (DEE) project, is an experiment to demonstrate newly developed equipment and methods that make for a dexterous manipulator which can be used on the Space Shuttle or other space missions. The goals of the project, the objectives of the flight experiment, the experiment equipment, and the tasks to be performed during the demonstration are discussed.

## 1.0 INTRODUCTION

The DEE project is managed out of the Technology and Commercial Projects Office of the NASA Johnson Space Center (JSC) New Initiatives Office. The project, with its precursors, began almost 5 years ago as an effort to develop a force torque sensor (FTS) for the Shuttle Remote Manipulator System (RMS). As it currently exists, the DEE project includes a flight experiment and other development activities. After a brief overview of the project goals and background, this paper will focus primarily on the flight experiment.

### 1.1 PROJECT GOALS

DEE project goals are (1) to gain experience in operation of the RMS with the FTS system; (2) to develop an end effector with improved capability for performing space operations in less time and with greater facility; (3) to develop an improved target and alignment system; and (4) to gain experience with the hardware and software developed.

### 1.2 PROJECT BACKGROUND

The magnetic end effector (MEE) was conceived and developed at JSC. Since the first tests of the MEE/FTS concept demonstration prototype in September 1987, the DEE project has operated periodically at the Manipulator Development Facility (MDF). The Targeting and Reflective Alignment Concept (TRAC) system was developed shortly after the MEE prototype was first used and has been employed in almost all of the MDF operations with the MEE and FTS. Each time a new feature was added to the MEE, MEE grapple plate, or the TRAC system and each time a new procedure was developed, this change was checked out and demonstrated. These demonstrations have been used to prove new capabilities of the tools, as well as to familiarize interested people with the work being done.

### 1.3 OBJECTIVES OF THE FLIGHT EXPERIMENT

The objectives of the flight experiment are the following:

a. To demonstrate constrained motion of the RMS with the use of the FTS to control the loads imposed upon the arm.

b. To demonstrate TRAC mirror docking with the RMS in zero-gravity dynamics.

c. To demonstrate magnetic grappling and ungrappling in space conditions on fixed payloads. This will determine the proximity required before the magnetic attraction overpowers the flexibility of the RMS, and the dynamic effect when the magnets close the final air gap and impact the grapple fixture on the payload.

d. To demonstrate the right-angle TV camera as an aid in docking and manipulating payloads.

e. To demonstrate the use of a magnetic holding fixture for cases where temporary stowage of a payload may be necessary.

f. To determine the RMS control resolution, which possibly is better than indicated by the published data.

## 2.0 GENERAL DESCRIPTION OF THE FLIGHT EXPERIMENT

The DEE flight experiment is a longeron-mounted payload (see figure 1), a portion of which is deployed when grappled by the RMS using the standard end effector (SEE). The DEE therefore does not affect the standard configuration of the RMS or any other payload using the RMS.

### 2.1 FORCE TORQUE SENSOR

The FTS is a complex system which provides six-axis force data to the RMS operator. A full description is beyond the scope of this paper; a few key features, however, are described as follows. The FTS is in two parts – the part in the payload bay (on the MAT), and the part in the aft flight deck (AFD). These are connected by the RMS special purpose end effector (SPEE) cables. The part in the payload bay is the FTS data collection assembly (DCA); and the portion in the AFD is the display electronics assembly, which consists mainly of the SC-1D computer and the video graphics generator (VGG). The monitor display of the VGG output is shown in figure 2.

### 2.2 MAGNETIC ATTACHMENT TOOL (MAT)

The MAT (see figure 3) is the assembly which is grappled by the SEE on the RMS for experiment operation. It is mounted on a longeron-mounted carrier for launch and landing. The magnetic attachment tool is made up of the MEE, the FTS data collection assembly (DCA), and the electrical flight grapple fixture (EFGF). There is also an adaptor, or spacer, between the FTS and the EFGF.

### 2.2.1 MAGNETIC END EFFECTOR

A structural housing contains the various MEE subsystems (see figure 4). The primary subsystems are the two magnet assemblies, the two TV cameras, the backup batteries, and the alignment pins. In addition, there are switches, indicators, camera lights, heaters, control circuit boards, and a TV remote select switch. MEE specifications are shown in the table below.

| MEE Specifications | |
|---|---|
| Holding force in tension | 1500 lb |
| Shear force (magnets alone) | 50 lb |
| Shear force (with alignment pins) | 750 lb |
| Torque | 700 ft-lb |
| Over turning moment | 530 ft-lb |
| Diameter | 11.38 in. |
| Length | 10.50 in. |
| Weight | 38 lb |

2.2.1.1 Electromagnets. The two magnets are U-shaped, with three separate coils on each. One is a high powered pull-in coil which produces an appreciable attractive force with a large air gap, and which is automatically switched off by the preload indication system after grapple has been achieved. The other two are hold-in coils and are identical, with each producing sufficient magnetization to saturate the core and thus develop the full rated holding performance of the MEE. The two hold-in coils are connected to separate power sources for redundant operation. The magnets are arranged with the pole faces within a 7.0-in. square footprint; they are independently mounted on a spring suspension system in such a way that the poles move slightly toward the grapple fixture during the grappling process. This motion is detected by microswitches as an indication of preload. The use of the springs does not reduce the attractive force, but rather ensures that a preload exists across the grapple interface.

2.2.1.2 TV Cameras. Two TV cameras are mounted in the MEE – one on the MEE centerline and the other normal to the centerline. The cameras are used only for targeting and alignment; thus they are preset to a fixed focus distance, and the lens apertures are also preset. Supplementary incandescent lighting is provided during the closeup portion of the targeting and alignment process. Since only one camera output can be utilized at a time, a remote camera selector switch (Government-furnished equipment, previously qualified) is used to power up and select the output of the proper camera.

2.2.1.3 Battery Backup. A failure condition of the RMS exists whereby the electrical connector at the EFGF can become disconnected, thus disconnecting the MEE from all Shuttle power and from all controls. The MEE must not release a grappled payload because of this failure. To accommodate this possible situation, the MEE is equipped with two 18-V battery backup systems, each of which independently powers two of the magnet hold-in coils. The MEE can therefore survive two failures without danger of an inadvertent release of a grappled payload.

2.2.1.4 Alignment Pins. The MEE is designed with two spring-loaded alignment pins which ensure accurate alignment and provide increased capability for shear and torsion loads. Microswitches detect the fully out position of the pins.

### 2.2.2 FTS DATA COLLECTION ASSEMBLY

The DCA, which is located in the payload bay, consists of the sensor element (see figure 5) and the data collection electronics (DCE).

### 2.2.3 ELECTRICAL FLIGHT GRAPPLE FIXTURE

The EFGF is a piece of standard STS-provided equipment. For this flight experiment it will be modified by removing a portion of the target plate to improve visibility around the EFGF when the TRAC system is used with the RMS wrist TV camera.

## 2.3 GRAPPLE FIXTURE

The grapple fixture for the MEE is simply a 7.5-in. square metal plate. Figure 6 shows a generic grapple fixture which could be attached to any payload; the base would be designed for or integrated into specific payloads. (The grapple fixture used by the DEE flight experiment is slightly different.) The material must be magnetically soft; and for maximum performance in load-critical applications, permendure is the preferred material. The thickness of the grapple fixture for the DEE flight experiment is 0.6 in. The surface of the grapple fixture is used as a mirror in the TRAC system and is polished to the point of producing a good spectral reflection. The surface is also fitted with a TRAC target pattern. Holes for the alignment pins are also provided.

## 2.4 EXPERIMENT STOWAGE AND ACTIVITY PLATE (ESAP)

The ESAP (see figure 7) is the device that supports the components of the DEE flight experiment during launch and landing via two latch assemblies. In addition, it provides two sockets and a grid, which are used in carrying out the experiment operations.

### 2.5 TASK BAR

The task bar, a short panel structure as shown in figure 8, is the device to which the MEE grapples and manipulates during the task operations. One end of the task bar simulates a heat pipe panel, and the other end simulates a module servicing tool (MST).

### 2.6 AFT FLIGHT DECK INSTALLATION

The installation in the AFD consists of parts of the FTS, one-half of a standard switch panel, interconnecting cables, and some standard Orbiter equipment.

### 2.7 TARGETING AND REFLECTIVE ALIGNMENT CONCEPT

The TRAC system uses a TV camera viewing its own image in a mirror on the grapple fixture (or on an offset surface) to achieve alignment in all six axes. The TRAC consists of a TV camera and a TV monitor (both with alignment marks) and a mirror/cross hair assembly (see figure 9). Mirror/cross hair assemblies are located on objects to be grappled and areas to be targeted. The system can be utilized with the centerline camera, the right-angle camera, or the RMS wrist camera.

## 3.0 EXPERIMENT OPERATION

The task operations for the flight experiment include the following:

a. Radiator panel replacement / rotating panel task

b. Magnetic hold down task

c. MST simulation task

d. RMS control resolution task

### 3.1 INITIAL HARDWARE CHECKOUT

The RMS is powered up and uncradled, and the SEE is placed in the vicinity of the MAT. The RMS operator then aligns the SEE with the MAT's EFGF. After alignment is complete, the latch assembly electromagnets are energized, and upon loading verification, the mechanical latches are released. Then the SEE grapples the MAT. MAT operational capability is now verified. Operational verification consists of turning on the FTS and performing a "health check." MAT TV cameras, lights, and electromagnets are turned on. When the MAT operational verification is complete, latch assembly electromagnets are turned off and the RMS moves away from the latch assembly (see figure 8) and the ESAP. Once the RMS is configured, the experiment tasks begin. In each of the following tasks, the task is initially performed without the operator using the FTS. The observer crewmember monitors the FTS output and alerts the operator if the loads exceed a predetermined value. The task is then repeated with the operator using the FTS. The differences between the two sets of data are examined in the postflight analysis.

### 3.2 TASK BAR GRAPPLE

Using TRAC for alignment, the MAT is magnetically grappled to the task bar located as shown in figure 8. The task bar is then released from the experiment carrier and its latch assembly, and is translated to approximately 7 ft above the ESAP structure. A wrist roll is commanded, and the task bar is perpendicular to the ESAP.

### 3.3 RADIATOR PANEL REPLACEMENT / ROTATING PANEL TASK

The RMS is translated to the rotating panel task area. Using the TRAC mirror and MAT right-angle TV camera, the task bar is aligned with the mating slot. After alignment, the FTS point-of-resolution (FTS-POR) is changed to reflect the new MAT orientation. With the correct FTS display showing and being monitored, and TRAC alignment maintained as shown by the right-angle view, the task bar is inserted into its mating slot. The task bar is adjusted such that the FTS-displayed forces and torques are minimized and remain below a predetermined threshold. Full insertion is detected by monitoring the digital readouts on the RMS display and control panel and by observing a low-magnitude compressive force on the FTS display. A clockwise roll (wrist camera point-of-view) will be performed (see figure 10) in multiple, small-degree segments, so that the loading on the task bar can be monitored, up to approximately 30°. The task bar is then returned, using small-degree increments, to the vertical position. During the rotation, the calculated torque required to rotate the panel will be compared to the FTS-measured torque. This task will be repeated for a counterclockwise roll of 30°.

### 3.4 MAGNETIC HOLD DOWN TASK

The task bar is removed from the slot and temporarily restowed on its latch assembly. The MAT then releases the task bar, leaving it on the latch assembly with only the electromagnets holding the task bar. This demonstrates the magnetic hold down task. Next, the MAT is rolled 180° and regrappled to the task bar. The task bar is released from the ESAP and the FTS-POR is changed to reflect the new orientation of the FTS.

### 3.5 MODULE SERVICING TOOL SIMULATION TASK

Simulation of the MST operations begins with the MAT grapple of the task bar and the subsequent wrist roll of the task bar to the vertical position. Using the corresponding TRAC target, the task bar probe is aligned with the receptacle and inserted into the receptacle while forces and torques exerted on the task bar are minimized as before. Several loading cases may be examined as time permits.

### 3.6 RMS CONTROL RESOLUTION TASK

Once the MST simulation is completed, the task bar is returned to its original stowage configuration, using the TRAC system for alignment and the FTS for force and torque minimization. The task bar latch assembly electromagnets are powered up, and the preload indicator is checked. When loading has been verified, the MAT releases the task bar and the latch assembly rigidizes its grapple using the mechanical latches. The latch assembly electromagnets are turned off when latching is complete and verified.

The MAT is repositioned at the control resolution grid and is aligned with the basic TRAC pattern. Several motion types will be examined – specifically, those involving translation (Y and Z axes) and rotation (wrist roll). The commanded motion versus actual RMS response will be compared in the postflight data analysis.

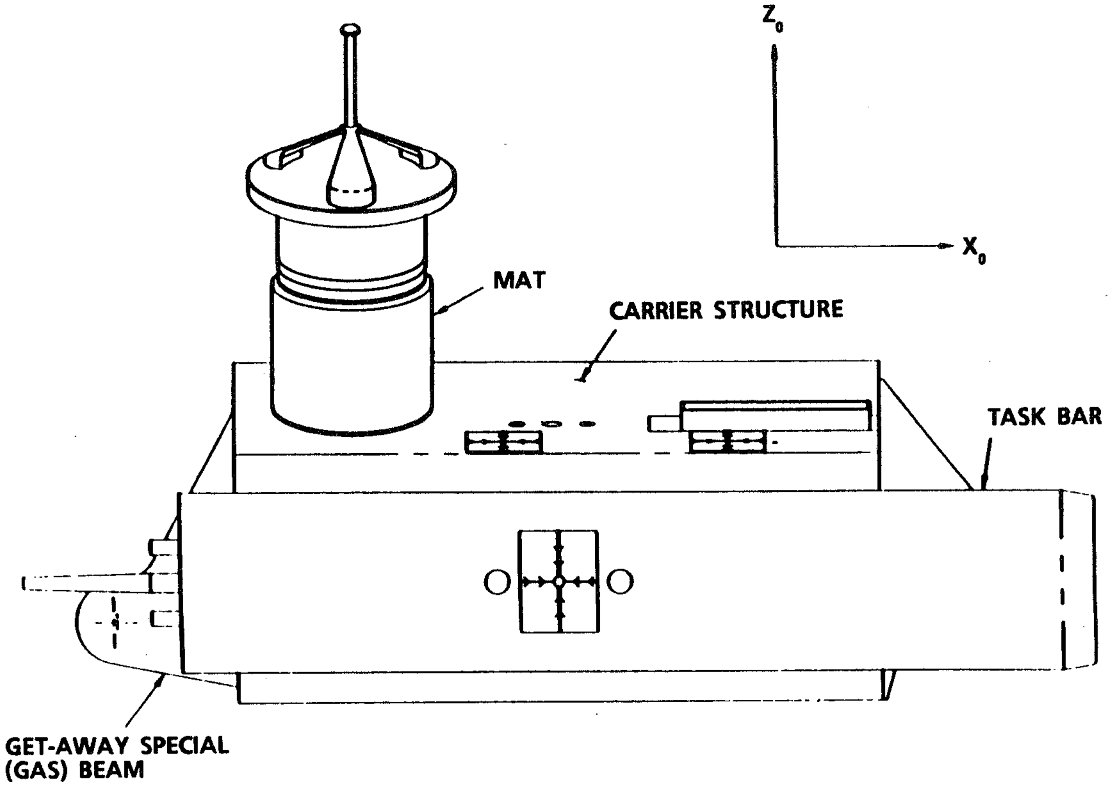
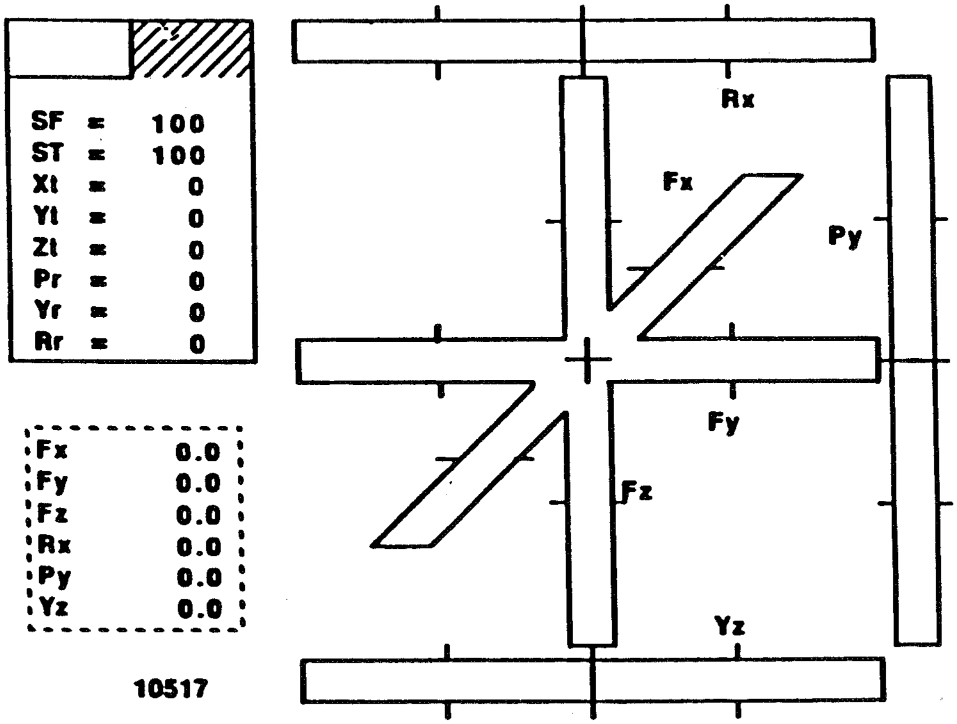


Figure 1.- DEE flight experiment launch configuration.


SF = 100
ST = 100
Xt = 0
Yt = 0
Zt = 0
Pr = 0
Yr = 0
Rr = 0

Fx 0.0
Fy 0.0
Fz 0.0
Rx 0.0
Py 0.0
Yz 0.0

10517

Figure 2.- Force-torque display layout.

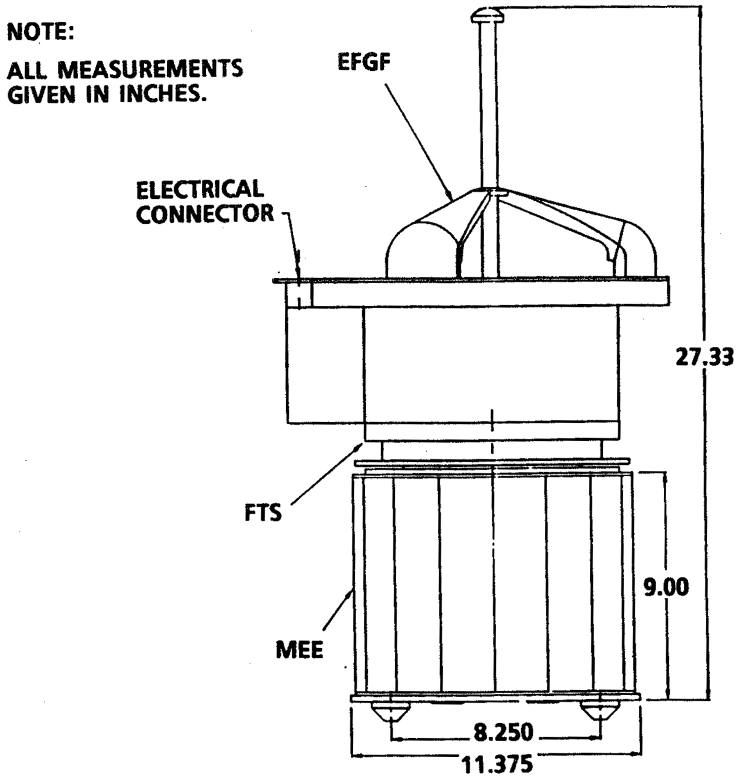


Figure 3.- Magnetic attachment tool (MAT).

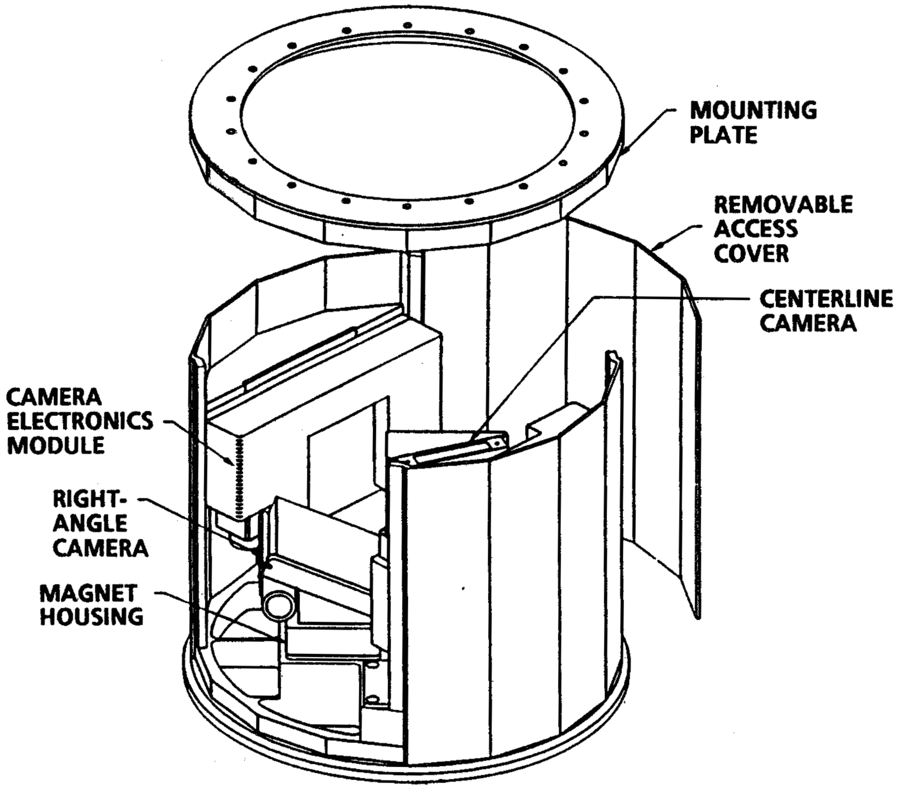


Figure 4.- Magnetic end effector (MEE).

**LOAD PATH: ARM → RING → DEFLECTION SYSTEM → CENTER BLOCK → END EFFECTOR.**

**SAFETY FEATURE: PINS SHARE OVERLOAD OR HOLD LOAD IF FLEXTURE SHOULD BREAK → FAIL SAFE**

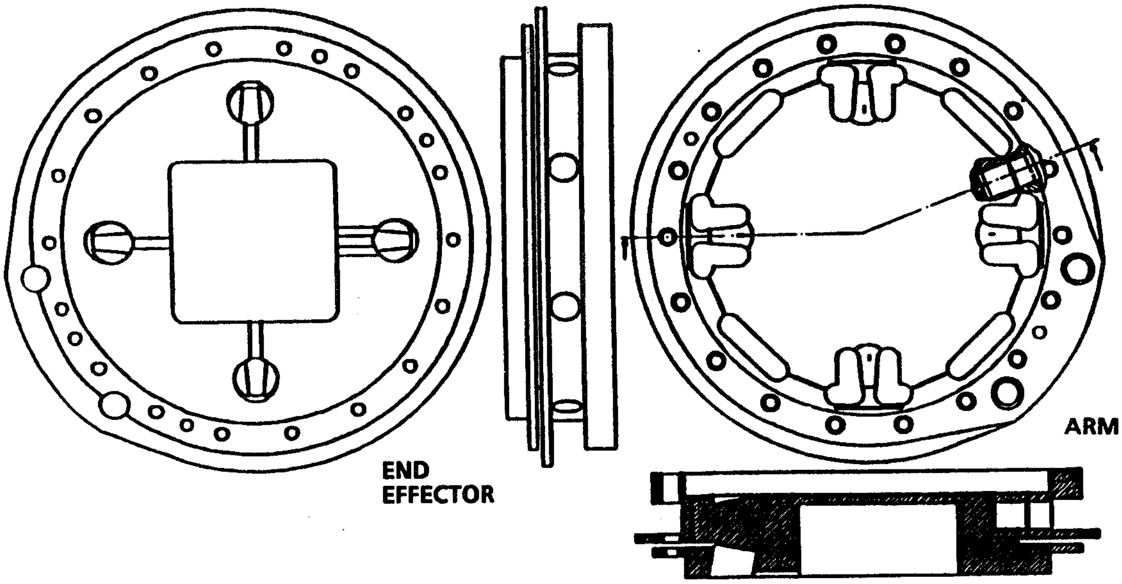


Figure 5.- FTS sensor element – mechanical design.

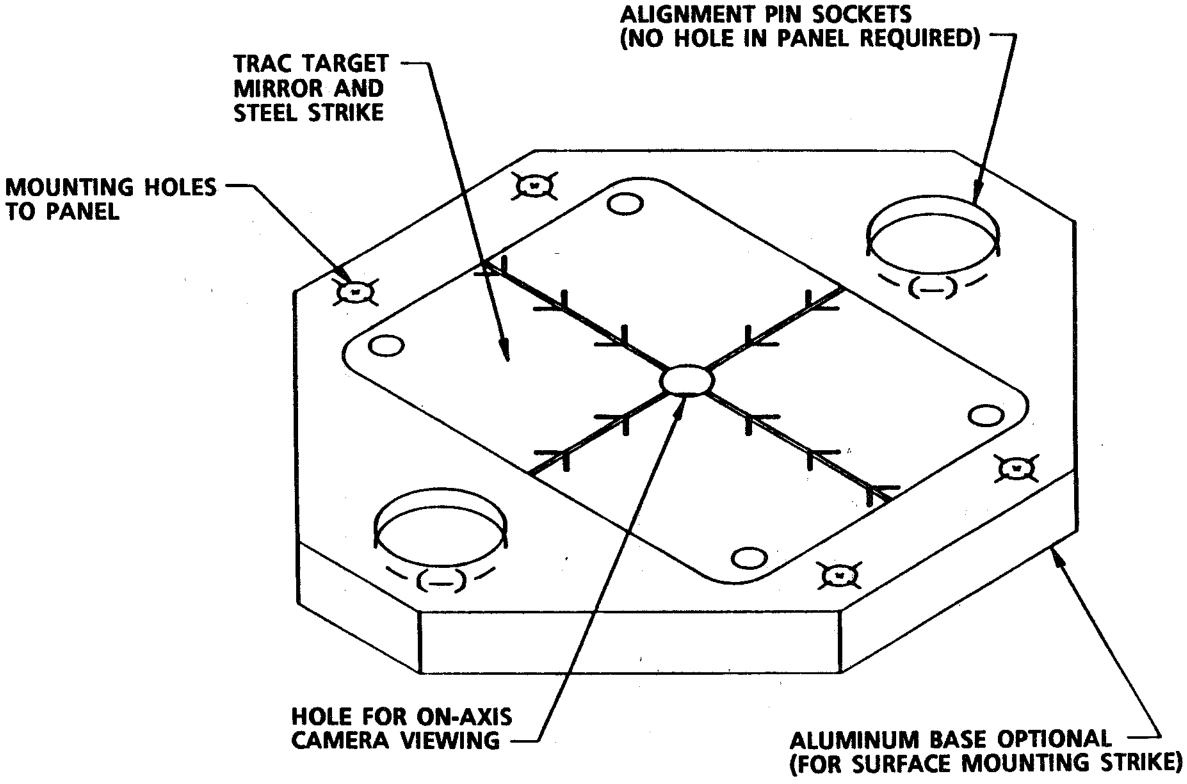

ALIGNMENT PIN SOCKETS (NO HOLE IN PANEL REQUIRED)
TRAC TARGET MIRROR AND STEEL STRIKE
MOUNTING HOLES TO PANEL
HOLE FOR ON-AXIS CAMERA VIEWING
ALUMINUM BASE OPTIONAL (FOR SURFACE MOUNTING STRIKE)

Figure 6.- Grapple fixture.

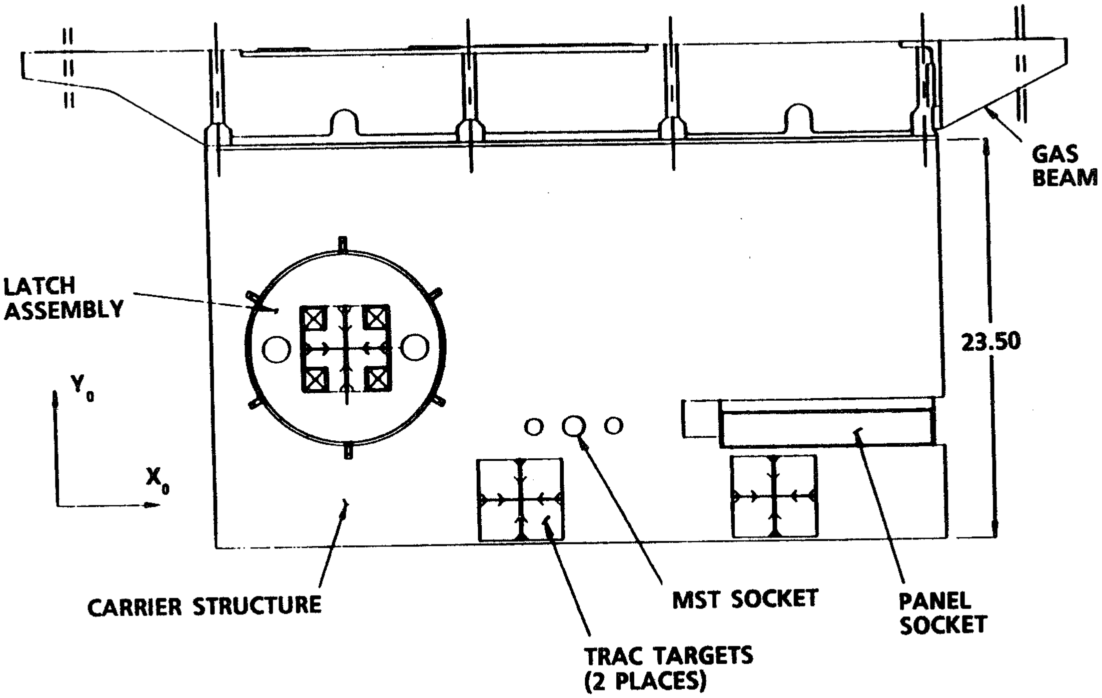


Figure 7.- Experiment stowage and activity plate (ESAP) plan view.

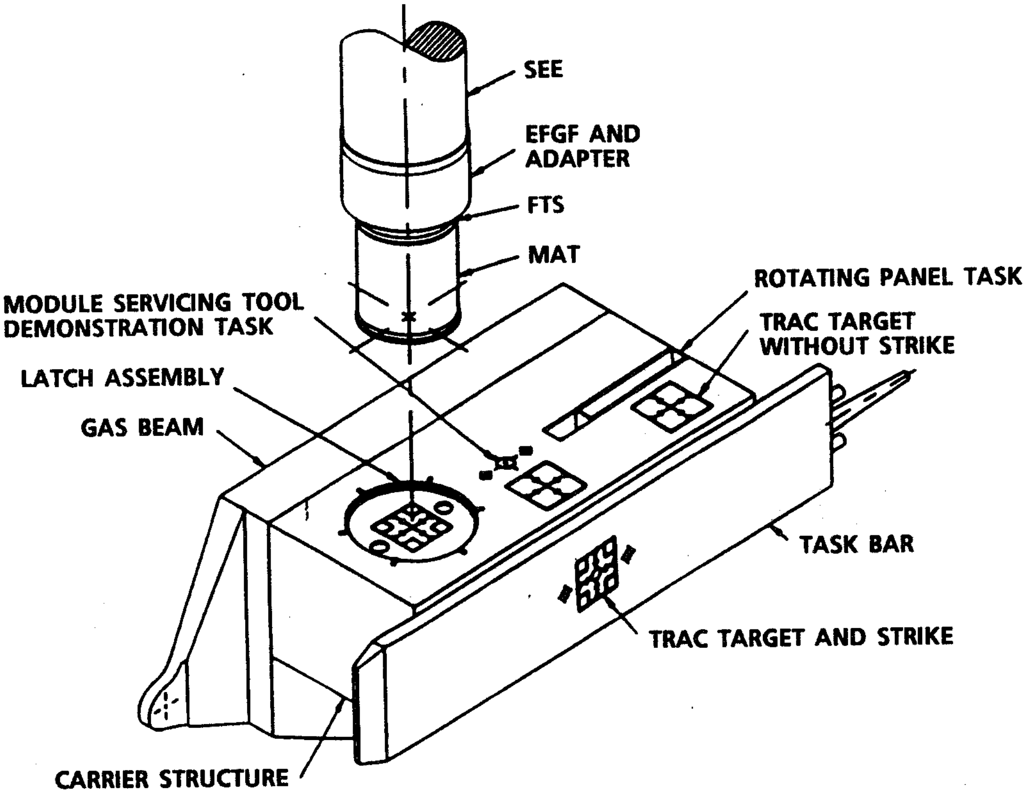


Figure 8.- MAT removed from the ESAP.

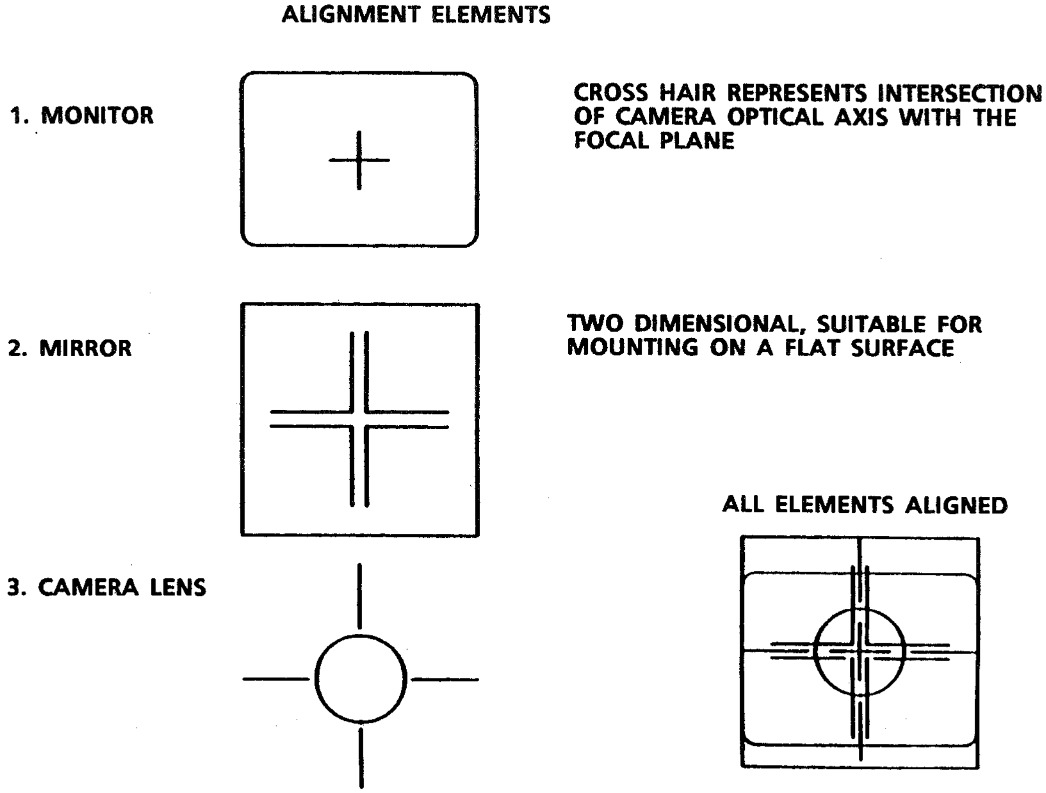

ALIGNMENT ELEMENTS
1. MONITOR
CROSS HAIR REPRESENTS INTERSECTION OF CAMERA OPTICAL AXIS WITH THE FOCAL PLANE
2. MIRROR
TWO DIMENSIONAL, SUITABLE FOR MOUNTING ON A FLAT SURFACE
ALL ELEMENTS ALIGNED
3. CAMERA LENS

Figure 9.- Targeting and Reflective Alignment Concept (TRAC) targets.

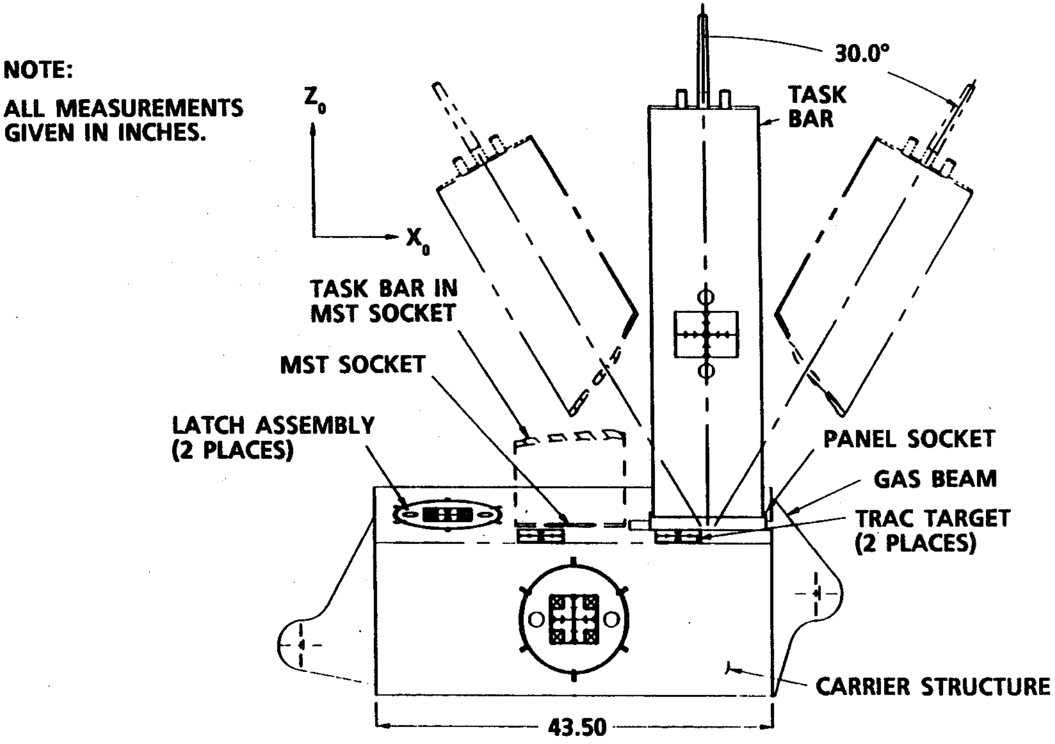


Figure 10.- Radiator panel replacement / rotating panel task – view looking starboard.

# An Intelligent, Free-flying Robot

G. J. Reuter, C. W. Hess, D. E. Rhoades, L. W. McFadin, K. J. Healey, J. D. Erickson
NASA Johnson Space Center Houston, Texas 77058
and
D.E. Phinney
Lockheed Engineering and Sciences Company Houston, Texas 77058-3711

## ABSTRACT

The ground-based demonstration of EVA Retriever, a voice-supervised, intelligent, free-flying robot, is designed to evaluate the capability to retrieve objects (astronauts, equipment, and tools) which have accidentally separated from the Space Station. The major objective of the EVA Retriever Project is to design, develop, and evaluate an integrated robotic hardware and on-board software system which autonomously: (1) performs system activation and check-out, (2) searches for and acquires the target, (3) plans and executes a rendezvous while continuously tracking the target, (4) avoids stationary and moving obstacles, (5) reaches for and grapples the target, (6) returns to transfer the object, and (7) returns to base.

## 1. INTRODUCTION

Space Station Freedom advanced automation and robotics has been the subject of numerous symposia and papers [1,2]. Appropriate roles for humans and machines in an evolving mix have been highlighted as a specific goal, with supervised intelligent system designs as ways to meet the needs of appropriate flexible-capability automation and robotics, thereby giving people-amplifier-type productivity gains. These roles are extremely important. New role definitions are enabled by symbolic processing and machine intelligence approaches to software, which also gives an ability to earn human trust while evolving in demonstrated reliable and capable operation.

The concept of supervised, intelligent, autonomous robotics provides for autonomous behavior of an intelligent type where human control is normally at a high level of goal-setting and involved in mixed initiative communication as a means of implementing decentralized, delegated management. By contrast, telerobotics provides a partially automated remote extension of human task performance with occasional control delegation for specific parts of tasks given to the telerobot for efficiency reasons. Teleoperation and telepresence provide remote extension of human task performance with the human essentially always in the loop.

This paper presents the need for extravehicular activity retrieval of objects and a potential solution in the form of a supervised, intelligent, free-flying space robot. An overview of a 3-year, 3-phase ground demonstration project is given , as are the eventual characteristics of the EVA Retriever. A description of the Phase I robot is given and Phase I results from an air-bearing floor demonstration are discussed. The Phase II software design is presented, including systems engineering studies of requirements, sensor-controlled motion based on real-time updating of a dynamic world model, and elements of the sensing, perception, reasoning and planning, action, and performance measurement.

## 2. THE NEED

Due to the extensive extravehicular activity (EVA) operations required by Space Station, there is a finite separation probability for astronauts, even when normally tethered, and for equipment and tools. A glove and camera have been separated and not retrieved in space operations previous to Space Shuttle, a tethered torque-wrench was accidentally separated on STS 51A, and other small item losses and near-misses have occurred.

The Space Station cannot chase separated crew or equipment even though crew safety is top priority. Other vehicles such as the Space Shuttle orbiter or orbital maneuvering vehicle will not usually be available. Many hours of real-time simulation of manned maneuvering unit (MMU) retrievals indicated short response time was critical and major risk to a second astronaut was involved, which was not acceptable. Equipment may be too valuable to lose because it is required in operations and replacement is not available on the station. There is also collision potential on later orbits which, though small, has occurred previously. The Space Station Program is considering making this retrieval a requirement.

## 3. POTENTIAL SOLUTION

A mobile (free-flying) space robot offers a potential solution. This might be teleoperated, but the quicker response and greater productivity of a supervised, intelligent, autonomous robot was judged to be the best solution if it could be made available in practical terms. However, significant technology advances will be necessary before even this simple, crucial application can be practically addressed. These advances will only be gained by implementing autonomous robot simulations and testbeds so as to gain experience with the developing technology.

Several previous efforts have laid a foundation for autonomous robot development including Shakey [3], JASON [4], the RPI Rover [5], the JPL Rover [6], and the Stanford Cart [7], among others. These first-generation autonomous robots were used to explore basic issues in vision, planning, and control. However, they were all seriously hampered by primitive sensing and computing hardware. More recent efforts have overcome many of these limitations, and very sophisticated second generation autonomous robot testbeds have evolved. Some of these efforts include the developments of HILARE [8], the FMC Autonomous Vehicle [9], the Autonomous Land Vehicle (ALV) [10], the various CMU mobile robots [7], and the Ground Surveillance Robot (GSR) [11]. A more general and complete discussion of autonomous vehicle history and technical issues has been given by Harmon [12]. While operational versions don't exist, much advantage can be obtained from these efforts.

By comparison, the space retrieval task seems simpler in some respects. While automatic control, such as is available in automatic guided vehicles (AGV), remotely piloted vehicles (RPV), and missiles, is not adequate here due to the dynamic environment, the more general solutions to vision and planning in completely unknown environments are not required. There are few objects in space; these are cooperative, and largely knowable. Supervision by voice is a natural, flexible means of providing the primary human-machine interface (supplemented with helmet displays) required. This requires limited natural language understanding integrated with the environment and task as well as functions like planning and reasoning. Complete intelligent autonomy of the R2D2/C3P0-type is not required nor achievable.

The free-flying space robot would operate near a spacecraft such as the Space Station in a primarily voice-supervised, autonomous mode for mobility and manipulation. It is intended to be an evolutionary system improving in capability over time and as it earns crew trust through reliable operation. It will operate in a dynamic much less well-structured environment than current industrial robots. Most planned actions cannot be tested except at execution. There is little repetition in its actions in the short term. Its sensing and perception provides real-time updates to a dynamic "world" model which is the basis of plans and actions. Knowledge by the EVA Retriever of its own past experience is intended through an episodic memory and retrospective processes such as summarization. Self-awareness is provided through sensing of internal states such as manipulator joints and health from fault detection and diagnosis with impact on planning. An intelligent human-machine interface with speech recognition, limited natural language understanding based on "state-change semantics" and voice synthesis is intended. EVA Retriever and crew will often cooperate in the same work envelopes. Safety, reliability, robustness, and maintainability in space are key attributes.

## 4. GROUND DEMONSTRATIONS

The EVA Retriever technology demonstration was established to design, develop, and demonstrate an integrated robotic hardware/software system which supports development of a space borne crew rescue and equipment retrieval capability through a phased set of ground-based simulations and physical demonstrations and evaluations. A Space Shuttle flight experiment would be a needed and plausible following step, preceding any space operations-related efforts.

Goals for each phase of a three phase project were established [13] in support of the overall goal of building and evaluating the capability to retrieve objects (astronauts, equipment, and tools) which have accidentally separated from their spacecraft. The Phase I goals were to design, build, and test a retriever system testbed by demonstrating supervised retrieval of a fixed target. Phase II goals are to initiate simulations and to enhance the testbed subsystems with significant intelligent capability by demonstrating target retrieval with avoiding of fixed obstacles. Phase III goals are to more fully achieve supervised, intelligent, autonomous behavior by demonstrating retrieval of a moving target while avoiding moving obstacles.

## 5. PHASE I DESCRIPTION

Phase I consisted primarily of an integration of the hardware and software into a functional system and subsequent demonstration of certain features of the intended behavior in simple form on the JSC Precision Air Bearing Floor (PABF), namely, supervised retrieval of a fixed target.

The integrated testbed free-flyer (Fig. 1) is an anthropomorphic manipulator unit with dexterous arms and hands built from a modified qualification test unit of the manned maneuvering unit (MMU) for mobility, Remote Technology Corporation dual 6-degree-of-freedom manipulators, a 3-fingered hand developed by the JSC Crew and Thermal Systems Division (CTSD), Inmos transputer 10-MIP processors, a Votan speech recognition and voice synthesis system, a 3-D laser imager from Odetics, a video camera and tracker processor by McDonnell Douglas, a robot body built at JSC, an arm and hand open-loop electronic control system, and software built by two JSC divisions. The vision sensors have 60 by 60 degree fields of vision.

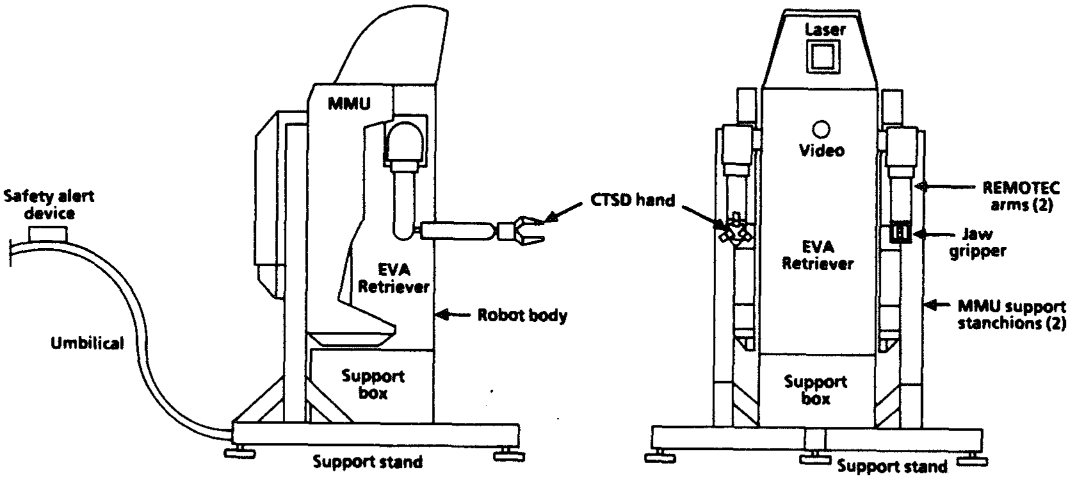


Figure 1. Retriever test article.

Onboard software includes: (1) supervisory activation and monitoring, (2) simple predefined plans for rendezvous, station keeping, and grappling, (3) plan sequencing and execution with sensory feedback in a benign, initially unknown environment, (4) sequential robotic movements of the MMU, arms, and hands, (5) supervisory interruption, direct control, and resumption of autonomous sequences, and (6) sensor fusion for rendezvous tasks.

Primary communication and control of the EVA Retriever is performed by voice commands. The testbed voice commands are: (1) activate and quick activate, (2) search (parameter: astronaut, tool, home, generic), (3) rendezvous, (4) reach, (5) grapple, (6) wait, (7) manual, and (8) shutdown. The EVA Retriever provides pre-recorded voice responses based on its sensory data and status. The response options to the voice commands are: (1) activating, and ready to search, (2) searching for target, target not found, tracking target, and ready to rendezvous, (3) rendezvous, or rendezvous failure, (4) closing hand, (5) wait acknowledged, and waiting, (6) manual mode, and (7) fail-safing in progress, and shutdown complete. A standard personal computer provides command, data, and video displays for backup/additional control and status monitoring.

In addition to successfully integrating the hardware and software, as verified by collecting a number of test point-data sets, two retrieval scenarios were successfully demonstrated [14] on the PABF. The first scenario was for the EVA Retriever to search for and retrieve a tool and return to the home base. Tasks of this scenario included: activation, search for the tool, rendezvous with the tool, reach for the tool, grapple the tool, search for home base, and rendezvous with the home base. In the second scenario, the EVA Retriever initially was directed to search for an astronaut, then was redirected to search for and retrieve the tool and return to home base.

## 6. PHASE II SOFTWARE DESIGN

A number of systems engineering studies were conducted in support of the Phase II software design. Level A requirements for a projected Space Station version were developed in a conceptual design study [15]. Level B software requirements were derived in greater detail for this possible future Space Station application [16]. Space Station scenarios [17] were also described to aid in definition of dynamic situations needing reactive planning, and which will also be useful in defining a set of design reference missions. Phase II Level B software requirements were developed [18] as were Phase II PABF scenarios.

Various functions representing reasoning are introduced for the first time in Phase II software. Planning and replanning is based on: (1) simple reasoning about multiple conflicting goals whose priority is context dependent, (2) sensor-based knowledge of the environment, and (3) constraints such as flight rules or resource availability. Path planning to targets while avoiding obstacles is based on visual perception updating of the dynamic world model and reasoning about potential degraded capability. Grapple/grasp planning is based on visual perception updating for coordinated MMU, arm, and hand motions. Reasoning about data quality is provided. A mission control and assessment module provides decision making capability.

The choice of software architecture was based on experience with the Brooks layered subsumption architecture [19], reported experience with strict hierarchical and blackboard approaches, reported success of the DARPA ALV approach, and a desire to be compatible with the NASREM architecture [20].

The EVA Retriever software architecture incorporates a hierarchical decomposition of the control system that is horizontally partitioned into six major functional subsystems: sensing, perception, world model, reasoning, acting and performance measuring (See fig. 2). The design presented here utilizes hierarchical flow of command and status messages but allows horizontal flow of data between components at the same level. This approach handles multiple levels of abstraction well and permits the incorporation of special data paths between time critial components.

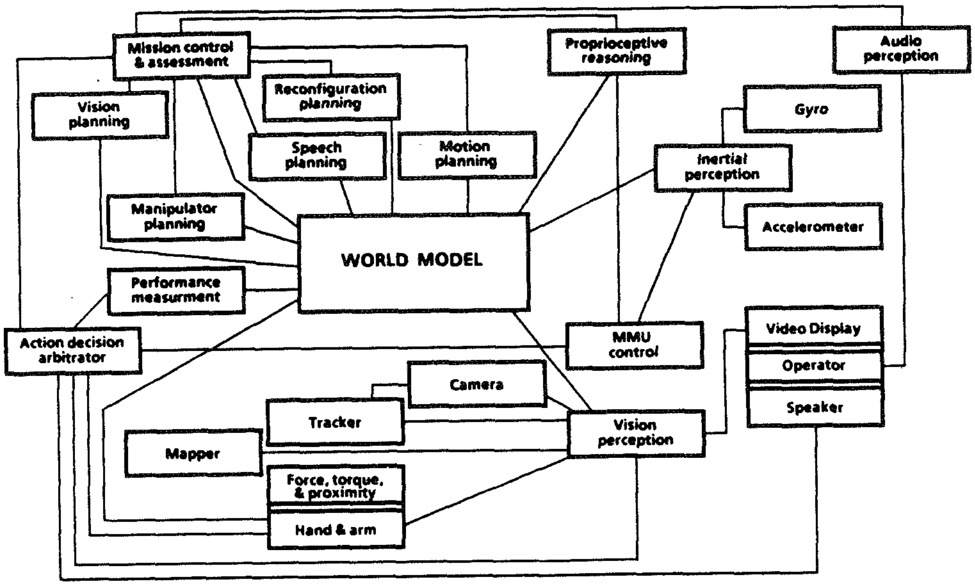


Figure 2. EVA Retriever Phase 2 software components.

Improvements in Phase II software in the sensing functions are: processing of range images to provide usable sketches, multisensor fusion to provide a sensor controlled robot, and expanded vocabulary recognition for supervisory override of most operations. The selected multisensor imaging approach to vision sensing and visual perception is based on the assessments that video intensity images and range images are basically complementary in information content, that they give enhanced segmentation of an image over either source alone, and that the combination is much more robust.

Perception software improvements in Phase II are: visual perception for multiple obstacle avoidance and grasp of different targets, information on arbitrary location and orientation of targets, self-awareness (proprioceptive perception) from fault detection and diagnosis of a portion of the MMU, and improved robot location from gyro, accelerometer, and vision.

The world model provides a representation of the external environment and internal status that is three-dimensional in space and dynamic in time. Acting as an intelligent central data base the world model has an episodic memory, data evaluation and estimation capabilities, and (limited) model based predictions of the EVA Retriever internal and external state. It contains a variety of general world knowledge as well as specific mission related facts such as space station proximity operations traffic control rules (simulated), mission rules/constraints, and object specific grapple rules/constraints.

The reasoning functions of the system are partitioned among the mission control and assessment, the action arbitrator, and the planning modules.

The mission control and assessment module develops and directs the execution of a mission plan. Commands are received from the operator through a voice recognition processor with comfirmation via voice synthesis. The module acts primarily as a plan execution monitor and as a meta planner delegating the creation and execution of detailed plans. Misson plan generation is based on dynamic knowledge of mission goals, constraints, and status as expressed in the world model. A set of cue action modules are utilized to initiate a wide range of monitoring, planning, and control actions. An internal assessment module triggers replanning whenever an expected cue fails to appear within an reasonable period of time.

The action arbitrator is the primary interface with the action subsystems. Under the supervision of the mission control and assessment module plan fragments are retrieved from the world model and transmitted to the appropriate action subsystem interface. Depending on the subsystem actions may be occurring in both a serial and parallel manner.

The planning module communicates with the mission control and assessment component and the world model. In general, the planning module responds to requests from the mission control and assessment component, issues data requests to the world model, and sends plans and status information to the world model. The total set of action primitives available to the planning module is based on the action requests recognized by the hand/arm, MMU, and the camera turntable subsystems.

The planning module consists of five functional planning components: vision, speech, motion, manipulator, and reconfiguration. For the current technology demonstration, the speech planning component was not implemented. The motion planner calculates a grid-based transition cost field based on safety zone and world model estimates of target/obstacle location and size. The transition cost field is dynamically updated in response to changes in obstacle location or number. The path is constructed of straight line segments with node location determined by a change in direction or the need maintain the target/body orientation required by the vision system. The vision planner constructs motion plans which support vision processing such as a search for an obscured target or positioning of the retriever for an analysis of a target grapple or degrapple. The module maintains internal models of vision sensors field of view given obstacles and plan move to see most probable part of a target region which has not been previously observed. The manipulator planner handles gross hand/arm positioning for grappling based on target type. A variety of closed-loop vision based control algorithms are being tested for fine grasping of small tools including an interesting reactive planner utilizing a potential field analysis. The reconfiguration planning module deals only with the MMU in the current prototype. This module uses both procedural fault diagnosis and isolation routines that have been incorporated into the MMU and a high level knowledge-based system to select reconfiguration strategies.

Research for other purposes in several related areas is being coordinated to possibly contribute to EVA Retriever Phase III. Notable here are: an effort on an autonomous agent with some emphasis in natural language understanding, general world knowledge, and autobiographical (episodic) memory for events experienced [21]; two related efforts in automated reactive planning [22, 23] with supervisory aspects; and an effort in machine qualitative reasoning [24].

## 7. CONCLUSIONS

A real need for retrieval of crew and objects in space near their prime spacecraft has been identified. The evaluation of the practical realization of a potential solution has been initiated in the form of a voice-supervised, intelligently autonomous robot. Successful demonstration of the first phase has been completed. Preliminary integration of the second phase software is largely complete at the time of writing this paper. Significant advances in intelligent software are planned to be evaluated in Phases II and III. Assessment of practicality will rest on experimental evidence when these are completed.

## 8. ACKNOWLEDGMENTS

This paper is based largely on an earlier version [25]. The EVA Retriever is a joint project of the Crew and Thermal Systems Division (CTSD), the Tracking and Communication Division (TCD), the Systems Development and Simulation Division SDSD), the Avionics Systems Division (ASD), and the Structures and Mechanisms Division (SMD), all of the Engineering Directorate at JSC, under funding from the JSC Director's Discretionary Fund. The contributions of the project teams encompassing civil servants and contractors to what is reported here are gratefully acknowledged.

## 9. REFERENCES

1. A. Cohen and J. Erickson, "Future Uses of Machine Intelligence and Robotics for the Space Station and Implications for the U.S. Economy," IEEE Journal of Robotics and Automation, Vol. RA-1, No. 3, September 1985.
2. J. Erickson, "Intelligent Systems and Robotics for an Evolutionary Space Station," Journal of the British Interplanetary Soc., Vol. 40, No. 10, October 1987, pp. 471-481.
3. N. Nilsson, "A Mobile Automaton: An Application of Artificial Intelligence Techniques," Proc. Int. Joint Conf. Art. Intell., Washington, D.C., May 1969, pp. 509-520.
4. M. Smith et al, "The system design of JASON, A Computer Controlled Mobile Robot," Proc. IEEE Conf. Cybernetics and Society, San Francisco, CA, Sept. 1975, pp. 72-75.
5. S. Yerazunis, D. Frederick, and J. Krajewski, "Guidance and Control of an Autonomous Rover for Planetary Exploration," Proc. IEEE Milwaukee Symp. Auto. Computation and Control, Milwaukee, WI, April 1976, pp. 7-16.
6. J. Miller, "A Discrete Adaptive Guidance System for a Roving Vehicle, " Proc. IEEE Conf. Decision and Control, New Orleans, LA, Dec. 1977, pp. 566-575.
7. H. Moravec, "The Stanford Cart and the CMU Rover," Proc. IEEE, Vol. 71, July 1983, pp. 872-884.
8. G. Giralt, R. Chatila, and M. Vausset, "An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots," Proc. 1st Int. Conf. Robotics Res., Bretton Woods, NH, Aug. 1983.
9. J. Nitao and A. Parodi, "A Real-time Reflexive Pilot for an Autonomous Land Vehicle," IEEE Control Sys. Mag., Vol 6, Feb. 1986, pp.14-23.
10. J. Lowrie et al, "Autonomous Land Vehicle," Annual Report, MCR-85-627, ETL-0413, Martin Marietta Aerospace, Denver, CO, Dec. 1985.
11. S. Harmon, "The Ground Surveillance Robot: An Autonomous Vehicle Designed to Transit Unknown Terrain," IEEE J. of Robotics and Auto., Vol. RA-3, No. 3, June 1987, pp. 266-279.
12. S. Harmon, "Autonomous Vehicles," in Encycl. of Art. Intell., New York, Wiley, 1986.
13. EVA Retriever Program Plan, Johnson Space Center, Houston, TX, JSC-22144, May 1987.
14. EVA Retriever Phase I Final, Johnson Space Center, Houston, TX, JSC-23175, July 1988.
15. J. Freeman, "Autonomous Robot Control System Design Study," Ford Aerospace Corp., Houston, TX, JSC-12421, May 1988.
16. D. Taylor et al, "Level B Post-PMC Space Station EVAR Functional Requirements, (Draft)," Project Report, Houston, TX, March 1988.
17. D. Shapiro, "An EVAR Scenario," Advanced Decision System, Mountain View, CA, June 1988.
18. D. Taylor et al, "EVA Retriever Phase II Derived Software Requirements (Draft)," Project Report, Houston, TX, June 1988.
19. R. A. Brooks, "A Robot Layered Control System for a Mobile Robot," IEEE J. of Robotics and Auto., Vol. RA-2, No. 1, March 1986, pp. 14-23.
20. J. S. Albus, R. Lumia, and H. McCain, "Software Architecture for Manufacturing and Space Robotics," 2nd AIAA/NASA/USAF Symp. on Auto., Robotics and Adv. Computing for the Nat. Space Prog., Arlington, VA, March 1987.
21. S. Vere, T. Bickmore, and R. Hanson, "An Autonomous Agent," Lockheed AI Center Private Communication, June 1988.
22. D. Shapiro, "Architecture for Semi-Autonomous Planning in Unrestricted Environments," Advanced Decision Systems, Mountain View, CA, July 1988.
23. M. Schoppers, "Adjustable Autonomy for Hazardous Robotic Operations," Advanced Decision Systems, Mountain View, CA, July 1988.
24. B. J. Kuipers and Y. T. Byun, "A Qualitative Approach to Robot Exploration and Map Learning," Proc. IEEE Spatial Reasoning and Multisensor Fusion, Los Altos, CA, Morgan Kaufman, 1987, pp. 390-404.
25. G. J. Reuter, C. W. Hess, D. E. Rhoades, L. W. McFadin, K. J. Healey, and J. D. Erickson, "An Intelligent, Free-flying Robot," SPIE Symposium on Advances in Intelligent Robotic Systems, Space Station Automation IV, Cambridge, MA, Nov. 6-11, 1988.

APPENDIX A

PROGRAM SCHEDULE

# NASA CONFERENCE ON SPACE TELEROBOTICS

FINAL PROGRAM
(UPDATED TITLES FOR PROCEEDINGS)

January 31 – February 2, 1989
The Pasadena Center
Pasadena, California

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Ames Research Center
Goddard Space Flight Center
Jet Propulsion Laboratory
Johnson Space Center
Kennedy Space Center
Langley Research Center
Marshall Space Flight Center

Conference Co-chairs: A.K. Bejczy, H. Seraji, Jet Propulsion Laboratory

## International Program Committee

J. Amat, University of Barcelona, Spain
G.A. Bekey, University of Southern California
P.R. Belanger, McGill University, Canada
R.C. Bolles, Stanford Research Center
J.G. Bollinger, University of Wisconsin
W.J. Book, Georgia Institute of Technology
J.M. Brady, Oxford University, UK
F.E.C. Culick, California Institute of Technology
R.J. deFigueiredo, Rice University
W.R. Ferrell, University of Arizona
E. Freund, University of Dortmund, FRG
A.A. Goldenberg, University of Toronto, Canada
R. Jain, University of Michigan
T. Kanade, Carnegie Mellon University
I. Kato, Waseda University, Japan
A.J. Koivo, Purdue University
P.D. Lawrence, University of British Columbia
J.Y.S. Luh, Clemson University
H.E. Rauch, Lockheed Palo Alto Research Lab
A. Rovetta, Polytechnic University of Milan
G.N. Saridis, Rensselaer Polytechnic Institute
T.B. Sheridan, MIT
L. Stark, University of California, Berkeley
D. Tesar, University of Texas at Austin
H. Van Brussel, Catholic University of Leuven
R. Volz, Texas A&M University

The theme of the conference is man-machine cooperation in space. The conference provides a forum to exchange ideas on the research and development required for application of telerobotics technology to the space systems planned for the 1990s and beyond. The conference is designed to (i) provide a view of current NASA telerobotic research and development; (ii) stimulate technical exchange on man-machine systems, manipulator control, machine sensing, artificial intelligence and system architecture; and (iii) identify important unsolved problems of current interest which can be dealt with by future research.

For information regarding the conference, contact G. Rodriguez, M/S 198-330, Jet Propulsion Laboratory, 4800 Oak Grove Drive, Pasadena, CA 91109; telephone (818) 354-4057.

## Telerobotics Working Group

The conference is sponsored by the Telerobotics Working Group of the NASA Office of Aeronautics and Space Technology. M. Montemerlo of NASA Headquarters and S.Z. Szirmay of the Jet Propulsion Laboratory co-chair this working group. Representatives from NASA centers and other research organizations are

D. Akin, Massachusetts Institute of Technology
J. Bull, Ames Research Center
R. Davis, Kennedy Space Center
S. Fisher, Ames Research Center
J. Haussler, Marshall Space Flight Center
A. Meintel, Langley Research Center
J. Pennington, Langley Research Center
D. Provost, Goddard Space Flight Center
C. Price, Johnson Space Center
L. Purves, Goddard Space Flight Center
C. Ruoff, Jr., Jet Propulsion Laboratory
E.C. Smith, Marshall Space Flight Center
J. Stocky, Jet Propulsion Laboratory
M. Zweben, Ames Research Center

# PROGRAM AT A GLANCE

## TUESDAY, JANUARY 31

| 7:00-8:30 | 8:30-9:45 | 10:00-1:00 | | | 2-2:30 | 2:45-6:00 |
|---|---|---|---|---|---|---|
| REGISTRATION | OPENING SESSION C124 | TECHNICAL SESSIONS | C310 | TA1: Redundant Manipulators 1 | PLENARY SESSION 1: Ames RC C124 | TP1: Redundant Manipulators 2 |
| | | | C312 | TA2: Man-Machine Systems | | TP2: Teleoperation 1 |
| | | | C314/5 | TA3: Telerobot Architectures | | TP3: Telerobots 1 |
| | | | C324 | TA4: Robot Sensing & Planning | | TP4: Telerobot Perception |
| | | | C112 | TA5: Navigation | | TP5: Rovers |
| | | | C326 | TA6: Neural Networks | | TP6: Parallel Processing |
| | | | C301/2 | TA7: Fundamental AI Research | | TP7: Spatial Representations & Reasoning |
| | | | C124 | TA8: Reasoning Under Uncertainty | | TP8: Ames Research Center |
| | | PANELS | C305 | TA9: Planning & Reas. in Sensor-Based Robotics | | TP9: Machine Learning |
| | | | C307 | | | TP10: Human-like Design for Robotics: Part I |
| | ▨ | | | | ▨ | ▨ |

## WEDNESDAY, FEBRUARY 1

| 8:30-9:30 | 9:45-1:00 | 2-2:30 | 2:45-6:00 |
|---|---|---|---|
| PLENARY SESSION 2: JPL and GSFC C124 | WA1: Flexible Arms | PLENARY SESSION 3: LaRC C124 | WP1: Manipulator Control 1 |
| | WA2: Robotic End-Effectors | | WP2: Telemanipulation |
| | WA3: Tele-Autonomous Systems | | WP3: Flight Experiments: Systems & Sim. |
| | WA4: Robotic Vision | | WP4: Sensor-Based Planning |
| | WA5: Telerobots 2 | | WP5: Special Topics |
| | WA6: Multi-Arm Control | | WP6: Robot Kinem., Dynamics & Control |
| | WA7: Coupling of Symbolic & Numeric Systems | | WP7: Robot Task Planning & Assembly |
| | WA8: Goddard Space Flight Center | | WP8: Langley Research Center |
| | WA9: Reasoning with Geometry | | WP9: Dexterity & AI |
| | WA10: Graphics in Teleoperation | | WP10: Human-like Design for Robotics: Part II |
| ▨ | ▨ | ▨ | ▨ |

## THURSDAY, FEBRUARY 2

| 8:30-9:00 | 9:15-12:30 | 2:00-4:00 | 4:00-6:00 |
|---|---|---|---|
| PLENARY SESSION 4: JSC C124 | THA1: Robot Arm Modeling & Control | PANEL DISCUSSION C124 | CLOSING RECEPTION C316 |
| | THA2: Special Topics in Teleoperation | | |
| | THA3: Telerobotic Space Operations | | |
| | THA4: Manipulator Control 2 | | |
| | THA5: Flight Experiment Concepts | | |
| | THA6: Manipulator Coordination | | |
| | THA7: Issues in AI Systems | | |
| | THA8: Johnson Space Center | | |
| ▨ | ▨ | | |

KEY
▨ Coffee Break C316

# *Plenary Sessions*

**OPENING SESSION**
Chair: Dr. G. Varsi, Jet Propulsion Laboratory
Tuesday, January 31, 8:30-9:45 a.m.

**Opening Remarks**
Dr. G. Varsi, Jet Propulsion Laboratory

**Conference Welcome**
Dr. T.E. Everhart, Caltech

**Evolving Space Teleoperation to Space Telerobotics: Research and Systems Considerations**
Dr. M. Montemerlo, NASA Headquarters

**Space Telerobotics Conference Objectives**
Dr. A.K. Bejczy, Jet Propulsion Laboratory

**PLENARY SESSION 1: NASA Ames Research Center**
Tuesday, January 31, 2:00-2:30 p.m.
**"Artificial Intelligence Research at NASA Ames"**
M. Zweben, NASA Ames Research Center

**PLENARY SESSION 2:**
**Jet Propulsion Laboratory**
Wednesday, February 1, 8:30-9:00 a.m.
**"JPL Space Robotics Program"**
C. Ruoff, Jr., Jet Propulsion Laboratory

**NASA Goddard Space Flight Center**
Wednesday, February 1, 9:00-9:30 a.m.
**"The Flight Telerobotic Servicer: NASA's First Operational Space Robot"**
C. Fuechsel, NASA Goddard Space Flight Center

**PLENARY SESSION 3: NASA Langley Research Center**
Wednesday, February 1, 2:00-2:30 p.m.
**"Telerobotics Research at Langley Research Center"**
J. Pennington, NASA Langley Research Center

**PLENARY SESSION 4: NASA Johnson Space Center**
Thursday, February 2, 8:30-9:00 a.m.
**"Telerobotic Activities at Johnson Space Center"**
C. Price, NASA Johnson Space Center

**PANEL DISCUSSION: Future Research Directions**
Moderator: J. Stocky, Jet Propulsion Laboratory
Thursday, February 2, 2:00-4:00 p.m.

The intent of this session is to provide views on the state of telerobotics research and to identify the near-term barriers to the application of telerobotics to be addressed in future advanced technology activities. The panel members give opening statements, which are followed by a general discussion with the audience.

# Tuesday Morning, January 31
# 10:00 a.m.-1:00 p.m.

**TA1 - REDUNDANT MANIPULATORS 1**
Chair: H. McCain, NASA Goddard Space Flight Center
**C310**

10:00 **"A 17 Degree of Freedom Anthropomorphic Manipulator"**
H. Vold, J. Karlen, J. Thompson, J. Farrell, P. Eismann, Robotics Research Corporation

10:30 **"A New Approach to Global Control of Redundant Manipulators"**
H. Seraji, Jet Propulsion Laboratory

11:00 **"Kinematic Functions for the 7 DOF Robotics Research Arm"**
K. Kreutz, M. Long, H. Seraji, Jet Propulsion Laboratory

11:30 **"Cartesian Control of Redundant Robots"**
R. Colbaugh, K. Glass, New Mexico State University

12:00 **"Kinematics, Controls, and Path Planning Results for a Redundant Manipulator"**
B. Gretz, S. Tilley, Ford Aerospace Corporation

12:30 **"A Complete Analytical Solution for the Inverse Instantaneous Kinematics of a Spherical-Revolute-Spherical (7R) Redundant Manipulator"**
R. Podhorodeski, R. Fenton, A. Goldenberg, University of Toronto

**TA2 - MAN-MACHINE SYSTEMS**
Chair: T. Sheridan, MIT
**C312**

10:00 **"Adjustable Impedance, Force Feedback, and Command Language Aids for Telerobotics"**
T. Sheridan, G. Raju, F. Buzan, W. Yared, J. Park, MIT

10:30 **"Variable Force and Visual Feedback Effects on Teleoperator Man/Machine Performance"**
M. Massimino, T. Sheridan, MIT

11:00 **"Teleoperator Comfort and Psychometric Stability: Criteria for Limiting Master-Controller Forces of Operation and Feedback During Telemanipulation"**
S. Wiker, E. Hershkowitz, J. Zik, Wisconsin Center for Space Automation and Robotics

11:30 **"Measurement of Hand Dynamics in a Microsurgery Environment: Preliminary Data in the Design of a Bimanual Telemicro-Operation Test Bed"**
S. Charles, R. Williams, Center for Engineering Applications

12:00 **"Human Factors Model Concerning the Man-Machine Interface of Mining Crewstations"**
J. Rider, R. Unger, U.S. Bureau of Mines

12:30 **"Development of a Flexible Test-Bed for Robotics, Telemanipulation and Servicing Research"**
B. Davies, British Aerospace, UK

**TA3 - TELEROBOT ARCHITECTURES**
Chair: E. Freund, University of Dortmund
**C314/5**

10:00 **"Control of Intelligent Robots in Space"**
E. Freund, C. Buhler, University of Dortmund

10:30 **"Modularity in Robotic Systems"***
D. Tesar, M. Butler, University of Texas at Austin

11:00 **"A System Architecture for a Planetary Rover"**
D. Smith, J. Matijevic, Jet Propulsion Laboratory

11:30 **"The NASA/OAST Telerobot Testbed Architecture"**
J. Matijevic, W. Zimmerman, S. Dolinsky, Jet Propulsion Laboratory

12:00 **"Formulation of Design Guidelines for Automated Robotic Assembly in Outerspace"**
S. Dwivedi, West Virginia University, G. Jones, GSFC/NASA, S. Banerjee, S. Srivastava, Bowie State University

12:30 **"Automation and Robotics Technology for Intelligent Mining Systems"**
J. Welsh, U.S. Bureau of Mines

**TA4 - ROBOT SENSING AND PLANNING**
Chair: A. Koivo, Purdue University
**C324**

10:00 **"A Fast Lightstripe Rangefinding System with Smart VLSI Sensor"**
A. Gruss, L. Carley, T. Kanade, Carnegie Mellon University

10:30 **"Methods and Strategies of Object Localization"**
L. Shao, University of Michigan, R. Volz, Texas A&M University

11:00 **"A Laser Tracking Dynamic Robot Metrology Instrument"**
G. Parker, J. Mayer, University of Surrey, UK

11:30 **"Robot Acting on Moving Bodies (RAMBO): Interaction with Tumbling Objects"**
L. Davis, D. DeMenthon, T. Bestul, S. Ziavras, H. Srinivasan, M. Siddalingaiah, D. Harwood, University of Maryland

12:00 **"Real-Time Edge Tracking Using a Tactile Sensor"**
A. Berger, R. Volpe, P. Khosla, Carnegie Mellon University

12:30 **"Planning 3-D Collision-Free Paths Using Spheres"**
S. Bonner, R. Kelley, Rensselaer Polytechnic Institute

*No paper received for conference proceedings

**TA5 - NAVIGATION**
Chair: B. Wilcox, Jet Propulsion Laboratory
**C112**

10:00 **"Map Learning with Indistinguishable Locations"**
K. Basye, T. Dean, Brown University

10:30 **"Three-dimensional Motor Schema Based Navigation"**
R. Arkin, Georgia Institute of Technology

11:00 **"Periodic Gaits for the CMU Ambler"**
S. Mahalingam, Carnegie Mellon University,
S. Dwivedi, West Virginia University

11:30 **"Exploiting Map Plans as Resources for Action"**
D. Payton, Hughes Artificial Intelligence Center

12:00 **"Learned Navigation in Unknown Terrains: A Retraction Method"**
N. Rao, Old Dominion University, N. Stoltzfus,
S. Iyengar, Louisiana State University

**TA6 - NEURAL NETWORKS**
Chair: J. Barhen, Jet Propulsion Laboratory
**C326**

10:00 **" 'Computational' Neural Learning Formalisms for Manipulator Inverse Kinematics"**
S. Gulati, J. Barhen, Jet Propulsion Laboratory,
S. Iyengar, Louisiana State University

10:30 **"Multi-Layer Neural Networks for Robot Control"**
F. Pourboghrat, Southern Illinois University

11:00 **"A Hybrid Architecture for the Implementation of the Athena Neural Net Model"**
C. Koutsougeras, Tulane University,
C. Papachristou, Case Western Reserve University

11:30 **"A Design Philosophy for Multi-Layer Neural Networks With Applications to Robot Control"**
N. Vadiee, M. Jamshidi, University of New Mexico

12:00 **"A Neural Network for Controlling the Configuration of Frame Structure With Elastic Members"**
K. Tsutsumi, Kobe University

12:30 **"Real Time Neural Network Based Learning Control of a Telerobotic System with Visual Feedback"***
W.T. Miller, University of New Hampshire

**TA7 - FUNDAMENTAL AI RESEARCH**
Chair: A. Tate, University of Edinburgh, UK
**C301/2**

10:00 **"Coordinating the Activities of a Planner and an Execution Agent"**
A. Tate, University of Edinburgh, UK

10:30 **"Plan Recognition for Space Telerobotics"**
B. Goodman, BBN Systems and Technologies Corp., D. Litman, AT&T Bell Labs

11:00 **"Causal Simulation and Sensor Planning in Predictive Monitoring"**
R. Doyle, Jet Propulsion Laboratory

11:30 **"State-Based Scheduling: An Architecture for Telescope Observation Scheduling"**
N. Muscettola, S. Smith, Carnegie Mellon University

12:00 **"Focus of Attention in an Activity-Based Scheduler"**
N. Sadeh, M. Fox, Carnegie Mellon University

12:30 **"Cognitive Values and Causal Ordering"***
R. Bhaskar, A. Nigam, IBM T.J. Watson Research Center

**TA8 - REASONING UNDER UNCERTAINTY**
Chair: H. Stephanou, George Mason University
**C124**

10:00 **"A Boltzmann Machine for the Organization of Intelligent Machines"**
M. Moed, G. Saridis, Rensselaer Polytechnic Institute

10:30 **"Accumulation of Uncertain Evidence in Spatial Reasoning"***
A. Kak, Purdue University

11:00 **"Grasp Planning Under Uncertainty"**
A. Erkmen, H. Stephanou, George Mason University

11:30 **"Approximation Algorithms for Planning and Control"**
M. Boddy, T. Dean, Brown University

12:00 **"Multiresolutional Models of Uncertainty Generation and Reduction"**
A. Meystel, Drexel University

*No paper received for conference proceedings

# *Tuesday Afternoon, January 31*
# *2:45-6:00 p.m.*

**TP1 - REDUNDANT MANIPULATORS 2**
Chair: Y. Nakamura, University of California, Santa Barbara
**C310**

2:45 **"Redundancy Management Issues of Intelligent Machines"***
Y. Nakamura, University of California, Santa Barbara

3:15 **"Characterization and Control of Self-motions in Redundant Manipulators"**
J. Burdick, California Institute of Technology, H. Seraji, Jet Propulsion Laboratory

3:45 **"ARMS: The Audrey Redundant Manipulator Simulator for Interactive Exploration of Design and Control"***
D. Mittman, Jet Propulsion Laboratory

4:30 **"Multiple Cooperating Manipulators: The Case of Kinematically Redundant Arms"**
I. Walker, R. Freeman, S. Marcus, University of Texas at Austin

5:00 **"Reflexive Obstacle Avoidance for Kinematically-Redundant Manipulators"**
J. Karlen, J. Thompson, Jr., J. Farrell, H. Vold, Robotics Research Corporation

5:30 **"Preliminary Study of a Serial-Parallel Redundant Manipulator"**
V. Hayward, R. Kurtz, McGill University

**TP2 - TELEOPERATION 1**
Chair: S. Fisher, NASA Ames Research Center
**C312**

2:45 **"The JPL Telerobot Operator Control Station: Part I - Hardware"**
E. Kan, Jet Propulsion Laboratory, J. Tower, G. Hunka, G. VanSant, GE Aerospace

3:15 **"The JPL Telerobot Operator Control Station: Part II - Software"**
E. Kan, Jet Propulsion Laboratory, B. Landell, S. Oxenberg, C. Morimoto, GE Aerospace

3:45 **"Design of a Monitor and Simulation Terminal Master for Space Station Telerobotics and Telescience"**
L. Lopez, C. Konkel, Teledyne Brown Engineering, P. Harmon, System Dynamics, Inc., S. King, Teledyne Brown Engineering

4:30 **"Performance Evaluation of a 6 Axis High Fidelity Generalized Force Reflecting Teleoperator"**
B. Hannaford, L. Wood, Jet Propulsion Laboratory

5:00 **"Implementation and Design of a Teleoperation System Based on a VMEbus/68020 Pipelined Architecture"**
T. Lee, Jet Propulsion Laboratory

5:30 **"Human/Machine Interaction via the Transfer of Power and Information Signals"**
H. Kazerooni, W. Foslien, B. Anderson, T. Hessburg, University of Minnesota

**TP3 - TELEROBOTS 1**
Chair: R. Lumia, National Institute of Standards and Technology
**C314/5**

2:45 **"Trajectory Generation for Space Telerobots"**
R. Lumia, A. Wavering, National Institute of Standards and Technology

3:15 **"On the Simulation of Space Based Manipulators with Contact"**
M. Walker, J. Dionise, University of Michigan

3:45 **"Preliminary Results on Noncollocated Torque Control of Space Robot Actuators"**
S. Tilley, C. Francis, K. Emerick, M. Hollars, Ford Aerospace

4:30 **"Portable Dextrous Force Feedback Master for Robot Telemanipulation (P.D.M.F.F.)"**
G. Burdea, Rutgers University, T. Speeter, AT&T Bell Labs

5:00 **"Experiences with the JPL Telerobot Testbed 1 - Issues and Insights"**
H. Stone, B. Balaram, J. Beahan, Jet Propulsion Laboratory

5:30 **"The KALI Multi-Arm Robot Programming and Control Environment"**
P. Backes, S. Hayati, Jet Propulsion Laboratory, V. Hayward, McGill University, K. Tso, Jet Propulsion Laboratory

**TP4 - TELEROBOT PERCEPTION**
Chair: R. deFigueiredo, Rice University
**C324**

2:45 **"Image Enhancement of Convex Polyhedral Objects by Optimal Illumination"***
A. Gateau, R. deFigueiredo, Rice University

3:15 **"How Do Robots Take Two Parts Apart?"**
R. Bajcsy, C. Tsikos, University of Pennsylvania

3:45 **"Techniques and Potential Capabilities of Multi-Resolutional Information (Knowledge) Processing"**
A. Meystel, Drexel University

4:30 **"Perceptual Telerobotics"**
P. Ligomenides, University of Maryland

5:00 **"Reasoning about Perception for Robotic Control"***
M. Schoppers, R. Shu, Advanced Decision Systems

5:30 **"Building an Environment Model Using Depth Information"**
Y. Roth-Tabak, R. Jain, University of Michigan

*No paper received for conference proceedings

**TP5 - ROVERS**
Chair: C. Weisbin, Oak Ridge National Laboratory
**C112**

2:45 **"HERMIES-III: A Step Toward Autonomous Mobility, Manipulation and Perception"**
C. Weisbin, B. Burks, J. Einstein, R. Feezell, W. Manges, D. Thompson, Oak Ridge National Laboratory

3:15 **"First Results in Terrain Mapping for a Roving Planetary Explorer"**
E. Krotkov, C. Caillas, M. Hebert, I. Kweon, T. Kanade, Carnegie Mellon University

3:45 **"Planetary Rover Technology Development Requirements"**
R. Bedard, Jr., B. Muirhead, Jet Propulsion Laboratory, M. Montemerlo, M. Hirschbein, NASA

4:30 **"Rice-obot I: An Intelligent, Autonomous, Mobile Robot"**
R. deFigueiredo, L. Ciscon, D. Berberian, Rice University

5:00 **"Satellite-Map Position Estimation for the Mars Rover"**
A. Hayashi, University of Texas at Austin, T. Dean, Brown University

5:30 **"Robotic Sampling System for an Unmanned Mars Mission"**
W. Chun, Martin Marietta Space Systems

**TP6 - PARALLEL PROCESSING**
Chair: C.S.G. Lee, Purdue University
**C326**

2:45 **"Efficient Mapping Algorithms for Scheduling Robot Inverse Dynamics Computation on a Multiprocessor System"**
C.S.G. Lee, C. Chen, Purdue University

3:15 **"Robot Control Computation in Microprocessor Systems with Multiple Arithmetic Processors"***
B. Li, S. Ahmad, Purdue University

3:45 **"Parallel Algorithms for Computation of the Manipulator Inertia Matrix"**
M. Amin-Javaheri, D.E. Orin, Ohio State University

4:30 **"Neural Computation for Real-Time Assembly Scheduler"***
P. Chang, C.S.G. Lee, Purdue University

5:00 **"Parallel Complexity: Which Algorithm to Choose or Develop for Parallelization? A Case Study of Computation of the Manipulator Inertia Matrix"***
A. Fijany, Jet Propulsion Laboratory

5:30 **"Parallel Algorithms and Architectures for Computation of the Manipulator Inverse Dynamics"***
A. Fijany, A. Bejczy, Jet Propulsion Laboratory

**TP7 - SPATIAL REPRESENTATIONS AND REASONING**
Chair: C. Laugier, LIFIA-IMAG, France
**C301/2**

2:45 **"Planning Robot Actions Under Position and Shape Uncertainty"***
C. Laugier, LIFIA-IMAG, France

3:15 **"Planning Robot Actions Under Position and Shape Uncertainty (Continued)"**
C. Laugier, LIFIA-IMAG, France

3:45 **"Innovative Design Systems: Where Are We, and Where Do We Go From Here?"***
D. Navinchandra, Carnegie Mellon University

4:30 **"Organising Geometric Computations for Space Telerobotics"**
S. Cameron, University of Oxford, UK

5:00 **"GARE: Geometric Analysis and Reasoning Engine"***
R. Desai, R. Doshi, R. Lam, J. White, Jet Propulsion Laboratory

5:30 **"A Tesselated Probabilistic Representation for Spatial Robot Perception and Navigation"**
A. Elfes, Carnegie Mellon University

**TP8 - NASA AMES RESEARCH CENTER**
Chair: M. Zweben, NASA Ames Research Center
**C124**

2:45 **"A Survey of Planning and Scheduling Research at the NASA Ames Research Center"**
M. Zweben, NASA Ames Research Center

3:15 **"Situated Control Rules"***
M. Drummond, NASA Ames Research Center/Sterling Software

3:45 **"Integrating Planning and Reactive Control"**
S. Rosenschein, L. Kaelbling, Teleos Research

4:30 **"Learning in Stochastic Neural Networks for Constraint Satisfaction Problems"**
M. Johnston, Space Telescope Science Institute, H-M. Adorf, Space Telescope 1 - European Coordinating Facility

5:00 **"Integrating Planning, Execution and Learning"**
D. Kuokka, Carnegie Mellon University

5:30 **"An Integrated Architecture for Intelligent Agents"***
K. Thompson, P. Langley, University of California, Irvine

*No paper received for conference proceedings

# Wednesday Morning, February 1
# 9:45 a.m.-1:00 p.m.

## WA1 - FLEXIBLE ARMS
Chair: W. Book, Georgia Institute of Technology
**C310**

9:45 **"Modeling, Design, and Control of Flexible Manipulator Arms: Status and Trends"**
W. Book, Georgia Institute of Technology

10:15 **"Dynamical Modeling of Serial Manipulators with Flexible Links and Joints Using the Method of Kinematic Influence"**
P. Graves, Lockheed Engineering & Sciences Co.

10:45 **"Capture of Free-Flying Payloads With Flexible Space Manipulators"**
T. Komatsu, M. Uenohara, S. Iikura, Toshiba Corporation, H. Miura, I. Shimoyama, University of Tokyo

11:30 **"Technology and Task Parameters Relating to the Effectiveness of the Bracing Strategy"**
W. Book, J. Wang, Georgia Institute of Technology

12:00 **"Manipulators with Flexible Links: A Simple Model and Experiments"**
I. Shimoyama, University of Tokyo,
I. Oppenheim, Carnegie Mellon University

12:30 **"Experiments in Identification and Control of Flexible-Link Manipulators"**
S. Yurkovich, A. Tzes, F. Pacheco, Ohio State University

## WA2 - ROBOTIC END-EFFECTORS AND HAND CONTROLLERS
Chair: G. Bekey, University of Southern California
**C312**

9:45 **"Autonomous Dexterous End-Effectors for Space Robotics"**
G. Bekey, T. Iberall, H. Liu, University of Southern California

10:15 **"Design and Control of a Multi-Fingered Robot Hand Provided With Tactile Feedback"**
H. Van Brussel, B. Santoso, D. Reynaerts, Catholic University of Leuven, Belgium

10:45 **"Traction-Drive Force Transmission for Telerobotic Joints"**
D. Kuban, D. Williams, Oak Ridge National Laboratory

11:30 **"Force/Torque and Tactile Sensors for Sensor-Based Manipulator Control"**
H. Van Brussel, H. Belien, C.-Y. Bao, Catholic University of Leuven

12:00 **"Redundant Sensorized Arm + Hand System for Space Telerobotized Manipulation"**
A. Rovetta, P. Cavestro, Polytechnic University of Milan, Italy

12:30 **"Impedance Hand Controllers for Increasing Efficiency in Teleoperations"**
C. Carignan, J. Tarrant, STX Systems Corporation

## WA3 - TELE-AUTONOMOUS SYSTEMS
Chair: R. Volz, Texas A&M University
**C314/5**

9:45 **"Tele-Autonomous Systems: New Methods for Projecting and Coordinating Intelligent Action at a Distance"**
L. Conway, University of Michigan, R. Volz, Texas A&M University, M. Walker, University of Michigan

10:15 **"An Advanced Telerobotic System for Shuttle Payload Changeout Room Processing Applications"**
M. Sklar, D. Wegerif, McDonnell Douglas Space Systems Co., L. Davis, NASA Kennedy Space Center

10:45 **"Robotic Tele-Existence"**
S. Tachi, H. Arai, T. Maeda, Ministry of International Trade and Industry, Japan

11:30 **"Redundancy of Space Manipulator on Free-Flying Vehicle and Its Nonholonomic Path Planning"**
Y. Nakamura, R. Mukherjee, University of California, Santa Barbara

12:00 **"Guidance Algorithms for a Free-Flying Space Robot"**
A. Brindle, H. Viggh, J. Albert, Boeing Aerospace

12:30 **"Telepresence System Development for Application to the Control of Remote Robotic Systems"**
C. Crane III, J. Duffy, R. Vora, S.-C. Chiang, University of Florida

## WA4 - ROBOTIC VISION
Chair: L. Stark, University of California, Berkeley
**C324**

9:45 **"3D Model Control of Image Processing"**
A. Nguyen, L. Stark, University of California, Berkeley

10:15 **"Weighted Feature Selection Criteria for Visual Servoing of a Telerobot"**
J. Feddema, C.S.G. Lee, O. Mitchell, Purdue University

10:45 **"Model-Based Computer Vision Applied to Structured Objects"***
W. Wolfe, University of Colorado, M. Magee, University of Wyoming, D. Mathis, C. Sklair, Martin Marietta Astronautics Group

11:30 **"Trinocular Stereovision using Figural Continuity, Dealing with Curved Objects"**
R. Vaillant, O. Faugeras, INRIA, France

12:00 **"A Fast 3-D Object Recognition Algorithm for the Vision System of a Special-Purpose Dexterous Manipulator"**
S. Hung, National Research Council of Canada

12:30 **"Use of 3D Vision for Fine Robot Motion"**
A. Lokshin, T. Litwin, Jet Propulsion Laboratory

*No paper received for conference proceedings

**WA5 - TELEROBOTS 2**
Chair: K. Corker, BBN Systems and Technologies Corp.
**C112**

9:45 **"Telerobotic Workstation Design Aid"**
K. Corker, E. Hudlicka, D. Young, N. Cramer, BBN Systems and Technologies Corp.

10:15 **"Space Robotic System for Proximity Operations"**
P. Magnani, M. Colomba, Tecnospazio S.p.A., Italy

10:45 **"Modeling and Sensory Feedback Control for Space Manipulators"**
Y. Masutani, F. Miyazaki, S. Arimoto, Osaka University

11:30 **"Control Strategies for a Telerobot"**
J. O'Hara, Brookhaven National Laboratory, B. Stasi, Grumman Space Systems

12:00 **"Autonomous Sensor-Based Dual-Arm Satellite Grappling"**
B. Wilcox, K. Tso, T. Litwin, S. Hayati, B. Bon, Jet Propulsion Laboratory

12:30 **"Thread: A Programming Environment for Interactive Planning-level Robotics Applications"**
J. Beahan, Jr., Jet Propulsion Laboratory

**WA6 - MULTI-ARM CONTROL**
Chair: J. Luh, Clemson University
**C326**

9:45 **"Compliance of Dual-Robot Systems for Internal and External Forces"***
J. Luh, J. Tao, Clemson University

10:15 **"Stability Analysis of Multiple-Robot Control Systems"**
J. Wen, Rensselaer Polytechnic Institute, K. Kreutz, Jet Propulsion Laboratory

10:45 **"Experiments in Cooperative Manipulation: A System Perspective"**
S. Schneider, R. Cannon, Jr., Stanford University

11:30 **"On the Manipulability of Dual Cooperative Robots"**
P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano, University of Naples, Italy

12:00 **"Controlling Multiple Manipulators Using RIPS"**
Y. Wang, S. Jordan, A. Mangaser, S. Butner, University of California, Santa Barbara

12:30 **"Time Optimal Movement of Cooperating Robots"**
J. McCarthy, J. Bobrow, University of California, Irvine

**WA7 - COUPLING OF SYMBOLIC AND NUMERIC SYSTEMS**
Chair: M. Fox, Carnegie Mellon University
**C301/2**

9:45 **"Reflections on the Relationship Between Artificial Intelligence and Operations Research"**
M. Fox, Carnegie Mellon University

10:15 **"What Kind of Computation Is Intelligence? A Framework for Integrating Different Kinds of Expertise"**
B. Chandrasekaran, Ohio State University

10:45 **"A Design Strategy for Autonomous Systems"**
P. Forster, University of Edinburgh, UK

11:30 **"Learning in Tele-autonomous Systems using Soar"**
J. Laird, E. Yager, C. Tuck, M. Hucka, University of Michigan

12:00 **"Robust Robot Execution and Task Combination"***
W. Troxell, Colorado State University

12:30 **"Design of a Structural and Functional Hierarchy for Planning and Control of Telerobotic Systems"**
L. Acar, University of Missouri-Rolla, U. Ozguner, Ohio State University

**WA8 - NASA GODDARD SPACE FLIGHT CENTER**
Chair: H. McCain, NASA Goddard Space Flight Center
**C124**

9:45 **"The Flight Telerobotic Servicer Project: A Technical Overview"**
H. McCain, NASA Goddard Space Flight Center

10:15 **"The Flight Telerobotic Servicer Tinman Concept: System Design Drivers and Task Analysis"**
J. Andary, D. Hewitt, S. Hinkal, Goddard Space Flight Center

10:45 **"The Flight Telerobotic Servicer: From Functional Architecture to Computer Architecture"**
R. Lumia, J. Fiala, National Institute of Standards and Technology

11:30 **"Research and Development Activities at the Goddard Space Flight Center for the Flight Telerobotic Servicer Project"**
S. Ollendorf, Goddard Space Flight Center

12:00 **"The Goddard Space Flight Center (GSFC) Robotics Technology Testbed"**
R. Schnurr, M. O'Brien, Goddard Space Flight Center, S. Cofer, Digital Equipment Corp.

12:30 **"Test and Validation for Robot Arm Control Dynamics Simulation"**
H. Yae, S.-S. Kim, E. Haug, University of Iowa, W. Seering, K. Sundaram, B. Thompson, MIT, J. Turner, H. Chun, Cambridge Research, H. Frisch, R. Schnurr, Goddard Space Flight Center

*No paper received for conference proceedings

# *Wednesday Afternoon, February 1*
# *2:45-6:00 p.m.*

**WP1 - MANIPULATOR CONTROL 1**
Chair: T. Hsia, University of California, Davis
**C310**

2:45 **"Cartesian Control Schemes for Robot Manipulators"***
T. Hsia, University of California, Davis

3:15 **"An Improved Adaptive Control for Repetitive Motion of Robots"**
F. Pourboghrat, Southern Illinois University

3:45 **"Direct Adaptive Control of a PUMA 560 Industrial Robot"**
H. Seraji, T. Lee, Jet Propulsion Laboratory, M. Delpech, C.N.E.S., France

4:30 **"Model Based Manipulator Control"**
L. Petrosky, Westinghouse Advanced Energy Systems, I. Oppenheim, Carnegie Mellon University

5:00 **"Discrete-Time Adaptive Control of Robot Manipulators"**
M. Tarokh, University of California, San Diego

5:30 **"A Discrete Decentralized Variable Structure Robotic Controller"**
Z. Tumeh, Georgia Institute of Technology

**WP2 - TELEMANIPULATION**
Chair: D. Tesar, University of Texas at Austin
**C312**

2:45 **"Construction and Demonstration of a 9-String 6 DOF Force Reflecting Joystick for Telerobotics"**
R. Lindemann, Jet Propulsion Laboratory, D. Tesar, University of Texas at Austin

3:15 **"Response to Reflected-Force Feedback to Fingers in Teleoperations"**
P. Sutter, J. Iatridis, N. Thakor, Franklin and Marshall College

3:45 **"The Jau-JPL Anthropomorphic Telerobot"**
B. Jau, Jet Propulsion Laboratory

4:30 **"A Procedure Concept for Local Reflex Control of Grasping"**
P. Fiorini, J. Chang, Jet Propulsion Laboratory

5:00 **"Performance Limitations of Bilateral Force Reflection Imposed by Operator Dynamic Characteristics"**
J. Chapel, Martin Marietta Space Systems

5:30 **"Sensor-based Fine Telemanipulation for Space Robotics"**
M. Andrenucci, M. Bergamasco, P. Dario, University of Pisa, Italy

**WP3 - FLIGHT EXPERIMENTS: SYSTEMS AND SIMULATORS**
Chair: A. Bejczy, Jet Propulsion Laboratory
**C314/5**

2:45 **"ROTEX-TRIIFEX: Proposal for a Joint FRG-USA Telerobotic Flight Experiment"**
G. Hirzinger, German Aerospace Research Establishment, A. Bejczy, Jet Propulsion Laboratory

3:15 **"Test and Training Simulator for Ground-Based Teleoperated In-Orbit Servicing"**
B. Schafer, German Aerospace Research Establishment

3:45 **"Concept Synthesis of an Equipment Manipulation and Transportation System (EMATS)"**
W. De Peuter, ESTEC, E. Waffenschmidt, Dornier-System GmbH, West Germany

4:30 **"Force-Reflective Teleoperated System With Shared and Compliant Control Capabilities"**
Z. Szakaly, W.S. Kim, A. Bejczy, Jet Propulsion Laboratory

5:00 **"Information management in an Integrated Space Telerobot"**
S. Di Pippo, ASI-Italian National Space Agency, G. Pasquariello, IESI-CNR, Italy, G. Labini, ASI-Italian Space Agency

5:30 **"Redundancy in Sensors, Control and Planning of a Robotic System for Space Telerobotics"**
A. Rovetta, S. Vodret, M. Bianchini, Polytechnic University of Milan, Italy

**WP4 - SENSOR-BASED PLANNING**
Chair: M. Mason, Carnegie Mellon University
**C324**

2:45 **"How to Push a Block Along a Wall"**
M. Mason, Carnegie Mellon University

3:15 **"Global Models: Robot Sensing, Control, and Sensory-Motor Skills"**
P. Schenker, Jet Propulsion Laboratory

3:45 **"3-D Vision System Integrated Dexterous Hand"**
R. Luo, Y.-S. Han, North Carolina State University

4:30 **"A Layered Abduction Model of Perception: Integrating Bottom-up and Top-down Processing in a Multi-Sense Agent"**
J. Josephson, Ohio State University

5:00 **"RCTS: A Flexible Environment for Sensor Integration and Control of Robot Systems - The Distributed Processing Approach"**
R. Allard, B. Mack, M. Bayoumi, Queen's University at Kingston

5:30 **"Vehicle Path-Planning in Three Dimensions Using Optics Analogs for Optimizing Visibility and Energy Cost"**
N. Rowe, D. Lewis, U.S. Naval Postgraduate School

*No paper received for conference proceedings

**WP5 - SPECIAL TOPICS**
Chair: S. Hackwood, University of California at Santa Barbara
**C112**

2:45 **"Vacuum Mechatronics"**
S. Hackwood, S. Belinski, G. Beni, University of California, Santa Barbara

3:15 **"Uniform Task Level Definitions for Robotic System Performance Comparisons"**
C. Price, NASA Johnson Space Center, D. Tesar, University of Texas at Austin

3:45 **"Linear Analysis of a Force Reflective Teleoperator"**
K. Biggers, S. Jacobsen, C. Davis, University of Utah

4:30 **"Real-Time Cartesian Force Feedback Control of a Teleoperated Robot"**
P. Campbell, Lockheed Engineering & Sciences Company

5:00 **"Optimal Payload Rate Limit Algorithm for Zero-G Manipulators"**
M. Ross, D. McDermott, Lockheed Engineering & Sciences Co.

5:30 **"Assembly of Objects With Not Fully Predefined Shapes"**
M. Arlotti, V. Di Martino, IBM Rome Scientific Center

**WP6 - ROBOT KINEMATICS, DYNAMICS AND CONTROL**
Chair: F.E.C. Culick, California Institute of Technology
**C326**

2:45 **"Recursive Multibody Dynamics and Discrete-Time Optimal Control"**
G. D'Eleuterio, C. Damaren, University of Toronto

3:15 **"The Effects of Gear Reduction on Robot Dynamics"**
J. Chen, University of Maryland

3:45 **"Recursive Newton-Euler Formulation of Manipulator Dynamics"**
M. Nasser, Lockheed Engineering & Sciences Company

4:30 **"Kinematic Sensitivity of Robot Manipulators"**
M. Vuskovic, San Diego State University

5:00 **"Efficient Conjugate Gradient Algorithms for Computation of the Manipulator Forward Dynamics"**
A. Fijany, R. Scheid, Jet Propulsion Laboratory

5:30 **"On the Stability of Robotic Systems with Random Communication Rates"**
H. Kobayashi, X. Yun, R. Paul, University of Pennsylvania

**WP7 - ROBOT TASK PLANNING AND ASSEMBLY**
Chair: A. Sanderson, Rensselaer Polytechnic Institute
**C301/2**

2:45 **"Precedence Relationship Representations of Mechanical Assembly Sequences"**
L. Homem de Mello, Carnegie Mellon University, A. Sanderson, Rensselaer Polytechnic Institute

3:15 **"Using Multiple Sensors for Printed Circuit Board Insertion"**
D. Sood, M. Repko, R. Kelley, Rensselaer Polytechnic Institute

3:45 **"An Overview of the Intelligent Machining Workstation"***
D. Bourne, Carnegie Mellon University

4:30 **"Design and Manufacturing Intent"***
D. Bourne, Carnegie Mellon University

5:00 **"Determining Robot Actions For Tasks Requiring Sensor Interaction"**
J. Budenske, Honeywell Systems and Research Center, M. Gini, University of Minnesota

**WP8 - NASA LANGLEY RESEARCH CENTER**
Chair: J. Pennington, NASA Langley Research Center
**C124**

2:45 **"The Laboratory Telerobotic Manipulator Program"**
J. Herndon, S. Babcock, P. Butler, H. Costello, R. Glassell, R. Kress, D. Kuban, J. Rowe, D. Williams, Oak Ridge National Laboratory

3:15 **"Robotic Control of the Seven-Degree-of-Freedom NASA Laboratory Telerobotic Manipulator"**
R. Dubey, J. Euler, R. Magness, University of Tennessee, S. Babcock, J. Herndon, Oak Ridge National Laboratory

3:45 **"The Control of Space Manipulators Subject to Spacecraft Attitude Control Saturation Limits"**
S. Dubowsky, MIT, E. Vance, Center for Naval Analyses, M. Torres, MIT

4:30 **"System Architectures for Telerobotic Research"**
F. Harrison, Langley Research Center

5:00 **"Comparison of Joint Space Versus Task Force Load Distribution Optimization for a Multiarm Manipulator System"**
D. Soloway, Langley Research Center, T. Alberts, Old Dominion University

*No paper received for conference proceedings

# *Thursday Morning, February 2*
# *9:15 a.m.-12:30 p.m.*

## THA1 - ROBOT ARM MODELING AND CONTROL
Chair: G. Saridis, Rensselaer Polytechnic Institute
**C310**

9:15 **"Dynamic Characteristics of Macro/Mini-Manipulators: Application to Space Robot Systems"***
O. Khatib, Stanford University

9:45 **"Application of Recursive Manipulator Dynamics to Hybrid Software/Hardware Simulation"**
C. Hill, Lockheed Engineering & Sciences Co., K. Hopping, Boeing Electronics Co., C. Price, NASA Johnson Space Center

10:15 **"Kinematics & Control Algorithm Development and Simulation for a Redundant Two-Arm Robotic Manipulator System"**
M. Hennessey, P. Huang, C. Bunnell, FMC Corporation

11:00 **"Inverse Dynamics of a 3 Degree of Freedom Spatial Flexible Manipulator"**
E. Bayo, M. Serna, University of California, Santa Barbara

11:30 **"A Control Approach for Robots With Flexible Links and Rigid End-Effectors"**
E. Barbieri, Tulane University, U. Ozguner, Ohio State University

## THA2 - SPECIAL TOPICS IN TELEOPERATION
Chair: W. Seering, MIT
**C312**

9:15 **"Preshaping Command Inputs to Reduce Telerobotic System Oscillations"**
N. Singer, W. Seering, MIT

9:45 **"Performance Constraints and Compensation For Teleoperation With Delay"**
J. McLaughlin, B. Staunton, The Aerospace Corporation

10:15 **"Flight Telerobotic Servicer Control From the Orbiter"**
T. Ward, D. Harlan, Lockheed Engineering & Sciences Company

11:00 **"Teleoperation Experiments with a Utah/MIT Hand and a VPL DataGlove"**
D. Clark, J. Demmel, J. Hong, G. Lafferriere, L. Salkind, X. Tan, New York University

11:30 **"Instruction Dialogues: Teaching New Skills to a Robot"**
C. Crangle, P. Suppes, Stanford University

12:00 **"Interset: A Natural Language Interface for Teleoperated Robotic Assembly of the EASE Space Structure"**
D. Boorsma, MIT

## THA3 - TELEROBOTIC SPACE OPERATIONS
Chair: D. Akin, MIT
**C314/5**

9:15 **"Space Operations Testing of Telerobots in Neural Buoyancy"***
D. Akin, MIT

9:45 **"Establishing Viable Task Domains for Telerobot Demonstrations"**
W. Zimmerman, Jet Propulsion Laboratory

10:15 **"The Telerobot Workstation Testbed for the Shuttle Aft Flight Deck: A Project Plan for Integrating Human Factors into System Design"**
T. Sauerwein, NASA Goddard Space Flight Center

11:00 **"Multi-Level Manual and Autonomous Control Superposition for Intelligent Telerobot"**
S. Hirai and T. Sato, Ministry of International Trade & Industry, Japan

11:30 **"An Alternative Control Structure for Telerobotics"**
P. Boissiere, R. Harrigan, Sandia National Laboratories

12:00 **"Integration of a Sensor Based Multiple Robot Environment for Space Applications: The Johnson Space Center Teleoperator Branch Robotics Laboratory"**
J. Hwang, P. Campbell, M. Ross, Lockheed Engineering & Sciences Company, C. Price, D. Barron, NASA Johnson Space Center

## THA4 - MANIPULATOR CONTROL 2
Chair: T. Tarn, Washington University
**C324**

9:15 **"Unified Approach to the Control of Non-redundant/ Redundant Rigid/Flexible Robot Arms"***
T. Tarn, Washington University, A. Bejczy, Jet Propulsion Laboratory

9:45 **"Requirements for Implementing Real-Time Control Functional Modules on a Hierarchical Parallel Pipelined System"**
T. Wheatley, J. Michaloski, R. Lumia, National Institute of Standards and Technology

10:15 **"The JPL Telerobot Manipulator Control and Mechanization Subsystem (MCM)"**
S. Hayati, T. Lee, K. Tso, P. Backes, E. Kan, Jet Propulsion Laboratory, J. Lloyd, McGill University

11:00 **"On Discrete Control of Nonlinear Systems With Applications to Robotics"**
M. Eslami, University of Illinois at Chicago

11:30 **"Robust Control of an Industrial Manipulator"***
M. Cohen, L. Daneshmend, McGill University

12:00 **"A Spatial Operator Algebra for Manipulator Modeling and Control"**
G. Rodriguez, K. Kreutz, A. Jain, Jet Propulsion Laboratory

*No paper received for conference proceedings

**THA5 - FLIGHT EXPERIMENT CONCEPTS**
Chair: L. Jenkins, NASA Johnson Space Center
**C112**

9:15 **"Flight Experiments in Telerobotics - Orbiter Middeck Concept"**
L. Jenkins, NASA Johnson Space Center

9:45 **"Robotic Systems: An Important Asset to the SDS"***
D. Nussman, S. Greene, Dynamics Research Corporation

10:15 **"Experimental Study on Two-Dimensional Free-Flying Robot Satellite Model"**
Y. Umetani, K. Yoshida, Tokyo Institute of Technology

11:00 **"The Astronaut and the Banana Peel: an EVA Retriever Scenario"**
D. Shapiro, Advanced Decision Systems

11:30 **"Computed Torque Control of a Free-Flying Cooperating-Arm Robot"**
R. Koningstein, M. Ullman, R. Cannon, Jr., Stanford University

12:00 **"Next Generation Space Robot"**
T. Iwata, M. Oda, R. Imai, National Space Development Agency of Japan

**THA6 - MANIPULATOR COORDINATION**
Chair: A. Meystel, Drexel University
**C326**

9:15 **"Coordination in a Hierarchical Multi-Actuator Controller"**
A. Meystel, Drexel University

9:45 **"Distributed Communications and Control Network for Robotic Mining"**
W. Schiffbauer, U.S. Bureau of Mines, Pittsburgh Research Center

10:15 **"Computer Simulation and Design of a Three Degree-of-Freedom Shoulder Module"**
D. Marco, U.S. Naval Postgraduate School,
L. Torfason, University of New Brunswick,
D. Tesar, University of Texas at Austin

11:00 **"A Collision Avoidance System for a Spaceplane Manipulator Arm"**
A. Sciomachen, P. Magnani, Tecnospazio S.p.A., Italy

**THA7 - ISSUES IN AI SYSTEMS**
Chair: N. Sridharan, FMC Corporation
**C301/2**

9:15 **"Real-Time Performance for Interactive AI Systems"***
N. Sridharan, FMC Corporation

9:45 **"Generic Task Problem Solvers in Soar"**
T. Johnson, J. Smith, Jr., B. Chandrasekaran, Ohio State University

10:15 **"Temporal Logics Meet Telerobotics"**
E. Rutten, L. Marce, IRISA/INRIA, France

11:00 **"An Efficient Temporal Logic for Robotic Task Planning"**
J. Becker, Martin Marietta Astronautics Group

11:30 **"The Indexed Time Table Approach for Planning and Acting"**
M. Ghallab, A. Alaoui, LAAS-CNRS, France

12:00 **"Reactive Behavior, Learning, and Anticipation"**
S. Whitehead, D. Ballard, University of Rochester

**THA8 - NASA JOHNSON SPACE CENTER**
Chair: C. Price, NASA Johnson Space Center
**C124**

9:15 **"Shuttle Remote Manipulator System Mission Preparation and Operations"**
E. Smith, Jr., NASA Johnson Space Center

9:45 **"A Comparison of the Shuttle Remote Manipulator System and the Space Station Freedom Mobile Servicing Center"**
E. Taylor, NASA Johnson Space Center,
M. Ross, Lockheed Engineering & Sciences Co.

10:15 **"Dexterous Manipulator Flight Demonstration"**
E. Carter, Lockheed Engineering & Sciences Co.

11:00 **"An Intelligent Free-Flying Robot"**
G. Reuter, C. Hess, D. Rhoades, L. McFadin, K. Healey, J. Erickson, NASA Johnson Space Center, D. Phinney, Lockheed Engineering & Sciences Company

11:30 **"Smart Hands for the EVA Retriever"***
C. Hess, NASA Johnson Space Center

12:00 **"Machine Vision"***
R. Juday, NASA Johnson Space Center

*No paper received for conference proceedings

# *Panels on Artificial Intelligence*

### TA9 - PLANNING AND REASONING IN SENSOR-BASED ROBOTICS

10:00 a.m.-1:00 p.m.
Moderator: A. Kak, Purdue University

Panel: S.-S. Chen, University of North Carolina
T. Kanade, Carnegie Mellon University
B. Kuipers, University of Texas at Austin
T. Linden, Advanced Decision Systems
A. Tate, University of Edinburgh, UK

This panel discussion is designed to address issues in planning, navigation, frameworks for reasoning, spatial databases, geometrical reasoning, sensor fusion and spatial reasoning.

### TP9 - MACHINE LEARNING, KNOWLEDGE ACQUISITION AND SEMI-AUTONOMOUS AGENTS

2:45-6:00 p.m.
Moderator: A. Rappaport, Carnegie Mellon Univ./Neuron Data

Panel: B. Gaines, University of Calgary, Canada
J. Laird, University of Michigan
T. Mitchell, Carnegie Mellon University
G. Boy, CERT/ONERA, France

The purpose of this panel is to review the state of the art in the fields of machine learning and knowledge acquisition and to relate those critical advances of AI to telerobotics. Telerobotics involves capturing knowledge where the domain theory is often incomplete, performing tasks using this knowledge and automatically adapting it to new situations. AI architectures involving multiple reasoning techniques and learning capabilities are necessary to assist human and robot performance. Telerobotics is a preferred application area; those advanced AI techniques and their use in this domain should in turn provide important insights to AI researchers.

### WA9 - REASONING WITH GEOMETRY

9:45 a.m.-1:00 p.m.
Moderator: H.R. Keshavan, Northrop Research Center

Panel: F. Arbab, University of Southern California
R. Desai, Jet Propulsion Laboratory
R. Hoffman, Northrop Corp.
D. Hunter, Northrop Corp.
F. Prinz, Carnegie Mellon University
S. Smith, Northrop Corp.

The panel was formed to present techniques and problems in geometric modeling, geometric reasoning, representing and reasoning with uncertainty, topological reasoning, process planning and CAD directed vision.

### WP9 - INTERACTIONS BETWEEN DEXTERITY AND AI FOR A ROBOT

2:45-6:00 p.m.
Moderator: J. Latombe, Stanford University

Panel: B. Donald, Cornell University
M. Genesereth, Stanford University
A. Haddad, Lockheed
M. Mason, Carnegie Mellon University
J. Pertin-Troccaz, CNRS, France
K. Salisbury, MIT
P. Schenker, Jet Propulsion Laboratory

The purpose of this panel is to discuss the interdependence of robot dexterity and autonomy; that is, the interactions between the physical capabilities and decision capabilities of a robot. The issues it was formed to address include dexterity, redundant arms, dexterous hands, sensor-based motion primitives, whole robot manipulation, AI methodologies and autonomy. The panel also has another function: to present new, challenging ideas for modeling, computational, inferential, and combinatorial issues in spatial reasoning and AI.

# Panels on Humanlike Design

## TP10 - HUMANLIKE DESIGN FOR ROBOTICS: A HELPFUL METAPHOR OR RED HERRING? PART I: VISION

2:45-6:00 p.m.
Moderator: L. Stark, Univ. of California, Berkeley

2:45-2:55
**"Human Scanpaths"**
L. Stark, Univ. of California, Berkeley
Cognitive models control active looking in a top-down perceptual process.

2:55-3:15
**"Statistical Dependency in Visual Scanning"**
S. Ellis, NASA Ames Research Center
Statistical testing provides evidence for the Stark/Norton/Ellis "scanpath" hypothesis.

3:15-3:40
**"Top-Down Image Processing for Robotic Control"**
A. Nguyen, University of California, Berkeley
An image processing scheme has been developed, for telerobotic control, that rests extensively on a model of the telerobot working environment to control the robots and the cameras viewing the robotic scene. Parameter updating of the model is obtained in a rapid, robust fashion since image processing only occurs in these specialized regions of interest where positional feedback is essential.

3:40-4:10
**"Image Processing"**
R. Brodersen, Univ. of California, Berkeley
Rapid processing of video images is possible with VLSI chips to compute Hough transforms, centroids, etc., and with these chips embedded in specialized computer architectures. This bottom-up processing is based upon engineering filtering theory and is in the leading edge of image processing.

4:10-4:15
**"Vision for Space Telerobots"**
B. Wilcox, Jet Propulsion Laboratory
The visual environment for space telerobotics is very different from the natural, terrestrial, visual environment which evolved human vision (e.g., harsh lighting, deep shadows, highly specular surfaces, man-made structures, etc.). A vision system designed for this environment is described.

4:30-5:00
**Panel Discussion**

## WP10 - HUMANLIKE DESIGN FOR ROBOTICS: A HELPFUL METAPHOR OR RED HERRING? PART II: LOCOMOTION

2:45-6:00 p.m.
Moderator: L. Stark, University of California, Berkeley

2:45-3:10
**"Human Locomotion"**
V. Krishnan, San Francisco State University
The multiple muscles involved in generating joint torques in human locomotion provide an opportunity for optimization to constrain the redundancy. A set of experimental studies was used to evaluate candidate optimum criteria and the results suggested that energy minimization was important.

3:10-3:40
**"Hopping Locomotion"**
M. Raibert, MIT
A set of demonstration hopping robots with one to four legs has been studied to evaluate this interesting mode of locomotion for a variety of tasks. Some insights into biological locomotion have come from this biomimetic system.

3:40-4:10
**"Mobility Enhancement Using Active Coordination"**
K. Waldron, Ohio State University
The kinematic and energetic elegant engineering solution that wheels provide has made wheeled locomotion ubiquitous in our modern industrial society.

4:10-4:30 BREAK

4:30-4:40
**"Specialized Wheels"**
G. Paine, Jet Propulsion Laboratory
Large hooped wheels prove to be successful on the irregular terrain of the moon. This disproves the notion that wheels are a good invention only after the invention of the road.

4:40-5:10
**Panel Discussion**

# Panel on Graphic Overlays in Teleoperation

**WA10 - GRAPHICS AND GRAPHIC OVERLAYS IN TELEOPERATION**
9:45 a.m.-1:00 p.m.
Moderator: D.B. Diner, Jet Propulsion Laboratory

9:45-10:15
**"Graphic Overlays in High-Precision Teleoperation: Current and Future Work at JPL"**
D.B. Diner, S. Venema, Jet Propulsion Laboratory

10:15-10:45
**"Virtual Environment Workstation for Telepresence and Telerobotic Supervisory Control"***
S. Fisher, NASA Ames Research Center

10:45-11:15
**"Head-Mounted Spatial Instruments II: Synthetic Reality or Impossible Dream"**
S. Ellis, A. Grunwald, Technion, Israel

11:30-12:00
**"The Effects of Overlay Graphics on Telepresence"***
R. Pepper, Naval Ocean Systems Center

12:00-12:30
**"Use of Graphics in Decision Aids for Telerobotic Control"**
T. Sheridan, J. Roseborough, H. Das, K.-P. Chin, S. Inoue, MIT

12:30-1:00
**Panel Discussion**

## APPENDIX B
## INDEX BY AUTHOR

# APPENDIX C

## ATTENDEES/PARTICIPANTS*

M. Al Abidi
ECE Department
University of Tennessee
Ferris Hall
Knoxville, TN 37996-2100

Levent Acar
University of Missouri-Rolla
112 Electrical Engineering
Rolla, MO 65401

Barbara Ackerman
TRW, R10/1368B
One Space Park
Redondo Beach, CA 90278

Thomas E. Alberts
Dept. of Mech. Engr. & Mechanics
Old Dominion University
Norfolk, VA 25529-0247

Harold L. Alexander
Aeronautics & Astronautics Dept.
Massachusetts Inst. of Technology
125 Massachusetts Avenue, Rm. 33-119
Cambridge, MA 02139

Raymond Allard
Queen's University
Dept. of Electrical Engineering
Kingston, Ontario, Canada K7M 1B5

Phillip Alvelda
JPL MS 198-330

Masoud Amin-Javaheri
Ohio State University
Dept. of Electrical Engineering
2015 Neil Avenue, Room 205
Columbus, OH 43210

James F. Andary
NASA Goddard Space Flight Center
Code 409
Greenbelt, MD 20771

David E. Anderson
McDonnell Douglas Space Systems Co.
Space Station Division
5301 Bolsa, MC 11-3, J893
Huntington Beach, CA 92647

Victor Anselmo
Rand Corporation
1700 Main Street
Santa Monica, CA 90293

Takashi Aoki
Fujitsu Laboratory Ltd.
2 Brattle Drive, #12
Arlington, MA 02174

Colin Archibald
National Research Council/Canada
Division of Electrical Engineering
Bldg. M50, Montreal Road
Ottawa, Ontario, Canada K1A 0R8

Ronald C. Arkin
Georgia Institute of Technology
School of Info. & Computer Science
Rich Building/0280
Atlanta, GA 30332

Paul Backes
JPL MS 198-330

Ruzena K. Bajcsy
University of Pennsylvania
Computer & Infor. Science Dept.
200 S. 33rd Street
Philadelphia, PA 19104-6389

J. (Bob) Balaram
JPL MS 198-330

Mark Banyai
Dynamics Research Corporation
1755 Jefferson Davis Hwy.
Suite 802
Arlington, VA 22202

* The complete address for JPL personnel is Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109.

Enrique Barbieri
Tulane University
Electrical Engineering Dept.
204 Stanley Thomas Hall
New Orleans, LA 70118

Jacob Barhen
JPL MS 198-330

Eduardo Bayo
University of California
Dept. of Mech. & Environmental Engr.
Santa Barbara, CA 93106

Mohamed Bayoumi
Queen's University
Dept. of Electrical Engineering
Kingston, Ontario, Canada K7M SR9

John J. Beahan, Jr.
JPL MS 198-330

Jeffrey M. Becker
Martin Marietta Astronautics Grp.
P.O. Box 179
MS 4372
Denver, CO 80201

Roger J. Bedard, Jr.
JPL MS 79-23

Stefan Begej
Begej Corporation
5 Claret Ash Road
Littleton, CO 80127

A.K. Bejczy
JPL MS 198-330

George A. Bekey
University of Southern California
Computer Science Department
Los Angeles, CA 90089-0782

Massimo Bergamasco
Scuola Superiore S. Anna
Via Carducci, 40
Pisa, Italy 56100

Eric Biefeld
JPL MS 301-440

Klaus B. Biggers
Center for Engineering Design
University of Utah
3176 MEB
Salt Lake City, UT 84112

Peter T. Boissiere
Sandia National Laboratories
P.O. Box 5800, Division 1414
Albuquerque, NM 87185

Eva Bokor
JPL MS 198-330

Mark Bolas
NASA Ames/Stanford University
220 Curtner, #0
Palo Alto, CA 94306

Bruce Bon
JPL MS 23

Wayne J. Book
Georgia Institute of Technology
School of Mechanical Engineering
Atlanta, GA 30332-0405

Daniel K. Boorsma
Massachusetts Inst. of Technology
77 Massachusetts Avenue
MC 145-100
Cambridge, MA 02139

Jan F.T. Bos
Dept. Mech. Engr. & Machine Tech.
Delft University of Technology
Mehelweg 2
2628 CD Delft, The Netherlands

John J. Bosley
JPL MS 301-285

John Bouvier
Ocean Systems Engineering
770 Leesburg Dike, Suite 200
Falls Church, VA 22043

Guy A. Boy
ONERA/CERT
2 Avenue Edouard Belin
31055 Toulouse, France

Anne F. Brindle
Boeing Aerospace
P.O. Box 3999
MS 82-58
Seattle, WA 98124

Robert Broderson
University of California, Berkeley
EECS Department
Berkeley, CA 94720

John Budenske
Honeywell, Inc.
Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

Charles T. Bunnell
FMC Corporation
Advanced Systems Center
1200 South Second Street
Minneapolis, MN 55421

Grigore C. Burdea
Rutgers University
Dept. of Elec. & Comp. Engineering
P.O. Box 909
Piscataway, NJ 08855-0909

Joel W. Burdick
California Institute of Technology
MC 104-44
Pasadena, CA 91125

Harrison Burris
TRW, R10/1368F
One Space Park
Redondo Beach, CA 90278

Nestor Burtnyk
National Research Council/Canada
Division of Electrical Engineering
Bldg. M50, Montreal Road
Ottawa, Ontario, Canada K1A 0R8

Stephen Cameron
Robotics Research Group
University of Oxford
Dept. of Engineering Science
Parks Road
Oxford, U.K. OX1 3PJ

Perry D. Campbell
Lockheed Engr. & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

David J. Cannon
Stanford University
84C Escondido Village
Stanford, CA 94305

Robert H. Cannon, Jr.
Dept. of Aero/Astronautics
Stanford University
Room 250, Durand Bldg.
Stanford, CA 94305

Craig Carignan
ST Systems Corporation
157 Spring Hill Road
Suite 500
Vienna, VA 22180

Per-Helge Carlsen
Center for Industrial Research
Box 124 Blindern
0314 Oslo 3, Norway

Elisabeth Carpenter
JPL
Reston, VA

Gil Carpenter
Grumman Aerospace
Corporate Research Center
MS A08-35
Bethpage, NY 11714

Ed L. Carter
Lockheed Engr. & Sciences Co.
2400 NASA Road 1
P.O. Box 58561
Houston, TX 77258

Francois Cellier
University of Arizona
Elect. & Computer Engr. Dept.
Tucson, AZ 85721

B. Chandrasekaran
Ohio State University
Dept. of Computer & Info. Science
Laboratory for AI Research
Columbus, OH 43210

Jeffrey Chang
JPL MS 189-133

Jim D. Chapel
Martin Marietta Astronautics Group
Space Systems Company
P.O. Box 179, MS B1690
Denver, CO 80201

Steve Charles
Center for Engineering Appl.
6401 Poplar Avenue, Suite 190
Memphis, TN 38119

Chun-Lung Chen
Purdue University
Dept. of Electrical Engineering
West Lafayette, IN 47907

Jigien Chen
University of Maryland
Dept. of Mechanical Engineering
College Park, MD 20742

Vincent Chen
Stanford University
Aero/Astro Department
Room 250, Durand Building
Stanford, CA 94305

Pasquale Chiacchio
Dept. of Info. and Systems
University of Naples
Via Claudio 21
80125 Naples, Italy

Keith Chrystall
Alberta Research Council
6815 8th Street, N.E.
Calgary, Alberta, Canada T2E 7H7

Wendell Chun
Martin Marietta Astronautics Group
Space Systems Company
P.O. Box 179, Mail Stop S8082
Denver, CO 80201

Lawrence Ciscon
Rice University
Dept. of Elec. & Computer Engr.
Houston, TX 77251-1892

Michael Clark
Apple Computer Inc.
292 S. La Cienega Blvd.
Suite 301
Beverly Hills, CA 90211

Kevin Cleary
University of Texas
Mechanical Engineering Dept.
Austin, TX 78712

Steve Cohan
Odetics, Inc.
1515 S. Manchester Avenue
Anaheim, CA 92802

Richard D. Colbaugh
New Mexico State University
Mechanical Engineering Dept.
Box 3450
Las Cruces, NM 88001

Terry Cole
JPL MS 180-500

Kent D. Copeland
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
P.O. Box 58561
Houston, TX 77258

Kevin Corker
Bolt, Beranek & Newman Inc.
10 Moulton Street
Cambridge, MA 02138

Carl D. Crane III
University of Florida
Dept. of Mechanical Engineering
30 MEB
Gainesville, FL 32611

Colleen Crangle
Stanford University
IMSSS, Ventura Hall (4115)
Stanford, CA 94305

Manuel I. Cruz
TRW, R11/2385
One Space Park
Redondo Beach, CA 90278

Fred Culick
Caltech
201 Karman Lab 301-46
Pasadena, CA 91125

Glenn E. Cunningham
JPL MS 264-726

Chris J. Damaren
University of Toronto
Institute for Aerospace Studies
4925 Dufferin Street
Downsview, Ontario, Canada  M3H 5T6

Barry F. Davies
British Aerospace PLC
Sowerby Research Centre
Human Factors Department, FPC 267
P.O. Box 5, Filton
Bristol, U.K.  BS12 7QW

Larry S. Davis
University of Maryland
Inst. for Advanced Computer Studies
A.V. Williams Bldg., 115
College Park, MD  20742-3251

Virgil Davis
NASA Kennedy Space Center
MC DM-MED-12
Kennedy Space Center, FL  32899

Thomas Dean
Brown University
Dept. of Computer Science
Box 1910
Providence, RI  02912

Rui J. deFigueiredo
Rice University
Dept. of Elec. & Computer Engr.
Houston, TX  77251-1892

Willem De Peuter
European Space Agency/ESTEC
Mail Code WKR, Mail Box 299
Noordwijk, Holland  2200 AG

Ralph H. Dergance
Martin Marietta Space Systems Co.
Astronautics Group
P.O. Box 179, MC B1690
Denver, CO  80201

Rajiv Desai
JPL MS 301-440

Alan Desrochers
Rensselaer Polytechnic Institute
JEC 6012
Troy, NY  12180-3590

William Dias
JPL MS 301-250D

Bill Dickson
Stanford University
Aero/Astro Department
Room 250, Durand Building
Stanford, CA  94305

Peter Dieleman
National Aerospace Laboratory (NLR)
Space Div., Systems Department
P.O. Box 153
8300 AD Emmeloord, The Netherlands

Daniel Diner
JPL MS 278

Joseph Dionise
University of Michigan
Elec. Engr. & Comp. Sci. Dept.
1101 Beal Avenue
Ann Arbor, MI  48105

Shlomo Dolinsky
JPL MS 303-308

Raj Doshi
JPL MS 301-440

Richard Doyle
JPL MS 301-440

Michael Drews
JPL MS 303-308

Mark Drummond
NASA Ames Research Center
MS 1244-17
Moffett Field, CA  94035

R.V. Dubey
Oak Ridge National Laboratory
P.O. Box 2008
Oak Ridge, TN  37831-6005

Steven Dubowsky
Massachusetts Inst. of Technology
Dept. of Mechanical Engineering
469A
Cambridge, MA  02139

Stephen J. Ducsai
Martin Marietta Astronautics Group
Space Systems Company
P.O. Box 179, MC B1690
Denver, CO 80201

Neil Duffie
(address not available)

Joseph Duffy
University of Florida
Dept. of Mechanical Engineering
300 MEB (CIMAR)
Gainesville, FL 32611

Jayson Durham
U.S. Naval Ocean Systems Center
Code 943
San Diego, CA 92152-5000

Suren N. Dwivedi
West Virginia University
Dept. of Mech. & Aerospace Engr.
ESB 333
Morgantown, WV 26506

Paul Eismann
Robotics Research Corporation
5400 DuPont Circle, TechneCenter
Milford, OH 45150

Alberto Elfes
Carnegie Mellon University
Robotics Institute
Pittsburgh, PA 15213

Neal Ely
USAF, Space Division
SD/ALI LA AFB
P.O. Box 92960 WWPC
Los Angeles, CA 90009-2960

Alfred English
USAF, Space Division
HQ Space Division/ALIW
P.O. Box 92960
Los Angeles, CA 90009

Mansour Eslami
University of Illinois at Chicago
Dept. Elec. Engr. & Computer Science
M/C 154
Chicago, IL 60680-4348

Ted Evans
McDonnell Douglas Space Support
5301 Bolsa Avenue
Huntington Beach, CA 92647

T.E. Everhart
California Institute of Technology
MC 204-31
Pasadena, CA 91125

Arwed Exner
DLR German Aerospace Research Agcy.
Linder Hoehe
P.O. Box 906058
5000 Cologne, West Germany

James D. Farrell
Robotics Research Corporation
5400 DuPont Circle, TechneCenter
Milford, OH 45150

John T. Feddema
Purdue University
School of Electrical Engr.
Electrical Engineering Bldg.
West Lafayette, IN 47907

Derek Fender
(address not available)

Amir Fijany
JPL MS 198-330

Paolo Fiorini
JPL MS 303-308

Rodolfo Fiorini
Politecnico Energetica
Plaza L. Da Vinci 32
20133 Milan, Italy

Jim Firby
JPL MS 301-440

Scott Fisher
NASA Ames Research Center
MS 239-3
Moffett Field, CA 94035

Carl Flatau
Telerobotics, Inc.
45 Knickerbocker Avenue
Bohemia, NY 11716-3104

Peter Forster
University of Edinburgh
Dept. of Artificial Intelligence
5 Forrest Hill
Edinburgh, Scotland, U.K. EH1 2QL

Judy Franklin
GTE Laboratories Inc.
40 Sylvan Road
Waltham, MA 02254

Robert French
JPL MS 198-330

Eckhard Freund
University of Dortmund
Institute of Robotics Research
Otto Hahn Str. 8, Postf. 500500
4600 Dortmund 50, West Germany

Gary Friedman
JPL MS 301-270C

Harold P. Frisch
NASA Goddard Space Flight Center
Code 712.1
Greenbelt, MD 20771

Charles F. Fuechsel
NASA Goddard Space Flight Center
Code 409
Greenbelt, MD 20771

Erann Gat
JPL MS 301-440

Pierre Gawthier
Hydro-Quebec
Institute of Research
1800 Montee Ste-Julie
Varennes, Quebec, Canada J0L 2P0

Wesley Gerriots
Schilling Development Inc.
720 Olive Drive, Unit D
Davis, CA 95616

Malik Ghallab
LAAS-CNRS
7 Avenue Colonel Roche
31077 Toulouse, France

Moji Ghodussi
University of California/
Santa Barbara
6740 Cortona Drive
Goleta, CA 93117

Gerd Goelz
DLR German Aerospace Establishment
PT-PM
Linder Hoehe
D-5000 Cologne 90, West Germany

Andre Goforth
NASA Ames Research Center
Mail Stop 244-7
Moffett Field, CA 94035

Andrew A. Goldenberg
University of Toronto
Dept. of Mechanical Engineering
5 King's College Road
Toronto, Ontario, Canada M5S 1A4

Alexander Golub
TRW, Space & Technology
R11/1757
One Space Park
Redondo Beach, CA 90278

Brad Goodman
BBN Systems & Technologies Corp.
Mail Stop 009
10 Moulton Street
Cambridge, MA 02138

Philip L. Graves
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

Stephen A. Greene
Dynamics Research Corporation
1755 Jefferson Davis Highway
Suite 802
Arlington, VA 22202

Arnold Greenman
McDonnell Douglas Automatn./Rbtcs.
5301 Bolsa Avenue
Huntington Beach, CA 92647

Helen Greiner
Massachusetts Inst. of Technology
410 Memorial Drive
Room 133C
Cambridge, MA 02139

Bruce Gretz
Ford Aerospace
Space Systems Division
3825 Fabian Way, MS G-77
Palo Alto, CA 94303

Andy Gruss
Carnegie Mellon University
Dept. of Electrical & Computer Engr.
Schenley Park
Pittsburgh, PA 15213-3890

Steve Gunter
JPL MS 198-231

Ashish Guptá
Caltech
MC 116-81
Pasadena, CA 91125

Susan Hackwood
University of California
Cntr./Robotic Syst. in Microelec.
Santa Barbara, CA 93106

Karen N. Halterman
NASA Goddard Space Flight Center
Code 409
Greenbelt, MD 20771

Blake Hannaford
JPL MS 198-330

Joe Hanson
JPL MS 303-308

Robert Harkins
Martin Marietta Space Systems Co.
Astronautics Group
P.O. Box 179, MC B1690
Denver, CO 80201

Don L. Harlan
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

Raymond W. Harrigan
Sandia National Laboratories
P.O. Box 5800, Division 1414
Albuquerque, NM 87185

Fenton Harrison
NASA Langley Research Center
Mail Stop 152D
Hampton, VA 23665

Samad Hayati
JPL MS 198-330

Vincent Hayward
McGill University
Electrical Engineering Dept.
3480 University Street
Montreal, Quebec, Canada H3A 2A7

Anthony Healey
U.S. Naval Postgraduate School
Monterey, CA 93943

Michael P. Hennessey
FMC Corporation
4800 East River Road
Minneapolis, MN 55421

J.N. Herndon
Oak Ridge National Laboratory
Oak Ridge, TN 37831

Christopher J. Hill
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

Shigeoki Hirai
Electrotechnical Laboratory, MITI
Intelligent Systems Division
1-1-4 Umezono, Tsukuba-shi, Ibaraki
305, Japan

Masayuku Hiroguchi
Stanford University
Robotics Laboratory
Stanford, CA 94305

William Hoff
Martin Marietta Astronautics
P.O. Box 179, MS 4372
Denver, CO 80201

Richard Hoffman
Northrop Rsch. & Technology Center
One Research Park
Palos Verdes Peninsula, CA 90274

Michael G. Hollars
Ford Aerospace Corp.
Space Systems Division
3825 Fabian Way, G-77
Palo Alto, CA 94303

Paul Houpt
GE Research and Development
Rm. KWD215
2 River Road
Schenectady, NY 12301

Michael Hucka
University of Michigan
Artificial Intelligence Laboratory
Ann Arbor, MI 48109-2110

Stephen H.Y. Hung
National Research Council/Canada
Division of Electrical Engineering
Bldg. M50, Montreal Road
Ottawa, Ontario, Canada K1A OR8

Daniel B. Hunter
Northrop Rsch. & Technology Center
One Research Park
Palos Verdes Peninsula, CA 90274

David Hunter
National Research Council/Canada
Space Station Program
Montreal Road, Bldg. R-105
Ottawa, Ontario, Canada K1A OR6

John Hussey
Grumman Space Division
Bethpage, NY 11714

C. James Hwang
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

Ryoichi Imai
National Space Dev. Agency/Japan
Tsukuba Space Center
2-1-1, Sengen, Tsukuba City 305
Japan

Imdad Imam
GE Corporate Research & Devel.
1 River Road, K1-3A 20
Schenectady, NY 12301

Toshiaki Iwata
University of Wisconsin - Madison
2120 University Avenue, #211
Madison, WI 53705

Abhinandan Jain
JPL MS 198-330

Warren Jasper
Stanford University
Aero/Astro Department
Rm. 250, Durand Building
Stanford, CA 94305

Lyle M. Jenkins
NASA Johnson Space Center
Houston, TX 77058

Mark D. Johnston
Space Telescope Science Institute
3700 San Martin Drive
Baltimore, MD 21218

John R. Josephson
Ohio State University
Lab. for AI Research, 228 CAE Bldg.
2036 Neil Avenue
Columbus, OH 43210-1277

Richard Juday
NASA Johnson Space Center
MC EE6
Houston, TX 77058

Leslie Pack Kaelbling
Teleos Research
576 Middlefield Road
Palo Alto, CA 94301

Avi Kak
Purdue University
School of Electrical Engineering
West Lafayette, IN 47907

Edwin Kan
JPL MS 198-330

James P. Karlen
Robotics Research Corporation
5400 DuPont Circle
TechneCenter
Milford, OH 45150

H. Kazerooni
University of Minnesota
Mechanical Engineering Dept.
111 Church St., SE
Minneapolis, MN 55455

Robert B. Kelley
Rensselaer Polytechnic Institute
Elec., Comp., & Systems Engr. Dept.
CII-8015
Troy, NY 12180

Michael Kelly
Georgia Institute of Technology
GTRI/SEL
Atlanta, GA 30332

Lothar Kerstein
c/o ERNO Raumfahrttechnik GmbH.
Huenefeldstr. 1-5
2800 Bremen, West Germany

H.R. Keshavan
Northrop Rsch. and Technology Corp.
One Research Park
Palos Verdes Peninsula, CA 90274

Pradeep K. Khosla
Carnegie Mellon University
Dept. of Elec. & Computer Engr.
The Robotics Institute
Pittsburgh, PA 15213

Eric Kilgore
Schilling Development Inc.
720 Olive Drive, Unit D
Davis, CA 95616

Richard Killion
Rockwell Science Center
Knowledge Systems Department
1049 Camino Dos Rios, MS A18
Thousand Oaks, CA 91360

Won Soo Kim
JPL MS 278

Hiroaki Kobayashi
Meiji University
Dept. of Precise Engineering
1-1-1 Higashi-Mita Tama
Kawasaki, 214 Japan

Anthi Koivo
Purdue University
Dept. of Electrical Engineering
West Lafayette, IN 47907

Ross Koningstein
Stanford University
Aerospace Robotics Laboratory
Rm. 250, Durand Building
Stanford, CA 94305

Cris Koutsougeras
Tulane University
Computer Science Department
New Orleans, LA 70118

Hiroshi Koyama
Mitsubishi Electric Corp.
Kamakura Works
325, Kamimachiya Kamakura
Kanagawa 247, Japan

Kenneth Kreutz
JPL MS 198-330

V.V. Krishnan
San Francisco State University
Dept. of Electrical Engineering
San Francisco, CA 94132

Eric Krotkov
Carnegie Mellon University
The Robotics Institute
Pittsburgh, PA 15213-3890

Benjamin Kuipers
University of Texas
Artificial Intelligence Lab
Taylor Hall 2.124
Austin, TX 78712

Daniel R. Kuokka
Carnegie Mellon University
Computer Science Dept.
Pittsburgh, PA 15213

Giovanni Sylos Labini
ASI - Italian Space Agency
202, V.le Regina Margerita
00100 Rome, Italy

Gerardo Lafferriere
New York University
Robotics Research Laboratory
715 Broadway
New York, NY 10003

John E. Laird
University of Michigan
Artificial Intelligence Lab
1101 Beal Avenue
Ann Arbor, MI 48109-2110

Raymond Lam
JPL MS 301-440

Ken Lambert
JPL MS 185-105

Jean-Claude Latombe
Stanford University
Robotics Laboratory
Stanford, CA 94305

Christian Laugier
LIFIA Laboratory/IMAG
Nat'l Polytechnic Inst./Grenoble
46 Avenue Felix Viallet
38031 Grenoble Cedex, France

Otto Ledford
Advanced Technology Inc.
222 N. Sepulveda, Suite 1310
El Segundo, CA 90245

C.S. George Lee
Purdue University
School of Electrical Engineering
MSEE 256
West Lafayette, IN 47907

Greg Lee
ST Systems
4400 Forbes Blvd.
Lanham, MD 20706

Sukhan Lee
University of Southern California
Dept. EE Systems, PHE 228, MC 0273
University Park
Los Angeles, CA 91011

Thomas S. Lee
JPL MS 198-330

Joy Leonard
McDonnell Douglas Automt. & Robtcs.
5301 Bolsa Avenue
Huntington Beach, CA 92647

Bob LeRoy
General Electric
16891 Roque Lane
Huntington Beach, CA 92647

H. Garton Lewis, Jr.
Fairchild Weston Systems, Inc.
300 Robbins Lane
Syosset, NY 11791

Robert Lewis
NASA Kennedy Space Center
DM-MED-12
Kennedy Space Center, FL 32899

Larry Li
(address not available)

Jerry Lilienthal
JPL MS 158-224

Randel Lindemann
JPL MS 158-224

Ted Linden
Advanced Decision Systems
1500 Plymouth Street
Mountain View, CA 94043

Harvey Lipkin
Georgia Institute of Technology
Mechanical Engineering Dept.
Atlanta, GA 30332-0405

Todd Litwin
JPL MS 23

Anatole Lokshin
JPL MS 198-330

Luis Lopez
Teledyne Brown Engineering
300 Sparkman Drive
Huntsville, AL 35758-7007

Michael Lou
JPL MS 157-410

James W. Lowrie
Martin Marietta Space Systems Co.
Astronautics Group
P.O. Box 179, MC B1690
Denver, CO 80201

John Luh
Clemson University
Dept. of Elec. & Computer Engr.
Clemson, SC 29634

Ron Lumia
National Inst. of Standards & Tech.
Bldg. 220, Room B124
Gaithersburg, MD 20899

Ren C. Luo
North Carolina State University
Dept. of Elec. & Computer Engr.
Box 7911
Raleigh, NC 27695-7911

Piergiovanni Magnani
Tecnospazio S.p.A.
Via Mercantesse
3-20021 Baranzate di Bollate
Milan, Italy

David Marco
U.S. Naval Postgraduate School
Dept. of Mechanical Engineering
Monterey, CA 93943

Matthew T. Mason
Carnegie Mellon University
Computer Science Department
Schenley Park
Pittsburgh, PA 15213

Michael J. Massimino
Massachusetts Inst. of Technology
Room 20C-218
Cambridge, MA 02139

Yasuhiro Masutani
Osaka University
Dept. of Mechanical Engineering
1, Machikane-yama
Toyonaka, Osaka, 560 Japan

Toshio Matsumoto
Kitakyushu Fukuoka
Yahatahigashi Gyon, Japan

Douglas A. McAffee
JPL MS 303-308

Harry G. McCain
NASA Goddard Space Flight Center
Code 409
Greenbelt, MD 20771

J. Michael McCarthy
University of California, Irvine
Dept. of Mechanical Engineering
616 Engineering
Irvine, CA 92717

Don J. McFarland
JPL MS 303-308

John S. McLaughlin
The Aerospace Corporation
P.O. Box 92957
Los Angeles, CA 90009-2957

William S. McMath
Communications Research Centre
P.O. Box 11490, State H
Ottawa, Ontario, Canada K2H 8S2

Al Meintel
NASA Langley Research Center
MS 152D
Hampton, VA 23665

Alex Meystel
Drexel University
Dept. of Elec. & Computer Engr.
Philadelphia, PA 19104

David Miller
JPL MS 301-440

Andrew Mishkin
JPL MS 303-308

Tom Mitchell
Carnegie Mellon University
Computer Science Department
Schenley Park
Pittsburgh, PA 15213

Michael C. Moed
Rensselaer Polytechnic Institute
CIRSSE, CII 8313
Troy, NY 12180-3590

Melvin D. Montemerlo
NASA Headquarters
MS B647
Washington, D.C. 20546

Carlos Moreno
JPL MS 158-224

Carl Morimoto
GE Aerospace/Western Systems
4041 N. First St.
San Jose, CA 95134

Brian Muirhead
JPL MS 158-224

Ranjan Mukherjee
University of California, SB
Mech. & Environmental Engr. Dept.
6667 El Colegio Road, #23
Goleta, CA 93117

Frank Nagneron
(address not available)

Yoshihiko Nakamura
University of California
Mech. & Env. Engr. Dept.
Cntr./Robotic Systems in Microelec.
Santa Barbara, CA 93106

Mahmoud George Nasser
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

C. Allan Nathan
Grumman Space Division
Mail Stop A09-25
Bethpage, NY 11714

Dundee Navinchandra
Carnegie Mellon University
Robotics Institute
Pittsburgh, PA 15213-3890

Ashok Nedungadi
Southwest Research Institute
Culebra Road, #6220
San Antonio, TX 78284

Edward W. Ng
JPL MS 180-701

An H. Nguyen
University of California, Berkeley
Telerobotics Unit
481 Minor Hall
Berkeley, CA 94720

David Nitzan
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

Don Noon
JPL MS 158-224

John H. Norris
Grumman Corporation
5200 West Century Blvd., #980
Los Angeles, CA 90045

Dale Nussman
Dynamics Research Corporation
1755 Jefferson Davis Hwy., #802
Arlington, VA 22202

Maureen O'Brien
NASA Goddard Space Flight Center
Code 735.3
Greenbelt, MD 20771

Stan Ollendorf
NASA Goddard Space Flight Center
Office of Telerobotic Engineering
Code 706
Greenbelt, MD 20771

Irving J. Oppenheim
Carnegie Mellon University
Depts. of Architecture & Civil Engr.
Pittsburgh, PA 15213

David E. Orin
Ohio State University
Dept. of Electrical Engineering
2015 Neil Avenue
Columbus, OH 43210

Hiroshi Otake
JPL MS 198-105

Umit Ozguner
Ohio State University
Dept. of Electrical Engineering
2015 Neil Avenue
Columbus, OH 43210

Garrett Paine
JPL MS 510-202

Graham A. Parker
University of Surrey
Dept. of Mechanical Engineering
Guildford, Surrey, U.K. GU2 5XH

David Payton
Hughes Research Labs, AI Center
Bldg. 254, MS RL96
3011 Malibu Canyon Road
Malibu, CA 90265

Stephen Peters
JPL MS 301-250D

Keith Phillips
New Mexico State University
Department of Mathematics
Las Cruces, NM 88003

Dale E. Phinney
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058-3711

Paul Pierson
GE/Advanced Technology Laboratories
Moorestown Corporate Center
Bldg. 145-1, Route 38
Moorestown, NJ 08057

Donna Pivirotto
JPL MS 264-726

Ron P. Podhorodeski
University of Toronto
Dept. of Mechanical Engineering
5 King's College Road
Toronto, Ontario, Canada M5S 1A4

Farzad Pourboghrat
Southern Illinois University
Dept. of Electrical Engineering
Carbondale, IL 62901

Walter Prendin
Tecnomare S.p.A.
S. Marco 2091
30124 Venice, Italy

Charles R. Price
NASA Johnson Space Center
Mail Code EF2
Houston, TX 77058

Fritz Prinz
Carnegie Mellon University
Schenley Park
Pittsburgh, PA 15213

Ugo Racheli
Martin Marietta Co.
P.O. Box 179
Denver, CO 20201

Sudhendu Rai
California Institute of Technology
MC 104-44
Pasadena, CA 91125

Marc Raibert
Massachusetts Inst. of Technology
Artificial Intelligence Lab
545 Technology Square
Cambridge, MA 02139

Jim Randolph
JPL MS 264-726

Nageswara S.V. Rao
Old Dominion University
Dept. of Computer Science
Norfolk, VA 23529-0162

Alain Rappaport
Neuron Data
444 High Street
Palo Alto, CA 94301

Steven Raymus
GE Astro Space Division
P.O. Box 800
Princeton, NJ 08543-0800

Olivier Retali
MATRA Aerospace
37, av. Louis-Breguet, B.P. 1
F-78146 Velizy-Villacoublay Cedex, France

Akhavan-Leiliabadi Reza
ST Systems Corporation
4400 Forbes Blvd.
Lanham, MD 20706

Eric Rhodes
NASA Kennedy Space Center
PT-AST
Kennedy Space Center, FL 32815

Eric Rice
Orbital Technologies Corp.
P.O. Box 861
Middleton, WI 53562

James P. Rider
U.S. Bureau of Mines
Pittsburgh Research Center
P.O. Box 18070
Pittsburgh, PA 15320

Kenneth S. Roberts
Columbia University
Dept. of Computer Science
Box 122, CS Bldg.
New York, NY 10027

G. Rodriguez
JPL MS 198-330

Timo Ropponen
University of California
Cntr. Robotic Systems in Microelec.
Santa Barbara, CA 93106

Robert L. Rosenfeld
DARPA
1400 Wilson Blvd.
Arlington, VA 22209-2308

Stanley J. Rosenschein
Teleos Research
576 Middlefield Road
Palo Alto, CA 94301

Donald Rosenthal
NASA Ames Research Center
MS 244-17
Moffett Field, CA 94035

Michael Ross
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

Yuval Roth-Tabak
University of Michigan
Elec. Engr. & Comp. Sci. Dept.
150 ATL Bldg.
Ann Arbor, MI 48109

Alberto Rovetta
Polytechnic University of Milan
Mechanical Engineering Dept.
Piazza Leonardo Da Vinci 32
20133 Milan, Italy

Neil C. Rowe
U.S. Naval Postgraduate School
Dept. of Computer Science
Code 52Rp
Monterey, CA 93943

Douglas E. Ruth
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

Eric Rutten
IRISA/INRIA - Rennes
Campus de Beaulieu
F-35042 Rennes Cedex, France

Norman Sadeh
Carnegie Mellon University
Computer Science Dept.
Robotics Institute
Pittsburgh, PA 15213

Kenneth Salisbury
Massachusetts Inst. of Technology
AI Lab
545 Technology Square
Cambridge, MA 02139

Arthur C. Sanderson
Rensselaer Polytechnic Institute
Elect., Comp. & Sys. Engr. Dept.
Troy, NY 12180-3590

Budi Santoso
Catholic University of Leuven
Dept. of Mechanical Engineering
Celestijnenlaan 300B
B-3030 Leuven, Belgium

George N. Saridis
Rensselaer Polytechnic Institute
CIRSSE
Troy, NY 12180-3590

Timothy Sauerwein
NASA Goddard Space Flight Center
Code 735.1
Greenbelt, MD 20771

Bernd E. Schaefer
DLR (German Aerospace Rsch. Estab.)
Muenchnerstrasse
D-8031 Oberpfaffenhofen, W. Germany

Paul Schenkér
JPL MS 23

William H. Schiffbauer
U.S. Bureau of Mines
Pittsburgh Research Center
P.O. Box 18070, Cochrans Mill Road
Pittsburgh, PA 15236

C. Schleimer
Grumman Data Systems
Bethpage, NY 11714-3584

Eike Schmidt
(address not available)

Stan Schneider
Stanford University
Aerospace Robotics Laboratory
Room 250, Durand Building
Stanford, CA 94305

Klaus C.A. Schnirring
Dornier G.M.B.H.
P.O. Box 1420
7950 Friedrichshafen, W. Germany

Wayne Schober
JPL MS 180-701

Larry Schooley
University of Arizona
Electrical & Computer Engr. Dept.
Tucson, AZ 85721

Anna Sciomachen
Tecnospazio S.p.A.
Via Mercantesse
3-20021 Baranzate di Bollate
Milan, Italy

Warren P. Seering
Massachusetts Inst. of Technology
Mechanical Engr. Dept., NE43-835
545 Technology Square
Cambridge, MA 02139

H. Seraji
JPL MS 198-330

William Shanney
The Aerospace Corporation
P.O. Box 3430
Sunnyvale, CA 94088-3430

Lejun Shao
University of Michigan
Dept. of Elec. Engr. & Comp. Sci.
Ann Arbor, MI 48109

Daniel G. Shapiro
Advanced Decision Systems
1500 Plymouth
Mountain View, CA 94043

Kevin Shelburne
McDonnell Douglas
16055 Space Center Blvd.
Houston, TX 77062

Thomas B. Sheridan
Massachusetts Inst. of Technology
Man-Machine Systems Laboratory
Bldg. 3, Room 346
Cambridge, MA 02139

Zv. Shiller
Univ. of California, Los Angeles
Mechanical Engineering
5732 Boelter Hall
Los Angeles, CA 90024

Isao Shimoyama
University of Tokyo
Mechanical Engineering Dept.
7-3-1 Hongo, Bunkyo-ku
Tokyo 133, Japan

Gary Silverman
IBM Corporation
11601 Wilshire Blvd.
Los Angeles, CA 90025

Reid Simmons
Carnegie Mellon University
Computer Science Department
Schenley Park
Pittsburgh, PA 15213

Herbert Simon
JPL MS 301-270

Mike Sklar
McDonnell Douglas Space Systems Co.
P.O. Box 21233
Dept. F880
Kennedy Space Center, FL 32812

Marc Slack
JPL MS 301-440

Ernest E. Smith, Jr.
NASA Johnson Space Center
Syst. Div./Mech. & Crew Syst. Br.
Houston, TX 77058

Aram Soghikian
T.Q. Automation
1441 155th Avenue
San Leandro, CA 94578

Donald I. Soloway
NASA Langley Research Center
MS 152D
Hampton, VA 23665-5225

Barry Soroka
JPL MS 198-330

Thomas H. Speeter
AT&T Bell Laboratories
Machine Percp. Research Dept.
Room 4E-638, Crawfords Corner Road
Holmdel, NJ 07733

John Spofford
Martin Marietta Astronautics
P.O. Box 179
MS 4372
Denver, CO 80201

N.S. Sridharan
FMC Corporation
Central Engineering Labs
Box 580, 1205 Coleman Avenue
Santa Clara, CA 95052

Richard Stanton
JPL MS 198-105

Lawrence Stark
University of California/Berkeley
Telerobotics Unit
481 Minor Hall
Berkeley, CA 94720

Gregory Starr
University of New Mexico
Mechanical Engineering Dept.
Albuquerque, NM 87131

John Staudhammer
University of Florida
Dept. of Electrical Engineering
550A Weil Hall
Gainesville, FL 32611

Brian D. Staunton
The Aerospace Corporation
3444 W. 171st Street
Torrance, CA 90504

Harry E. Stephanou
George Mason University
School of Infor. Tech. & Engr.
Fairfax, VA 22030

R. Rhoads Stephenson
JPL MS 198-105

Henry W. Stone
JPL MS 198-330

Phillip H. Sutter
Franklin and Marshall College
P.O. Box 3003
Lancaster, PA 17604-3003

Scott Swetz
ST Systems Corp.
4400 Forbes Road
Lanham, MD  20706

Zoltan Szakaly
JPL MS 198-330

Susumu Tachi
MITI
Mechanical Engineering Lab
1-2, Namiki
Tsukuba Science City 305, Japan

Mahmoud Tarokh
University of California, San Diego
California Space Institute
Mail Code A-016
La Jolla, CA  92093

Janice Tarrant
ST Systems Corp.
1577 Spring Hill Road
Suite 500
Vienna, VA  22180

Austin Tate
University of Edinburgh
AI Applications Institute
80 South Bridge
Edinburgh, U.K.  EH1 1HN

Russell W. Taylor
IBM Research
P.O. Box 704, H1-L06
Yorktown Heights, NY  10598

Antonio Terribile
Technomare S.p.A.
S. Marco 3584
30124 Venice, Italy

Delbert Tesar
University of Texas at Austin
Dept. of Mechanical Engineering
Austin, TX  78712

Hal Tharp
University of Arizona
Dept. of Electrical & Comp. Engr.
Tucson, AZ  85721

Jack M. Thompson, Jr.
Robotics Research Corporation
5400 DuPont Circle, TechneCenter
Milford, OH  45150

Scott W. Tilley
Ford Aerospace Corp.
Space Systems Division
3825 Fabian Way
Palo Alto, CA  94303

Chris Toffales
Fairchild Weston Systems, Inc.
300 Robbins Lane
Syosset, NY  11791

Don Trask
JPL MS 238-700

Jeffrey Trinkle
University of Arizona
Electrical & Comp. Engr. Dept.
Tucson, AZ  85721

Jocelyn Troccaz
(address not available)

Kam Tso
JPL MS 198-330

Kazuyoshi Tsutsumi
Kobe University
Grad. School of Science & Tech.
Division of System Science
Rokkodai, Kobe 657, Japan

Chris M. Tuck
University of Michigan
Artificial Intelligence Lab.
1101 Beal Avenue
Ann Arbor, MI  48109-2110

Takashi Uchiyama
Fujitsu Laboratory, Ltd.
1015 Kamikodanaka, Nakahara-ku
Kawasaki, Kanagawa 211, Japan

Marc Ullman
Stanford University
Aerospace Robotics Laboratory
Room 250, Durand Building
Stanford, CA  94305

Nader Vadiee-Haghighi
University of New Mexico
EECE Dept., CAD Lab Syst./Rob.
Albuquerque, NM 87131

Zia Vafa
GE - CRD
P.O. Box 8, KW-0209
1 River Road
Schenectady, NY 12301

R. Vaillant
INRIA Domaine de Voluceau
BP 105
78153 Le Chesnay Cedex, France

Kimon Valavanis
Northeastern University
Dept. of Electrical & Comp. Engr.
Boston, MA 02115

H. Van Brussel
Catholic University of Leuven
ME Dept., Mech. Construction & Prod.
300B Celestijnenlaan
B-3030 Leuven, Belgium

Evelyn E. Vance
Center for Naval Analyses
4401 Ford Avenue
P.O. Box 16268
Alexandria, VA 22302-0268

Jan Vandenbrande
University of Southern California
5222 Steveann Street
Torrance, CA 90503

Giulio Varsi
JPL MS 180-701

Steve Venema
JPL MS 198-330

Subramanian T. Venkataraman
JPL MS 198-330

Dan Venolia
Apple Computer
20705 Valley Green Drive
Cupertino, CA 95014

Herbert E.M. Viggh
Boeing Aerospace
P.O. Box 3999, MS 82-58
Seattle, WA 98124

Frank Vigneron
Communications Research Centre
Box 11490, Station H
Ottawa, Ontario, Canada K2K-1N8

Havard I. Vold
Robotics Research Corporation
5400 DuPont Circle, TechneCentre
Milford, OH 45150

Richard A. Volz
Texas A&M University
Dept. of Computer Science
Room 241, Zachry Engr. Center
College Station TX 77843

Henry Voss
Alberta Research Council
3rd Floor
6815 8th St., N.E.
Calgary, Alberta, Canada T2E 7H7

Marko I. Vuskovic
San Diego State University
Dept. of Math. Sciences
San Diego, CA 92182-0314

Eberhardt Waffenschmidt
Firma Dornier GMBH
Postfach 1420
D-7990 Friedrichshafen 1
West Germany

Kenneth J. Waldron
Ohio State University
Dept. of Mechanical Engineering
Columbus, OH 43210

Doug Walker
GHG Corporation
EVA Robotics
1300 Hercules, S. 111
Houston, TX 77058

Ian D. Walker
University of Texas at Austin
Dept. of Elec. & Computer Engr.
P.O. Box 7721
Austin, TX 78713

Michael W. Walker
University of Michigan
Elec. Engr. & Comp. Sci. Dept.
Robotics Research Laboratory
Ann Arbor, MI 48109-2110

Yulun Wang
University of California, SB
CRSM
Santa Barbara, CA 93106

Texas M. Ward
Lockheed Engineering & Sciences Co.
2400 NASA Road 1
Houston, TX 77058

Albert J. Wavering
National Inst. of Standards & Tech.
Bldg. 220, Room B-127
Gaithersburg, MD 20899

D. Wegerif
McDonnell Douglas Space Systems
P.O. Box 21233, Dept. F880
Kennedy Space Center, FL 32812

Charles R. Weisbin
Oak Ridge National Laboratory
Engr., Physics & Mathematics Div.
P.O. Box 2008
Oak Ridge, TN 37831-6364

Jeffrey H. Welsh
U.S. Bureau of Mines
Pittsburgh Research Center
P.O. Box 18070, Cochrans Mill Rd.
Pittsburgh, PA 15236

John T. Wen
Rensselaer Polytechnic Institute
CII 8229, ECSE Department
Troy, NY 12180

Thomas E. Wheatley
National Inst. of Standards & Tech.
Bldg. 220, Room B124
Gaithersburg, MD 20899

Leslie A. White
JPL MS 233-302

Steve D. Whitehead
University of Rochester
Computer Science Department
Rochester, NY 14627

Steven F. Wiker
University of Wisconsin
Dept. of Industrial Engineering
Madison, WI 53706

Brian Wilcox
JPL MS 23

Michael Will
GE - Western Systems
4041 N. First Street
San Jose, CA 95134

Daniel M. Williams
Oak Ridge National Laboratory
121 E. Irving Lane
Oak Ridge, TN 37830

Roy E. Williams
Center for Engineering Applications
6401 Poplar Avenue, Suite 190
Memphis, TN 38119

Peter Wong
TRW, R10/1368H
One Space Park
Redondo Beach, CA 90278

Laurie Wood
JPL MS 278

Gary Woods
Boeing/KSC
Kennedy Space Center, FL 32899

Eric S. Yager
University of Michigan
AI Laboratory
Ann Arbor, MI 48109-2110

Shinichi Yamamura
228 S. Central Avenue, #218
Los Angeles, CA 90012

Kazuhiko Yokoyama
6-3-104 Kosagida, Yahatanishi-ku
Kitakyushu, Japan

Kazuya Yoshida
Tokyo Institute of Technology
Dept. of Mechanical Engr. Science
2-12-1, O-Okayama, Meguro-ku
Tokyo 152, Japan

Stephen Yurkovich
Ohio State University
Dept. of Electrical Engineering
2015 Neil Avenue
Columbus, OH 43210

Xichi Zheng
International Submarine Engr. Ltd.
1734 Broadway Street
Port Coquitlam, British Columbia
W3C 2M8 Canada

David Zhu
Stanford University
2056 Hanover Street
Palo Alto, CA 94306

Wayne F. Zimmerman
JPL MS 303-308

Mohamed Zribi
Purdue University
EE Building, Box 289
West Lafayette, IN 47907

Monte Zweben
NASA Ames Research Center
Moffett Field, CA 94035

## TECHNICAL REPORT STANDARD TITLE PAGE

| | | |
|---|---|---|
| 1. Report No. 89-7 | 2. Government Accession No. | 3. Recipient's Catalog No. |

| | |
|---|---|
| 4. Title and Subtitle<br>PROCEEDINGS OF THE NASA CONFERENCE ON SPACE TELEROBOTICS (VOLUMES I-V) | 5. Report Date<br>January 31, 1989 |
| | 6. Performing Organization Code |
| 7. Author(s)<br>G. Rodriguez and H. Seraji (editors) | 8. Performing Organization Report No. |
| 9. Performing Organization Name and Address<br>JET PROPULSION LABORATORY<br>California Institute of Technology<br>4800 Oak Grove Drive<br>Pasadena, California 91109 | 10. Work Unit No. |
| | 11. Contract or Grant No.<br>NAS7-918 |
| | 13. Type of Report and Period Covered<br>JPL Publication |
| 12. Sponsoring Agency Name and Address<br>NATIONAL AERONAUTICS AND SPACE ADMINISTRATION<br>Washington, D.C. 20546 | 14. Sponsoring Agency Code |

15. Supplementary Notes

16. Abstract

These proceedings contain papers presented at the NASA Conference on Space Telerobotics held in Pasadena, January 31 - February 2, 1989. The Conference was sponsored by the NASA Office of Aeronautics and Space Technology, together with ARC, LRC, GSFC, JSC, MSFC, KSC and JPL. The theme of the Conference was man-machine collaboration in space. The Conference provided a forum for researchers and engineers to exchange ideas on the research and development required for application of telerobotics technology to the space systems planned for the 1990s and beyond. The Conference: (i) provided a view of current NASA telerobotic research and development; (ii) stimulated technical exchange on man-machine systems, manipulator control, machine sensing, machine intelligence, concurrent computation, and system architectures; and (iii) identified important unsolved problems of current interest which can be dealt with by future research. There were about 500 international participants including about 100 from abroad.

| | |
|---|---|
| 17. Key Words (Selected by Author(s))<br>Engineering | 18. Distribution Statement<br>Unclassified; unlimited |

| | | | |
|---|---|---|---|
| 19. Security Classif. (of this report)<br>Unclassified | 20. Security Classif. (of this page)<br>Unclassified | 21. No. of Pages<br>2,386 | 22. Price |