JPL Publication 89-7, Vol. IV

Proceedings of the NASA Conference on Space Telerobotics

Volume IV

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. IV

Proceedings of the NASA Conference on Space Telerobotics

Volume IV

G. Rodriguez
H. Seraji
Editors

January 31, 1989

NASA

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

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

MANIPULATOR CONTROL 1

AN IMPROVED ADAPTIVE CONTROL FOR REPETITIVE MOTION OF ROBOTS

F. Pourboghrat

Department of Electrical Engineering
Southern Illinois University
Carbondale, Illinois 62901-6603

Abstract

An adaptive control algorithm is proposed for a class of nonlinear systems, such as robotic manipulators, which is capable of improving its performance in repetitive motions. When the task is repeated, the error between the desired trajectory and that of the system is guaranteed to decrease. The design is based on the combination of a direct adaptive control and a learning process. This method does not require any knowledge of the dynamic parameters of the system.

1. Introduction

The position servo control is an important and basic problem in the successful operation of robot manipulators. Many methodologies regarding the solution of this problem have appeared in the literature. Recently, the interest in adaptive control of robot manipulators has been growing noticeably [1,3]. This growth is mainly due to the fact that, unlike the non-adaptive control methods, the adaptive control strategies do not require the explicit knowledge of robot dynamics parameters.

On the other hand, recently some works have been reported for the generation of the controlling input for the repetitive motion of dynamical systems [4,6]. These are called learning controllers because the control input generated this way is improved through repeated trials. The method proposed in [4] requires the derivative of the error function in the learning process to guarantee the uniform convergence. Moreover, the conditions on the system's transfer function is very restrictive, and it also requires that the system's inverse dynamics be proper and stable. In [5], the concept of a dual system is used for the recursive generation of the input. This result is mainly applied to linear, time-invariant systems and the conditions for it's convergence are not restrictive. However, the construction of the dual system requires the knowledge of the original system, and it is also not practical. In [6], an adaptive learning controller is designed which can be applied to robotic manipulators and can be easily implemented.

In this paper the concept of learning control is applied to the problem of model reference adaptive control of manipulators. Our objective is to design an adaptive controller capable of learning to improve its performance in repetitive motions. The approach is similar to that used in [6], with a slight modification where the concept of inner-loop/outer-loop control is used. The proposed adaptive controller can be applied to a robot manipulator in non-repetitive motion, in which case it performs as a standard model reference adaptive controller. But when it is commanded to perform a task repeatedly, the learning controller improves its performance in subsequent motions.

We first develop a model reference adaptive control strategy [1,2] to the robot manipulator so that the resulting closed-loop system is equivalent to the preselected reference model. The model reference adaptive controller designed in [1] is shown to force the robot dynamics to follow those of a predetermined linear time-invariant reference model very closely.

Then we develop a new learning controller for a linear time-invariant system L, with guaranteed convergence under very mild conditions. This is similar to the learning controller proposed in [6], which is based on the existance of an auxiliary system L* such that its composition with the original system (i.e., LL*), is positive real. It was shown in [6] that such an auxiliary system can always be found if the original system is stable.

Finally, we apply our learning strategy, as an outer-loop control, to the linearized inner-loop system resulted from applying the model reference adaptive controller to the robot. The proposed control system is shown in Figure 1. The overall closed-loop system is shown to be asymptotically stable, and does not require any knowledge about the dynamic parameters of the robot. The error between the desired response and the actual robot response is guaranteed to approach zero after executing the desired task repeatedly.

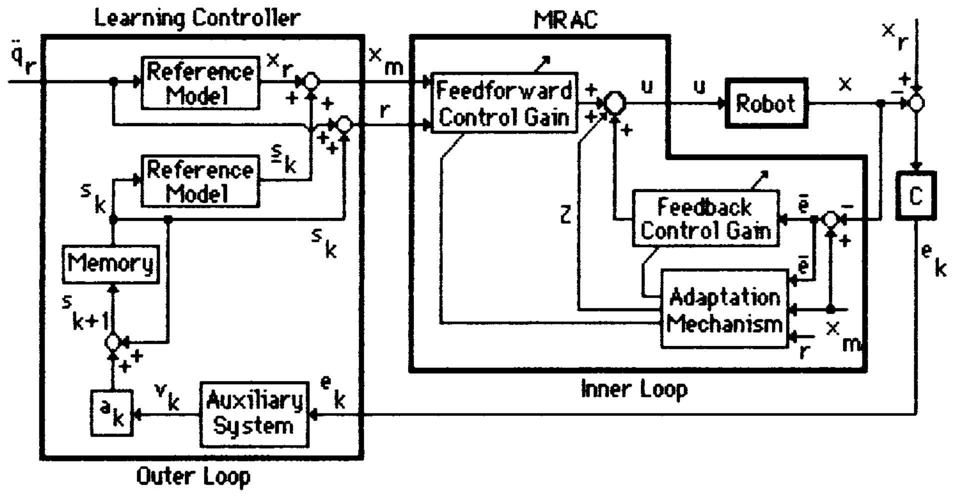


Figure 1

2. Robot Dynamics

The dynamic equation of a robot manipulator is highly nonlinear and is given by

$$M(q)\,q'' + H(q,q')\,q' + g(q) = u \tag{1}$$

where q(t) is the nx1 vector of joint angles, M(q) is the nxn symmetric, positive definite inertia matrix, H(q,q')q' is the nx1 Coriolis and centrifugal force vector, g(q) is the nx1 gravitational force vector, and u is the nx1 applied torque vector for the joint actuators. It can easily be shown that the above equation can also be written as

$$\begin{aligned} x' &= A(x)\,x + B(x)\,u \\ y &= C\,x \end{aligned} \tag{2}$$

where

$$x(t) = \begin{bmatrix} q \\ q' \end{bmatrix}$$

$$A(x) = \begin{bmatrix} 0 & I \\ A_0(x) & A_1(x) \end{bmatrix} \quad , \quad B(x) = \begin{bmatrix} 0 \\ B_0(x) \end{bmatrix}$$

$$C = [C_1 , C_2]$$

for some C_1, C_2, such that x(t) is 2nx1 state vector, u(t) is the nx1 input torque vector and

$A_0 = M^{-1}G$, $A_1 = {}^{-1}H$, $B_0 = M^{-1}$, and $g = Gq$.

3. Inner-Loop Adaptive Control

Consider the robotic system, given by equation (2). Let the dynamic equation of a reference model be given by a minimal, 2n-th order, stable linear system

$$\begin{aligned} x'_m &= A_m\, x_m + B_m\, r \\ y_m &= C_m\, x_m \end{aligned} \tag{3}$$

where x_m is the 2nx1 model state vector, u is the nx1 reference model's input vector,

$$A_m = \begin{bmatrix} 0 & I \\ A_{m0} & A_{m1} \end{bmatrix} \quad , \quad B_m = \begin{bmatrix} 0 \\ B_{m0} \end{bmatrix}$$

$$C_m = C = [C_1, C_2].$$

Now let us define the state error to be

$$\underline{e} = x_m - x\,. \tag{4}$$

Then the dynamic equation of the state error is given by

$$\begin{aligned} \underline{e}' &= A\,\underline{e} + (A_m - A)\,x_m + B_m\, r - h - B\,u \\ e &= C\,\underline{e} \end{aligned} \tag{5}$$

where h is assumed to be any unmodeled disturbances not already considered in the robot dynamics, and $e = y_m - y$ is the output error.

The problem is to design an adaptive controller so that error equation (5) is asymptotically stable, where the state error $\underline{e}$, and hence the output error e, approach zero as time increases.

Result 1

Consider the robot manipulator given by the dynamic equation (2), and the linear, time-invariant, stable, and controllable reference model, given by equation (3). Let the adaptive controller be given by

$$u = K_e \underline{e} + K_x x_m + K_r r + z \qquad (6)$$

with the adaptation law

$$\begin{aligned} K'_e &= \alpha_1 E P \underline{e}\,\underline{e}^T + \alpha_2 d/dt(E P \underline{e}\,\underline{e}^T) \qquad (7) \\ K'_x &= \beta_1 E P \underline{e}\, x_m^T + \beta_2 d/dt(E P \underline{e}\, x_m^T) \\ K'_r &= \gamma_1 E P \underline{e}\, r^T + \gamma_2 d/dt(E P \underline{e}\, r^T) \\ z' &= \eta_1 E P \underline{e} + \eta_2 d/dt(E P \underline{e}) \end{aligned}$$

where $\alpha_1, \beta_1, \gamma_1, \eta_1 > 0$, $E = [0, I]$, and P and Q are symmetric positive definite matrices such that for some stable matrix D of the designer's choice, we have

$$PD + D^T P = -Q\,. \qquad (8)$$

The above adaptive controller results in an asymptotically stable system such that the dynamics of the closed-loop robotic system follow that of the linear reference model. That is, the state error $\underline{e} = x_m - x$, and hence the output error $e = y_m - y$, approach zero as time increases.

Proof

The proof uses Liapunov's direct method for stability, and can be found in [1]. □

The above result indicates that using the proposed adaptive control, the dynamics of the nonlinear robot in the closed-loop matches that of the predetermined, stable, linear, time-invariant reference model. Therefore, the inner-loop control in Figure 1 can be assumed to approximately have the same dynamics as the linear reference model.

4. Outer-Loop Learning Control

Let us consider a linear, stable, time-invariant dynamical system, given by

$$x' = A\,x + B\,u \qquad (9)$$
$$y = C\,x$$

where x is the n-dimensional state vector, u is the m-dimensional input vector, and y is the m-dimensional output vector. The above system can also be denoted by the linear operator L, where $y = Lu$.

Now let $Y_d(t)$ be the desired output function (trajectory) of the system over the interval [0,T], with the initial state $x(0) = x_0$. Assume that $u_k(t)$ and $y_k(t)$ are the corresponding input and output functions of system (9) over the time interval [0,T] in trial k, with the initial state x_0. Then the learning control strategy is the updating rule that generates the input function $u_{k+1}(t)$ for the interval [0,T] from the knowledge of $u_k(t)$ and the error $e_k(t) = y_d(t) - y_k(t)$. This is then applied to system (9) with the same initial state x_0 at trial k+1 to drive the error function $e_k(t)$ to zero.

Let us define the norm and the inner-product to be given by

$$\langle x_1 , x_2\rangle = \int_0^T x_1^T(t)\, x_2(t)\, dt$$
$$\|x\|^2 = \int_0^T x^T(t)\, x(t)\, dt$$

Then the auxiliary system is defined to be a linear, stable, time-invariant system, given by

$$\underline{z}' = F\,\underline{z} + G\,e \qquad (10)$$
$$v = H\,\underline{z} + \underline{E}\,e$$

or equivalently by the operator L*, as

$$v = L^*\, e$$

such that LL* is a positive real operator. That is

$$\langle x , LL^* x\rangle > 0, \text{ for all } x . \qquad (11)$$

Now let the dynamics of a reference trajectory be given by the linear operator L such that

$$y_r = L\ s \qquad (12a)$$

or equivalently

$$x'_r = A_r\, x_r + B_r\, s \qquad (12b)$$
$$y_r = C_r\, x_r$$

where $s=q''_r$, and

$$x_r = \begin{bmatrix} q_r \\ q'_r \end{bmatrix}$$

$$A(x) = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, \quad B(x) = \begin{bmatrix} 0 \\ I \end{bmatrix}$$

$$C_r = C = [C_1 \quad C_2].$$

Utilizing the definition of auxiliary system (10), we can show the following result.

Result 2

Consider the linear, time-invariant, minimal, and stable reference system (12), denoted by operator L. Suppose that a linear auxiliary system, denoted by operator L*, is such that the operator LL* is positive real. Let the learning control strategy for trial k+1 be given by

$$s_{k+1}(t) = s_k(t) + a_k v_k(t) \tag{13}$$

with $s_0(t)=q''_r(t)$, where

$$a_k = \langle e_k(t), LL^* e_k(t)\rangle / \|LL^* e_k(t)\|^2$$

$$e_k(t) = y_r(t) - y_k(t)$$

and $v_k(t)$ is the output of the linear operator L*, i.e., $v_k = L^* e_k$, with zero initial values. The above learning controller is convergent in the sense that $e_k(t)$ vanishes over the interval [0,T] as the number of trials k increases.

Proof

Consider the linear system (12) with input (13). Now let a discrete Liapunov candidate be given by

$$J_k = \|e_k(t)\|$$

where $e_k(t) = y_r(t) - y_k(t)$ is the error at trial k. Then we have

$$\begin{aligned}\Delta J_k &= J_{k+1} - J_k \\ &= \|e_{k+1}(t)\|^2 - \|e_k(t)\|^2 \\ &= \|e_k(t) - a_k L v_k(t)\|^2 - \|e_k(t)\|^2 \\ &= a_k^2 \|L v_k(t)\|^2 - 2 a_k \langle e_k(t), L v_k(t)\rangle \\ &= a_k^2 \|L L^* e_k(t)\|^2 - 2 a_k \|e_k(t), L L^* e_k(t)\|^2 .\end{aligned}$$

Now taking $a_k = \langle e_k , L L^* e_k\rangle / \|L L^* e_k\|^2$, it is easy to check that ΔJ is minimized with respect to a_k, and hence $\Delta J_k < 0$. Therefore, from the discrete Liapunov method, the learning controller (13) guarantees that the error function $e_k(t)$ approaches zero as the number of trials k increases. □

5. Learning Adaptive Control

Combining Results 1 and 2, it is possible to design a learning adaptive controller for the robotic system. It is desired now that the robot output $y = C x$ follow $y_r = C_r x_r$ in repeated trials. From Figure 1, it can be seen that $x_m = x_r + \underline{s}$, where $\underline{s}_k = [s_k^T, s'_k{}^T]^T$ and $r = q''_r + s_k$. That is the dynamic equation of the reference model can be written as

$$\begin{aligned} x''_m &= A_r x_m + B_r r \\ y_m &= C_r x_m . \end{aligned} \tag{14}$$

Now, we have the following result.

<u>Result 3</u>

Consider the robot manipulator given by the dynamic equation (2), and the linear, time-invariant, stable, and controllable reference model given by the operator L as in equation (14). Let the learning adaptive controller be given by

$$u = K_e \underline{e} + K_x x_m + K_r r + z \tag{15}$$

with the adaptation law

$$\begin{aligned} K'_e &= \alpha_1 E P \underline{e}\,\underline{e}^T + \alpha_2 d/dt(E P \underline{e}\,\underline{e}^T) \\ K'_x &= \beta_1 E P \underline{e}\, x_m^T + \beta_2 d/dt(E P \underline{e}\, x_m^T) \\ K'_r &= \delta_1 E P \underline{e}\, r^T + \delta_2 d/dt(E P \underline{e}\, r^T) \\ z' &= \eta_1 E P \underline{e} + \eta_2 d/dt(E P \underline{e}) \end{aligned}$$

where the state error $\underline{e} = x_m - x$ is the error between the system and the model states. Also

$$s_{k+1}(t) = s_k(t) + a_k v_k(t) \qquad (16)$$

with $s_0(t)=q''_r(t)$ which is generated by reference dynamics equation (12), and

$$a_k = \langle e_k(t), LL^* e_k(t)\rangle / \|LL^* e_k(t)\|^2$$

and also

$$v(k) = L^* e_k(t)$$

with zero initial values, such that LL^* is positive real, $C_r=C=[C_1,C_2]$, and $e_k(t) = y_r(t) - y_k(t)$ is the error function between the desired output trajectory y_r and the robot's output trajectory y in the k-th trial. The above learning adaptive controller is convergent in the sense that starting from the same initial state x_0, the output error $e_k(t)$, and equivalently $e(t)$ and $e'(t)$, , approach zero over the interval [0,T] as the number of trials increases.

Proof

The proof is merely a combination of the proofs of Results 1 and 2, and is found in [6]. □

6. Conclusions

A new adaptive control is proposed, that is capable of improving its tracking performance in repetitive motions. The design can be applied to a class of nonlinear systems that includes robotic manipulators. The controller guarantees that the error between a desired trajectory and the robot's trajectory approaches zero as the number of trials increases. The proposed controller does not require any knowledge of the dynamic parameters of the robot and can be easily implemented.

7. References

[1] Seraji, H., "A New Approach to Adaptive Control of Manipulators," ASME, J. Dyn. Syst. Meas. Cont., Vol. 109, 1987.
[2] Pourboghrat, F., "Adaptive Model Following Control for Robotic Arms Using the Second Method of Liapunov," IEEE, Int. Symp. Circ. Syst., 1986.
[3] Oh, B.J., Jamshidi, M., and Seraji, H., "Decentralized Adaptive Control," IEEE, Int. Conf. Robotics Autom., 1988.
[4] Arimoto, S., Kawamura, S., and Miyazaki, F., "Bettering Operation of Dynamic Systems by Learning: A New Control Theory for Servomechanism or Mechatronics Systems," Proc., IEEE 23rd Conf. Dec. Cont., 1984.
[5] Furuta, K. and Yamakita, M., "Iterative Generation of Optimal Input of A Manipulator," Proc., IEEE Int. Conf. Robotics Autom., 1986.
[6] Pourboghrat, F., "Adaptive Learning Control for Robots," Proc., IEEE Int. Conf. Robotics Autom., 1988.

DIRECT ADAPTIVE CONTROL OF A PUMA 560 INDUSTRIAL ROBOT

Homayoun Seraji, Thomas Lee
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109, USA

Michel Delpech
DRT/TVE/EA
Centre National d'Etudes Spatiales
Toulouse 31055
France

Abstract

The paper describes the implementation and experimental validation of a new direct adaptive control scheme on a PUMA 560 industrial robot. The testbed facility consists of a Unimation PUMA 560 six-jointed robot and controller, and a DEC MicroVAX II computer which hosts the RCCL (Robot Control "C" Library) software. The control algorithm is implemented on the MicroVAX which acts as a digital controller for the PUMA robot, and the Unimation controller is effectively bypassed and used merely as an I/O device to interface the MicroVAX to the joint motors. The control algorithm for each robot joint consists of an auxiliary signal generated by a constant-gain PID controller, and an adaptive position-velocity (PD) feedback controller with adjustable gains. The adaptive independent joint controllers compensate for the inter-joint couplings and achieve accurate trajectory tracking without the need for the complex dynamic model and parameter values of the robot. Extensive experimental results on PUMA joint control are presented to confirm the feasibility of the proposed scheme, in spite of strong interactions between joint motions. Experimental results validate the capabilities of the proposed control scheme. The control scheme is extremely simple and computationally very fast for concurrent processing with high sampling rates.

1. Introduction

During the past decade, the control of robot manipulators has been the focus of considerable research, and many different control schemes have been suggested in the literature. With a few exceptions, all existing schemes are tested on manipulators through computer simulations *only*, often using the popular two-link arm paradigm. Although the simulations are useful for illustration and proof-of-concept, practical issues such as effect of friction, limitation on controller gains, and sampling rate constraint are often neglected. Despite the large number of proposed manipulator control schemes, the number of schemes that have actually been experimentally evaluated on manipulators, and particularly on industrial robots, is very small today.

This paper describes the implementation and experimental validation of a newly developed direct adaptive control scheme [1,2] on a six-jointed PUMA 560 industrial robot. The control scheme has a decentralized structure and consists of a number of simple local feedback controllers. Each local controller consists of an auxiliary signal generated by a constant-gain PID controller, and an adaptive position-velocity (PD) feedback controller whose gains are updated on-line in real time. The control scheme is implemented on a MicroVAX II computer using the RCCL software, and generates the control signals that drive the PUMA joint motors directly. This simple control scheme is *not* based on the complex PUMA dynamic model and is therefore implemented at a high sampling rate and yields a good tracking performance.

The paper is structured as follows. In Section 2, an overview of the design theory for adaptive joint controllers is given. The descriptions of the testbed facility at JPL and the RCCL software are given in Section 3. Section 4 discusses the experimental results on simultaneous control of all six joint angles of the PUMA 560 industrial robot. The conclusions drawn from the paper are discussed in Section 5.

2. Theory Overview

In this section, the design theory for direct adaptive control of manipulators is outlined. The proposed control scheme has a decentralized structure, where each manipulator joint is controlled independently of the others by use of a local feedback controller. Therefore, the dynamics of each joint will be considered separately.

Consider a robot manipulator with the n joint angles $\theta = [\theta_1, \theta_2, \ldots, \theta_n]$ and the corresponding n joint torques $T = [T_1, T_2, \ldots, T_n]$. The dynamic model of the i^{th} manipulator joint which relates θ_i to T_i can be represented by the second-order nonlinear differential equation

$$m_{ii}(\theta)\ddot{\theta}_i + \sum_{\substack{j=1 \\ \neq i}}^{n} m_{ij}(\theta)\ddot{\theta}_j + n_i(\theta, \dot{\theta}) + g_i(\theta) + h_i(\dot{\theta}_i) = T_i \tag{1}$$

where m_{ij} is the $(i,j)^{th}$ element of the inertia matrix, and n_i, g_i, h_i are the i^{th} elements of the Coriolis/centrifugal vector, gravity vector, and friction vector, respectively. The terms in equation (1) are highly complicated nonlinear functions of the manipulator configuration θ, the speed of motion $\dot{\theta}$, and the payload inertial parameters. The manipulator control problem is to generate the joint torques $T_i(t)$, for $i = 1, \ldots, n$, such that the joint angles $\theta_i(t)$ track some desired trajectories $\theta_{di}(t)$ as closely as possible.

In the proposed decentralized control scheme, the i^{th} joint is controlled by the local adaptive feedback control law [2]

$$T_i(t) = f_i(t) + k_{pi}(t)e_i(t) + k_{vi}(t)\dot{e}_i(t) \tag{2}$$

as shown in Figure 1, where $e_i(t) = \theta_{di}(t) - \theta_i(t)$ and $\dot{e}_i(t) = \dot{\theta}_{di}(t) - \dot{\theta}_i(t)$ are the position and velocity tracking-errors, and the controller terms are given by

Weighted tracking-error

$$r_i(t) = w_{pi}e_i(t) + w_{vi}\dot{e}_i(t) \tag{3}$$

Auxiliary signal

$$f_i(t) = f_i(0) + \delta_i \int_0^t r_i(t)dt + \rho_i r_i(t) \tag{4}$$

Position feedback gain

$$k_{pi}(t) = k_{pi}(0) + \alpha_i \int_0^t r_i(t)e_i(t)dt + \beta_i r_i(t)e_i(t) \tag{5}$$

Velocity feedback gain

$$k_{vi}(t) = k_{vi}(0) + \gamma_i \int_0^t r_i(t)\dot{e}_i(t)dt + \lambda_i r_i(t)\dot{e}_i(t) \tag{6}$$

In equations (3)-(6), $\{\delta_i, \alpha_i, \gamma_i\}$ are positive scalar integral adaptation gains, $\{\rho_i, \beta_i, \lambda_i\}$ are non-negative scalar proportional adaptation gains, and $\{w_{pi}, w_{vi}\}$ are scalar positive weighting factors of the position and velocity tracking-errors.

From the implementation viewpoint, the auxiliary signal $f_i(t)$ can be generated by a constant-gain PID feedback controller acting on the position tracking-error $e_i(t)$, since from equations (3) and (4), $f_i(t)$ can be expressed as

$$\begin{aligned} f_i(t) &= f_i(0) + \rho_i \left[w_{pi}e_i(t) + w_{vi}\dot{e}_i(t)\right] + \delta_i \int_0^t \left[w_{pi}e_i(t) + w_{vi}\dot{e}_i(t)\right] dt \\ &= f_i(0) + \left[\rho_i w_{pi} + \delta_i w_{vi}\right] e_i(t) + \left[\rho_i w_{vi}\right] \dot{e}_i(t) + \left[\delta_i w_{pi}\right] \int_0^t e_i(t)dt \end{aligned} \tag{7}$$

Hence, the joint torque (2) can be generated by the adaptive PID feedback controller

$$T_i(t) = T_i(0) + [\overline{k}_{pi}(t) + \overline{k}_{Ii} \int_0^t dt + \overline{k}_{vi}(t)\frac{d}{dt}]e_i(t) \tag{8}$$

where $\overline{k}_{pi}(t) = k_{pi}(t) + \rho_i w_{pi} + \delta_i w_{vi}$, $\overline{k}_{Ii} = \delta_i w_{pi}$, and $\overline{k}_{vi}(t) = k_{vi}(t) + \rho_i w_{vi}$ are the adjustable PID gains and $T_i(0) = f_i(0)$ is the initial joint torque.

The dynamics of manipulator joints are highly coupled, as can be seen from equation (1). The local control law (2) for each joint compensates, to a large extent, for the static and dynamic cross-couplings that exist between the manipulator joints. For fast simultaneous motion of all joints, the inter-joint coupling effects can be significant and may cause instability under the decentralized adaptive control scheme (2)-(6). In such cases, the controller adaptation laws are modified slightly in order to achieve robust stability in the presence of the unmodeled cross-coupling effects. A popular approach is the "σ-modification" method [3], which yields the adaptation laws

$$f_i(t) = f_i(0) + \delta_i \int_0^t r_i(t)dt + \rho_i r_i(t) - \sigma_i \int_0^t f_i(t)dt \tag{9}$$

$$k_{pi}(t) = k_{pi}(0) + \alpha_i \int_0^t r_i(t)e_i(t)dt + \beta_i r_i(t)e_i(t) - \sigma_i \int_0^t k_{pi}(t)dt \tag{10}$$

$$k_{vi}(t) = k_{vi}(0) + \gamma_i \int_0^t r_i(t)\dot{e}_i(t)dt + \lambda_i r_i(t)\dot{e}_i(t) - \sigma_i \int_0^t k_{vi}(t)dt \tag{11}$$

where σ_i is a positive scalar design parameter. The σ-modified adaptation laws produce a non-zero residual tracking-error of $0(\sqrt{\sigma_i})$ but guarantee stability in the presence of inter-joint couplings.

It is seen that the controller adaptation laws (9)-(11) and the control action (2) are based entirely on the observed manipulator performance through $\theta(t)$ and $\theta_d(t)$ rather than on the manipulator dynamic model (1). As a consequence, the knowledge of either the complex dynamic model and the parameter values of the manipulator or the inertial parameters of the payload is *not* required in the control law formulation. Thus, the adaptive controllers can cope with uncertainties or variations in the manipulator or the payload parameters. This is a highly desirable feature in practical applications, where some dynamic effects such as friction can not be modeled accurately and the payload mass can vary substantially. Furthermore, the proposed decentralized control scheme is extremely fast computationally, since the controller gains are generated on-line in real time by simple adaptation laws, and hence the control action can be evaluated very rapidly. Due to its simplicity and decentralized structure, the proposed scheme can be implemented on parallel processors for distributed concurrent computing with high sampling rates, yielding improved dynamic performance.

3. Description of Testbed Facility

In this section, we describe the robotic testbed facility at the Jet Propulsion Laboratory.

The testbed facility at the JPL Robotics Research Laboratory consists of a six-jointed Unimation PUMA 560 robot and controller, and a DEC MicroVAX II computer, as shown in the functional diagram of Figure 2. The major components of the Unimation controller are the LSI 11/73 microcomputer, six 6503 microprocessor boards (one per joint), and serial/parallel interface cards. The MicroVAX II hosts the RCCL (Robot Control "C" Library) software, which was developed by Hayward and Lloyd at Purdue and McGill Universities [4,5]. The original version runs on a DEC VAX 750 computer, and later the software is ported to a DEC MicroVAX II running UNIX 4.3 BSD operating system. The organization of the robot software is reflected in the control hierarchy diagram illustrated in Figure 2. The complete software resides on two different pieces of computing hardware. A MicroVAX computer, which plays the supervisory role, hosts a two-level software written in the "C" language. The lower level, called Robot Control Interface (RCI), provides the programmer a facility to write real-time control procedures. It serves as a substrate to the higher level (RCCL), which gets its collective name from the robot software. The higher level consists of routines to specify a robot trajectory in Cartesian coordinates. The second piece of hardware is the Unimation controller hosting an LSI 11/73 processor on which an I/O control program called "moper" executes to monitor communication between the 6503 joint microprocessors and the RCI control level.

At the lowest level of the hierarchy, robot servoing is achieved by the 6503 joint microprocessors. The processor has two different operational modes: position and current. In position mode, the 6503 processor accepts a position setpoint from the LSI 11/73 and servos to the desired position by executing a control code written in the assembly language that is stored in ROM (this servo code was developed by the Unimation Corporation). In this mode, the control task, triggered by a hardware clock, executes approximately at the sampling rate of 1 KHz. In current mode, each input is interpreted as current (or torque) by the 6503 processor and is simply converted into an analog value to be forwarded directly to the power drive electronics. This mode makes possible the implementation of any joint control law (e.g. force control and adaptive control) on a remote computer that can interface with the LSI 11/73. In order to implement the adaptive control algorithm, the joint PID servos provided by the resident Unimation code are in effect disconnected by selection of the 6503 current mode, and current inputs are supplied directly from the MicroVAX for driving the robot.

At the intermediate level, the LSI 11/73 executes the "moper" communication monitor program that transfers data and commands back and forth between the LSI 11/73 and the MicroVAX II host computer. The interface between the two processors is a DRV11 high-speed parallel link for control communication. The moper program synchronizes the operation of the entire system as described in the following. A hardware clock located in the Unimation controller constantly interrupts the LSI 11/73, and at each interrupt, the moper collects data relating to the robot state, such as joint positions, currents, and arm status, and transfers it to the MicroVAX via an interrupt. The MicroVAX receives this data and immediately sends position or current values that have been computed in the previous cycle by the control code to the LSI 11/73 for execution. Available system sampling time is in increments of 7 milliseconds, ranging from 7 to 56 milliseconds (18—143 Hz). The sampling time of 28 milliseconds is best suited for the 6503 position mode and is set as default. In our adaptive control implementation, the lowest sampling time of 7 milliseconds (143 Hz) is chosen to obtain the best control performance.

On the MicroVAX, two programs run concurrently: the foreground planning level (RCCL) and the background control level (RCI). The planning level executes in the foreground in the sense that it interacts with the user and performs high-level computations as well as communicating with the control level. It has access to standard I/O resources such as files, devices, and system calls. At this level, the programmer can specify the task in Cartesian coordinate system in terms of homogeneous transformations and define a robot end-effector trajectory with a series of via points. Essentially the planning level consists of task sequencing, motion planning and queueing, and modeling of the world in terms of homogeneous transformations.

RCI (Robot Control Interface) level executes a series of control routines during each sampling period to interface in real-time with the robot. In practice, the control level communicates with the planning level only through global external variables (i.e., shared memory). Both levels can communicate with the robot through predefined global variables: "how," that contains information describing the state of the arm, and "chg," that is used to control the arm. These variables are used for robot control and are usually accessed at the control level. To meet the constraints imposed by the sampling time in the range of milliseconds, the control level is executed in the UNIX kernel mode at the highest priority, which effectively locks out all hardware interrupts. Once initiated, it quickly loads the memory context of the robot control code, performs I/O with moper, and executes the control code and supporting RCI interface routines. Since the control level is in kernel mode, it cannot access the usual system calls or I/O facilities.

The RCCL software was originally designed to control the robot in the 6503 position mode (i.e., with the control action provided at the 6503 joint processor level at 1 KHz sampling rate). This way the user can concentrate mostly on the programming aspects of performing a task rather than the real-time control issues. For this purpose, by default, a set of standard real-time control routines is provided to perform functions such as trajectory generation, error checking, event synchronization, kinematic computations, and coordination with the planning level. In order to implement a different control scheme, however, the user selects the 6503 current mode to drive the robot, which in turn necessitates writing control procedures essentially to replace the nominal control functions. In addition, the user must consider meeting his newly imposed sampling time requirement; more often he wants to lower the sampling time to control the robot more effectively. As a consequence, smaller and more compact control code must be written to meet the real-time constraint. For example, to implement our adaptive controller in the sampling rate of 7 milliseconds (lowest rate available in RCCL), the need for a trajectory generator is met by writing a simple cycloidal generator that provides smooth series of setpoints without added features such as real-time trajectory modification. In addition, the adaptive control algorithm for computing the desired motor currents from the observed joint positions is coded in its most numerically efficient form.

4. Experimentation with a PUMA 560 Robot

In this section, the theoretical results outlined in Section 2 are applied to a six-jointed PUMA 560 industrial robot in the testbed facility described in Section 3.

To test and evaluate the control scheme of Section 2, the adaptive controllers are implemented on all six joints of the PUMA 560 robot. The dynamic model for a typical i^{th} joint of PUMA can be written as

$$m_{ii}(\theta)\ddot{\theta}_i + \sum_{\substack{j=1 \\ \neq i}}^{6} m_{ij}\ddot{\theta}_j + n_i(\theta,\dot{\theta}) + g_i(\theta) + h_i(\dot{\theta}_i) = T_i \tag{12}$$

where $\theta = [\theta_1, \ldots, \theta_6]$ and the terms in equation (12) are highly complicated nonlinear functions of θ and $\dot{\theta}$ as given in [6]. From equation (12), it is seen that the effective inertia $m_{ii}(\theta)$, the gravity loading $g_i(\theta)$, and the Coriolis/centrifugal torque $n_i(\theta,\dot{\theta})$ seen at each joint are nonlinear functions of *all* six joint angles; i.e., the robot configuration and speed. Furthermore, there are inertial couplings between joint motions, as indicated by $m_{ij}\ddot{\theta}_j$ terms, with the coupling factors dependent on the robot configuration. In the adaptive control implementation, the i^{th} joint is controlled independently by the local feedback law

$$T_i(t) = f_i(t) + k_{pi}(t)e_i(t) + k_{vi}(t)\dot{e}_i(t) \tag{13}$$

where $e_i(t) = \theta_{di}(t) - \theta_i(t)$ is the position tracking-error, $\theta_{di}(t)$ is the reference trajectory, and $[f_i, k_{pi}, k_{vi}]$ are the auxiliary signal, position and velocity feedback gains for the i^{th} joint, respectively. It is seen that although the joint dynamics (12) are coupled, the proposed control scheme (13) is decentralized, i.e., the control torque T_i does not depend on the joint angle θ_j for $j \neq i$.

In the experiment on adaptive control of PUMA, the sampling period is chosen as the smallest possible value $T_s = 7$ milliseconds (i.e., sampling frequency $f_s = 143$ Hz), since the on-line computations involved in the adaptive control law (13) are a few simple arithmetic operations. The adaptation gains in equations (3)-(6) are selected after a few trial-and-errors as *

$$\begin{gathered} w_{p1} = 15 \quad , \quad w_{v1} = 10 \quad , \quad w_{p2} = 40 \quad , \quad w_{v2} = 20 \quad , \quad w_{p3} = 12 \quad , \quad w_{v3} = 4 \\ w_{p4} = 3 \quad , \quad w_{v4} = 2 \quad , \quad w_{p5} = 3 \quad , \quad w_{v5} = 2 \quad , \quad w_{p6} = 3 \quad , \quad w_{v6} = 2 \\ \text{All joints}: \delta = 30, \alpha = 100, \gamma = 800, \rho = \beta = \lambda = \sigma = 0 \end{gathered} \tag{14}$$

The initial values of all controller gains are chosen as zero, i.e. $k_{pi}(0) = k_{vi}(0) = 0$ for $i = 1, \ldots, 6$. The initial values of the auxiliary signals are chosen as

$$\begin{aligned} f_2(0) &= 12\text{sgn}[\theta_{d2}(\tau) - \theta_2(0)] + 1.02 \sin\theta_2(0) \\ &\quad - 8.4 \sin[\theta_2(0) + \theta_3(0)] - 37.2 \cos\theta_2(0) \qquad \text{Nt.meter} \\ f_3(0) &= 2\text{sgn}[\theta_{d3}(\tau) - \theta_3(0)] + 0.25 \cos[\theta_2(0) + \theta_3(0)] \\ &\quad - 8.4 \sin[\theta_2(0) + \theta_3(0)] \qquad \text{Nt.meter} \\ f_1(0) &= f_4(0) = f_5(0) = f_6(0) = 0 \end{aligned} \tag{15}$$

In the above expressions, the first term is chosen empirically to overcome the large stiction (static friction) present in the joints, and the remaining terms are used to compensate for the initial gravity loading [6]. It is important to note that friction and gravity compensations are *not* used separately in addition to the adaptive controllers, and are used merely as the initial conditions of the auxiliary signals in order to improve the initial responses of the joint angles. Furthermore, no information about the PUMA dynamic model or parameter values is used for implementation of the control scheme.

In the experiment, the PUMA arm is initially at the "zero" position $\theta(0) = [0, 0, 0, 0, 0, 0]$ with the upper arm horizontal and the forearm vertical, forming configuration. All the six joint angles are then commanded to change simultaneously from the zero positions to the goal positions $\theta_d(3) = [60°, -60°, 60°, 60°, -60°, -60°]$ in three seconds, and the desired trajectories $\theta_{di}(t)$ are synthesized by a cycloidal trajectory generator software in RCCL. While the robot is in motion, the readings of the joint encoders at each sampling instant are recorded directly from the robot, converted into degrees and stored in a data file. Figures 3(i)-(vi) show the desired and actual trajectories of all six PUMA joint angles. It is seen that each joint angle $\theta_i(t)$ tracks the desired trajectory $\theta_{di}(t)$ very closely despite inter-joint couplings. The experimental results demonstrate that adaptive independent joint control of the PUMA robot is feasible, in spite of the static and dynamic couplings between the joints.

* The unit of angle in the control program is "radian," and hence the numerical values of the adaptation gains are large.

5. Conclusions

A decentralized direct adaptive control scheme has been experimentally validated on a PUMA 560 robot, where each robot joint is controlled independently by a simple local feedback controller at a high sampling rate. This avoids the computational burden of a centralized controller, which results in a slower sampling rate and hence degrades the robot performance. The experimental results demonstrate that accurate trajectory tracking is achieved by a simple control algorithm, without any knowledge of the complex PUMA dynamic model.

Adaptive control is particularly useful in applications (such as in space) where the manipulator has long light-weight arms and handles payloads of unknown and heavy weights. In such cases, the dynamics of the manipulator is dominated by the inertial terms due to the payload. The controller adaptation can then compensate for such terms and provide a stable and consistent performance under gross payload variations.

Finally, it is important to note that the rate of sampling, f_s, has a central role in the performance of any digital control system. In general, the value of f_s is dictated largely by the amount of on-line computations that need to be performed during each sampling period in order to calculate the required control action. The simplicity of the control scheme proposed in this paper allows joint servo loops to be implemented with a high sampling rate, yielding improved tracking performance.

6. 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.

7. References

1. H. Seraji: "A new approach to adaptive control of manipulators," ASME Journ. Dynamic Systems, Measurement and Control, 1987, 109(3), pp. 193-202.
2. H. Seraji: "Adaptive independent joint control of manipulators: Theory and experiment," Proc. IEEE Intern. Conf. on Robotics and Automation, Philadelphia, 1988, Vol. 2, pp. 854-861.
3. P.A. Ioannou: "Decentralized adaptive control of interconnected systems," IEEE Trans. Auto. Control, 1986, Vol. AC-31, No. 4, pp. 291-298.
4. V. Hayward and J. Lloyd: "RCCL User's Guide," CVaRL/McGill University, Canada, April 1984.
5. J. Lloyd: "Implementation of a Robot Control Development Environment," Master's Thesis, Electrical Engineering Department, McGill University, Canada, 1985.
6. B. Armstrong, O. Khatib, and J. Burdick: "The explicit dynamic model and inertial parameters of the PUMA 560 arm," Proc. IEEE Intern. Conf. on Robotics and Automation, San Francisco, 1986, pp. 510-518.

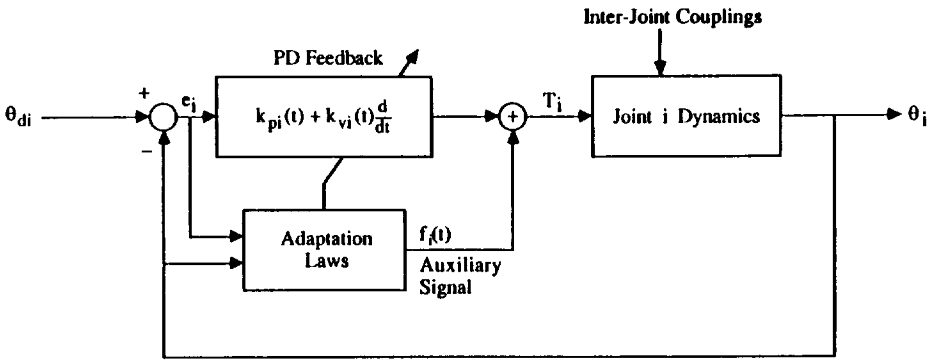


Figure 1. Decentralized Adaptive Control Scheme for Manipulator Joint i

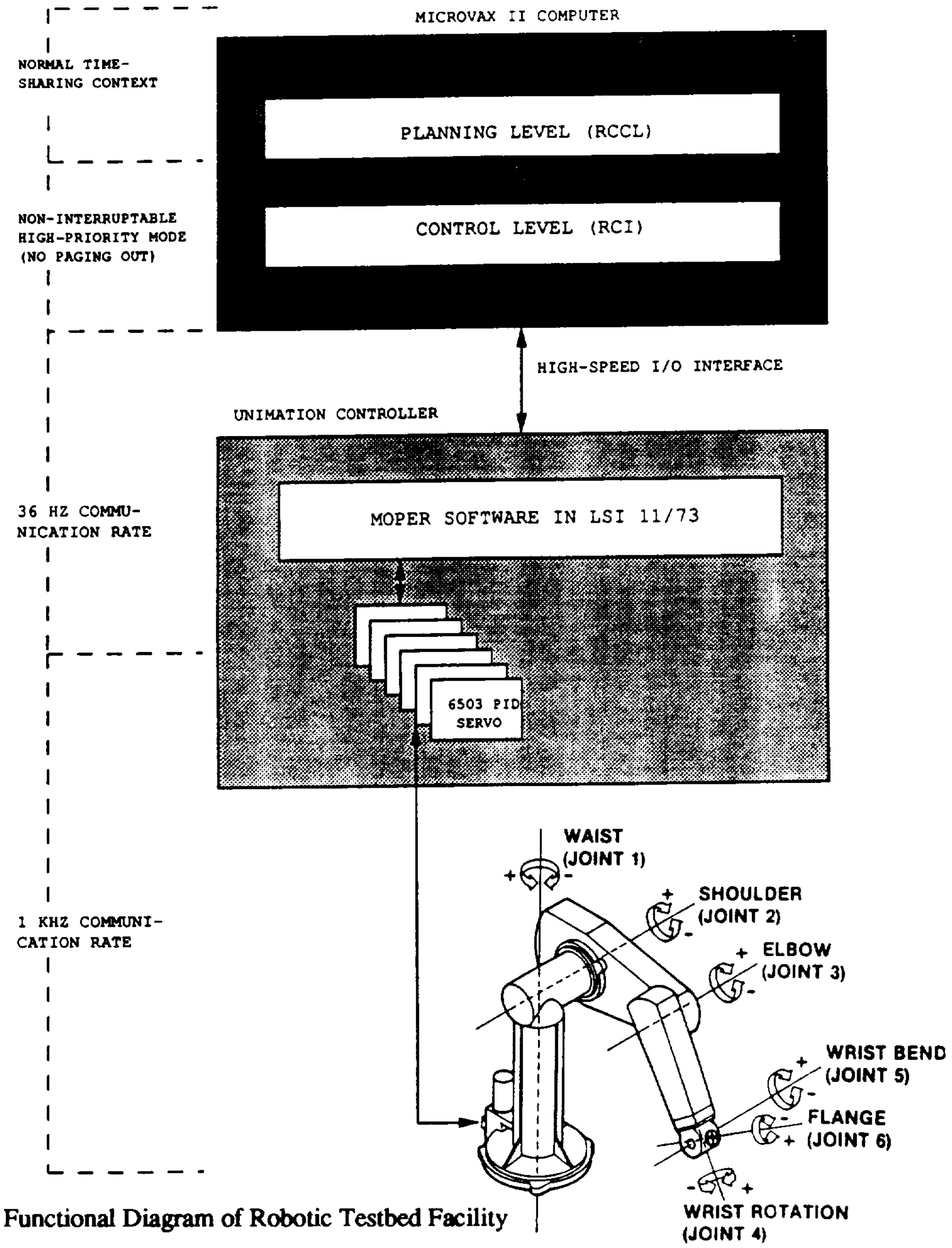


Figure 2. Functional Diagram of Robotic Testbed Facility

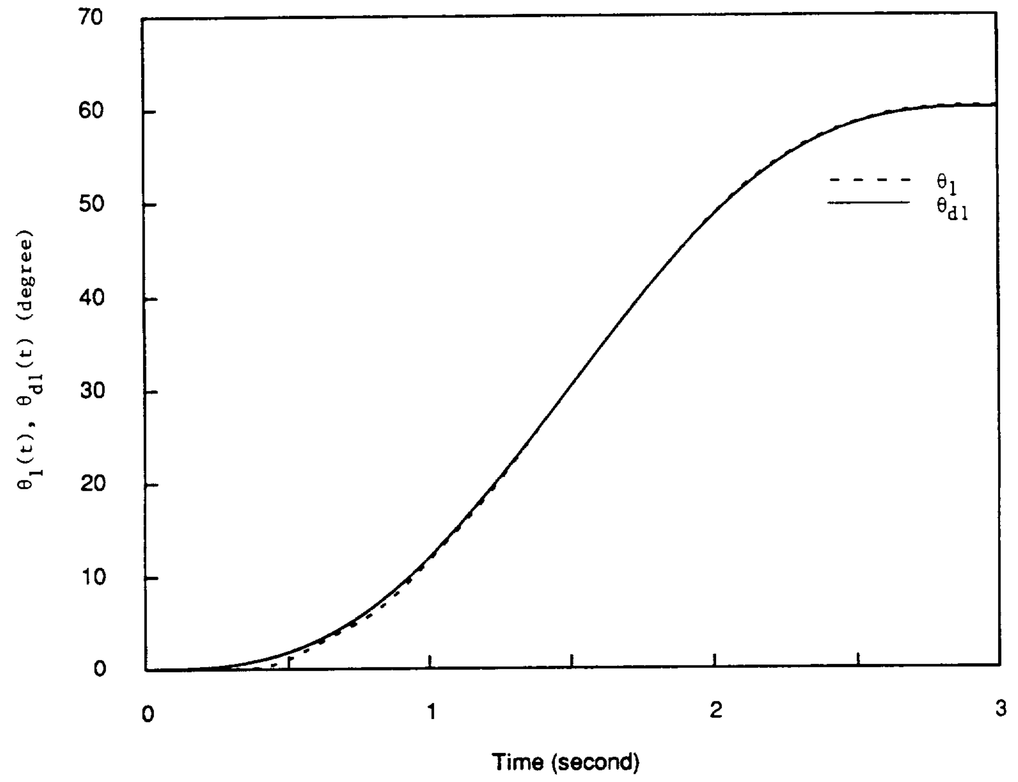


Figure 3(i). Response of Waist Angle θ_1 under Adaptive Controller

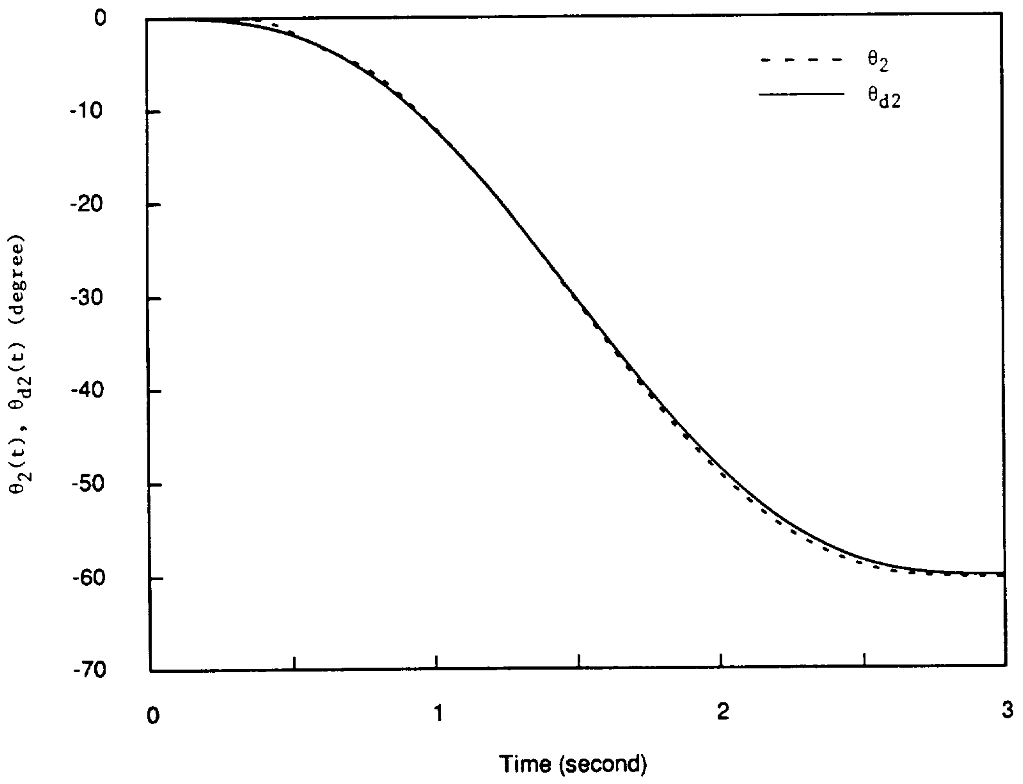


Figure 3(ii). Response of Shoulder Angle θ_2 under Adaptive Controller

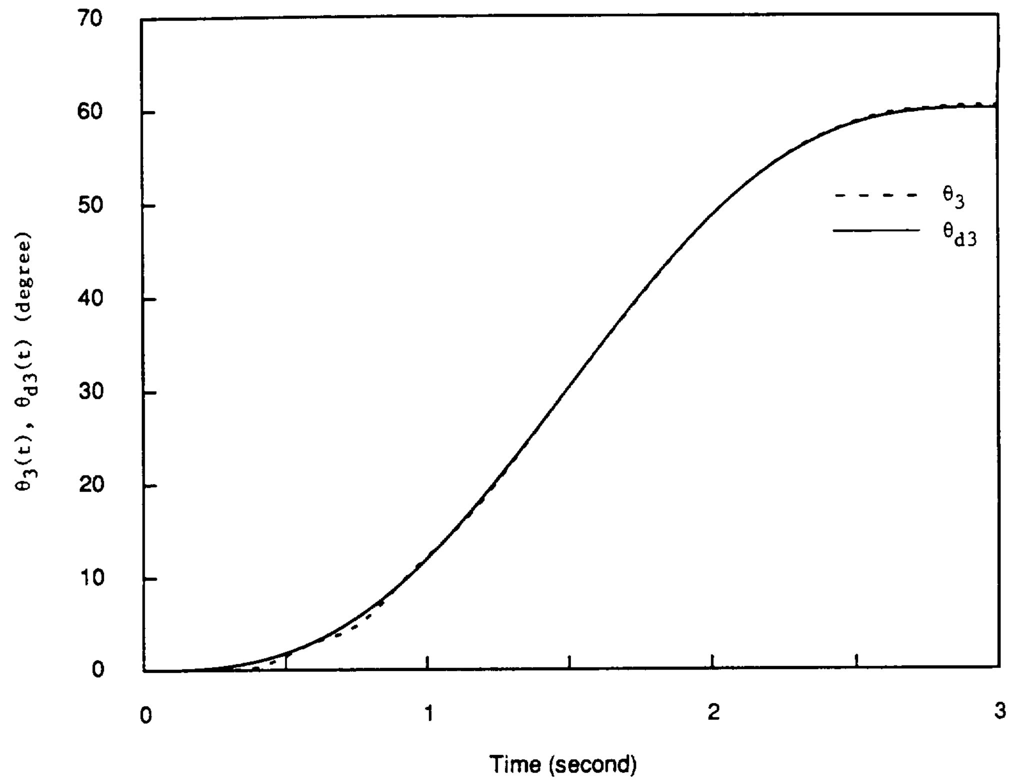


Figure 3(iii). Response of Elbow Angle θ_3 under Adaptive Controller

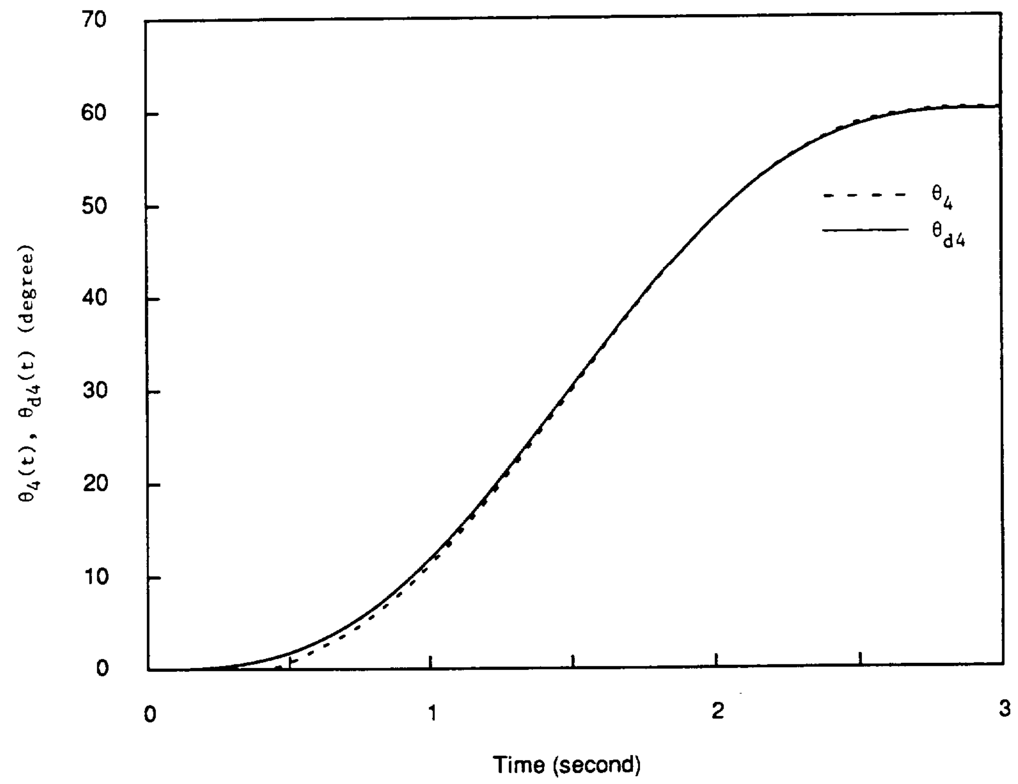


Figure 3(iv). Response of Wrist Angle θ_4 under Adaptive Controller

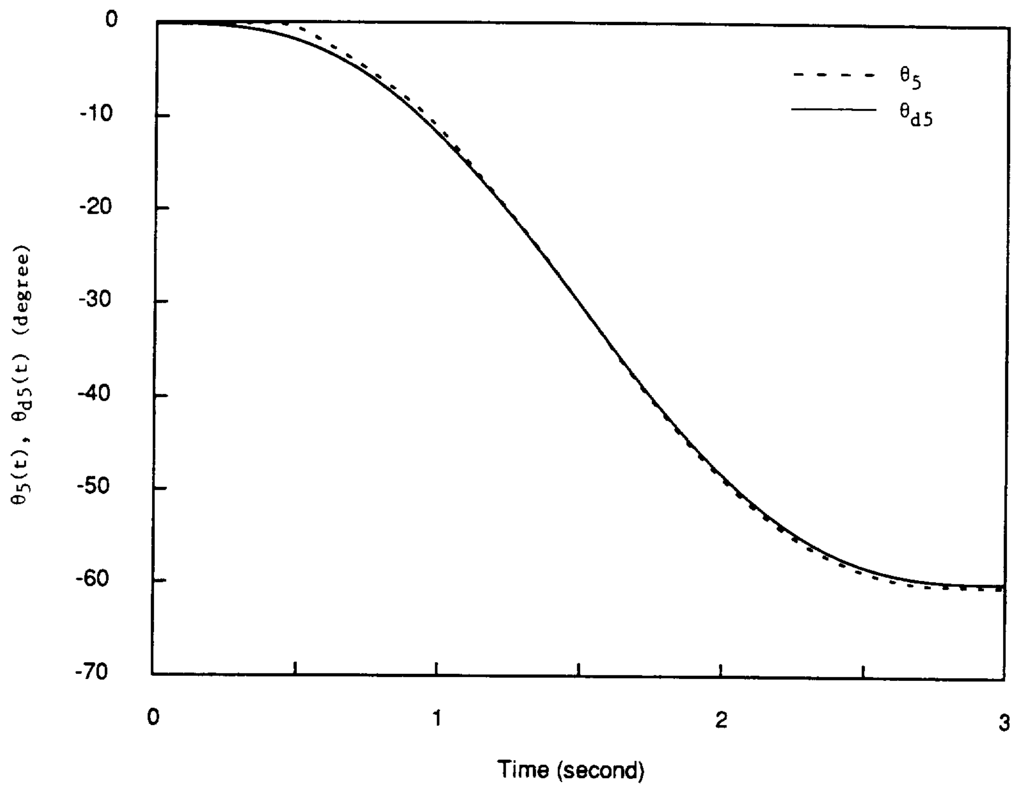


Figure 3(v). Response of Wrist Angle θ_5 under Adaptive Controller

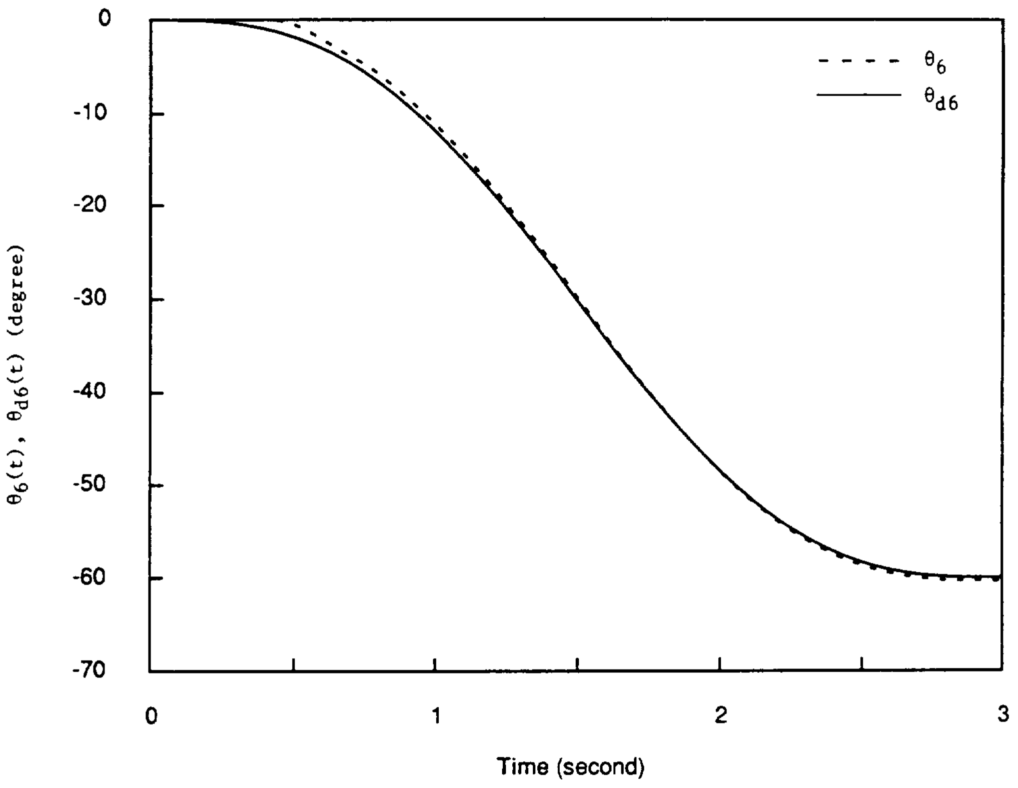


Figure 3(vi). Response of Wrist Angle θ_6 under Adaptive Controller

N90-29883

Model Based Manipulator Control

Lyman J. Petrosky
Westinghouse Advanced Energy Systems Division
Madison, PA, 15663

Irving J. Oppenheim
Departments of Architecture and Civil Engineering
Carnegie Mellon University
Pittsburgh, PA, 15213

1. INTRODUCTION

This research effort has its origins in an experimental study[1] of dynamically stable manipulation. The interest in dynamically stable systems was driven by the objective of high vertical reach, for which human balance was the inspiration, and the objective of planning inertially favorable trajectories for force and payload demands, for which human (animal) efficiency was also the general inspiration. A double inverted pendulum system was constructed as the experimental system for this mission, and the research effort led to activities in non-linear control methods, in trajectory planning (still to be completed), and in the use of *model based control*. The findings from that last task form the main emphasis of this paper. Sections 2, 3 and 4 herein are drawn in large part from a recent workshop paper [5] paper; we then discuss in sections 5 and 6 two general areas by which this work is pertinent to space tele/robotics.

The design of a control system for manipulators is a formidable task due to the complexity of the nonlinear coupled dynamics. The goal is the calculation of actuator torques which will cause the manipulator to follow any desired trajectory. In a broad sense, two basic categories of control design are found in the literature. The first contains the robust control methods in which the control is able to overpower the system's nonlinear coupled dynamics. The second contains the *model-based* control (MBC) methods in which many of the system nonlinearities are calculated using a systems dynamic model and the nonlinear system forces are then canceled by actuation forces. Recent advances in computational hardware have made it possible to evaluate in real time the equations of motion of robotic manipulators. Khosla [1]was the first to demonstrate the feasibility of real time MBC using an inexpensive computer system for control of a six degree of freedom manipulator, the CMU Direct Drive Arm II. The requirements for applying MBC can be satisfied for many manipulators of practical interest to space applications. Basically, the system must be amenable to mathematical modeling, and the mathematical model and the control law must be evaluated in real time.

[1]The project was sponsored by the Department of Energy, Advanced Reactor and Nuclear System Technology Support, Program NE-85-001, under contract DE-AC02-85NE37947, *Dynamic Stability for Robot Vertical Reach and Payload.* We are indebted to Clint Bastin of DOE for his particular interest and support, and are most grateful to Westinghouse AES for their cooperation in accommodating Mr. Petrosky's residence at Carnegie-Mellon. We further wish to acknowledge the contributions of Professors I. Shimoyama and J. Bielak, and graduate student Eric Hoffman. *Disclaimer: The view, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official U. S. Department of Energy or Carnegie-Mellon University position, policy or decision, unless designated by other documentation.*

2. CONTROL APPROACH

2.1. Computed-Torque Control

Computed-torque [2] control is a *model-based* control scheme which strives to use the complete dynamic model of a manipulator to achieve dynamic decoupling of all the joints using nonlinear feedback. The dynamic model of the manipulator is described by the system equations of motion which can be derived from Lagrangian mechanics:

$$\sum_{j=1}^{N} D_{ij}\,\ddot{q}_j + \sum_{j=1}^{N}\sum_{k=1}^{N} C_{jk}(i)\,\dot{q}_j\,\dot{q}_k + g_i = \tau_i \qquad for\ i = 1, \ldots, N. \tag{1}$$

where the q are the joint coordinates. The τ_i are the externally applied joint actuation torques/forces. The inertial D_{ij}, centrifugal and Coriolis $C_{jk}(i)$, and gravitational g_i coefficients of the closed-form dynamic robot model in Equation 1 are functions of the instantaneous joint positions q_i and the constant kinematic, dynamic and gravity manipulator parameters. The kinetic energy gives rise to the inertial and centrifugal and Coriolis torques/forces, while the potential energy leads to the gravitational torques/forces. Actuator dynamics can be incorporated in the dynamic robot model by additions to the Lagrangian energy function.

The *Computed-torque* algorithm begins with a calculation of the required torque to be applied to each of the joints (in vector notation):

$$\begin{aligned} \tau &= \tilde{\mathbf{D}}\mathbf{u} + \tilde{\mathbf{H}} + \tilde{\mathbf{g}} \\ \tilde{H}_i &\equiv \dot{\mathbf{q}}^T\,\tilde{\mathbf{C}}(i)\,\dot{\mathbf{q}} \end{aligned} \tag{2}$$

where **u** is the commanded joint accelerations. The "~" indicates that these matrices are calculated from the system model based on estimated system parameters. The resulting dynamic equations for the closed-loop system are:

$$\ddot{\mathbf{q}} = \mathbf{u} - \mathbf{D}^{-1}\{[\mathbf{D} - \tilde{\mathbf{D}}]\mathbf{u} + [\mathbf{H} - \tilde{\mathbf{H}}] + [\mathbf{g} - \tilde{\mathbf{g}}]\} \tag{3}$$

If the system dynamic parameters are known exactly, then $\tilde{\mathbf{D}} = \mathbf{D}$, $\tilde{\mathbf{H}} = \mathbf{H}$, and $\tilde{\mathbf{g}} = \mathbf{g}$, then the closed loop system is described by:

$$\ddot{\mathbf{q}} = \mathbf{u} \tag{4}$$

which is the equation for a set of decoupled second order integrators. This completes the formulation of the modeling and feed forward decoupling functions of the algorithm.

The feedback control law for the commanded joint acceleration u_i is formulated to incorporate the error feedback signal and the reference signal. After decoupling, each joint acts as a second order integrator, therefore the control law is given the form:

$$u_i = \ddot{q}_{id} - 2\zeta\omega(\dot{q}_i - \dot{q}_{id}) - \omega^2(q_i - q_{id}) \qquad (5)$$

which causes each joint to act as a second order damped oscillator with natural frequency ω and damping ratio ζ. The form of the equation causes the joint to track the desired joint values q_{id}, $\dot{q}_{id}$, and $\ddot{q}_{id}$.

The computed-torque control defined above is based on the assumptions that the system model is accurate and that all joints are actuated. In our experimental effort the dynamic parameters of the manipulator were manually measured to provide an accurate system model. We assume in all simulations that the dynamic model is accurate. Our experimental system, the double inverted pendulum depicted in Figure 1, does not conform to the second assumption (in that all joints are not actuated) and therefore the algorithm was extended as described in the next section.

2.2. Application to Balancing

Consider first the simplest balancing problem, the planar single inverted pendulum. Balancing is a fourth order control problem with a single input and in the context of this article is equivalent to controlling two manipulator joints with a single actuator. The presumption of the computed torque algorithm, that all joints are actuated, does not apply. However, a suitable control law was found by Petrosky [3]; that method, called *hierarchical partitioning*, is directly applicable to the balancing problem, is robust, and can be integrated with MBC. The balancing problem is partitioned into two second order subsystems, tilt and position. The input signal, base position acceleration, has a component driving the tilt subsystem. The tilt in turn is considered as the input to the position subsystem. This cascaded pair of subsystems is then controlled by a pair of control laws of the form of Equation 5 with the tilt subsystem given a faster time constant. By removing internal variables from the cascaded system, a nonlinear balancing control law is obtained for the manipulator base position variable. This is combined with the computed-torque control for the actuated joints to complete the manipulator control algorithm.

2.3. Determination of Applied Forces

Indirect determination of applied forces (*i.e.* without the use of load sensors) is accomplished by comparison of the manipulator mathematical model and the observed manipulator behavior. A simple example of this is the algorithm for payload determination for the balancing manipulator. Payload estimation can be performed for a balancing manipulator on-line in real time. Consider the equation of motion for pivoting about the base of the dynamically balanced manipulator:

$$\tau_i = \sum_{j=1}^{N} D_{ij}\ddot{q}_j + \sum_{j=1}^{N}\sum_{k=1}^{N} C_{jk}(i)\,\dot{q}_j\dot{q}_k + g_i \qquad \textit{for } i = \textit{base rotation} \quad (6)$$

The base joint of a balanced manipulator is not actuated, therefore $\tau_i = 0$. However, if the payload value is incorrect, then this equation will evaluate to a non-zero value of τ_i when the observed values of the joint variables are entered. The difference indicates the value of the payload which is given by:

$$\Delta P = -\left(\frac{\partial \tau_i}{\partial P}\right)^{-1} \tau_i \qquad (7)$$

where ΔP is the difference between the actual payload and the current estimated value. Under ideal conditions this equation would yield the correct payload value in a single sample; however, the accuracy of the values for $\ddot{q}_j$ can be exceedingly poor if obtained by double differentiation of position measurements. This was the case in the experimental system, but the problem was overcome by the use of a parameter estimator.

3. EXPERIMENTAL SYSTEM

The experimental manipulator is a planar double inverted pendulum as depicted in Figure 1; the system is presumed to traverse an approximately level surface, and requires constant active balancing motions. In its plane of motion there are three degrees-of-freedom: translation of the base position, q_1, rotation of the lower arm with respect to the vertical, q_2, and rotation of the upper arm with respect to the lower arm, q_3. It is q_2 which is not directly controlled in this system. The manipulator has a servo driven wheeled base, a hinged connection (free rotation) to the lower arm section, an elbow joint which is servo driven, the upper arm, and an electro-magnet pickup at the tip. It is constructed primarily of aluminum and has a total weight of 13 kg; the tip of the manipulator can reach a height of 1.8 meters in an erect stance.

The wheeled base and the elbow joint are driven by Aerotek servos rated at 1.3 N-m peak torque. The elbow joint has a chain reduction ratio of 57.6:1 and the drive wheels have a chain reduction of 4.8:1. The chain reduced servo arrangement was chosen over direct drive to save weight, and over gear-reduced or harmonic drive to mitigate costly damage in the event of a severe floor collision.

The sensors utilized for manipulator control are:

- Inclination RVDT - a rotary differential transformer measures the angle between the floor surface (via a feeler) and the lower arm.
- Motor Encoders - each servo has an optical encoder of 500 counts per revolution which runs a hardware counter read by the parallel interface board.

The control system hardware consists of a Motorola M68000 based single board computer as the master CPU, a Marinco Array Processor Board (APB), an Analog to Digital Converter (ADC) 32 channel input board, a Digital to Analog (DAC) 4 channel output board, a 96 line Parallel Input/Output (PIO) Interface board, and a terminal. The Marinco APB, with an instruction cycle of 125 *ns*, is used to perform the calculation intensive operations required to implement MBC. The board has fixed point multiplier and addition hardware which are used for floating point operations. The floating point addition or multiplication routines execute in approximately 1 μs. Negation requires 125 *ns*. Computation of the sine/cosine pair requires 15 μs. Additional routines perform data type conversion and other functions required to format sensor data.

Manipulator trajectory calculations are handled by the M68000 CPU on a time sharing basis. In operation a timer interrupts the CPU at each sampling interval. The CPU copies the sensor data to the APB memory and initiates APB execution. The APB formats the data, does scaling operations, performs the trigonometric functions, and then calculates the inverse dynamics. The formatted output data is ready in less than 0.5 *ms*. Data needed for control are returned to the CPU, which outputs them to the DAC's. Cycle time is sufficiently fast for the control algorithm and dynamic model to be evaluated at a sampling frequency in excess of 1000 Hz. However, 100 Hz appeared to be more than adequate for the experimental system.

4. EXPERIMENTAL RESULTS

The experimental manipulator was fully reliable in maintaining balance for long periods while performing a variety of tasks. The base moves approximately ± 3 mm to maintain balance and the tilt varies by ± 0.0063 radian. This motion does not indicate a flaw in the balancing algorithm, but rather the motion results from being at the limit of tilt resolution of the RVDT sensor used with the floor feeler; the RVDT signal variation corresponds to the magnitude of a single digital count. Because the base dimension of the experimental system is zero, it is physically impossible for the manipulator to balance without some minor motions.

The manipulator proved very resistant to upset; its recovery ability appears to exceed that of a human under similar magnitude disturbances. Figure 2 records the transient response of the manipulator to a severe impact applied 0.3 seconds into the record. The manipulator moved forward in order to balance, translating 0.75 meters, and then quickly returned to its original base position. Rotation through a range of 0.25 radians is recorded for the lower arm. The manipulator was also extremely forgiving (compliant) of collision. The manipulator would bounce lightly off an obstacle and come to rest simply leaning against it. When commanded to back away from the obstacle, the manipulator would resume balancing as soon as contact was broken.

Figure 3 records the transient response of the manipulator under the application and removal of a payload at the tip, with the upper arm near the horizontal; the payload was 0.811 kg, and the tip position was offset horizontally by 0.8 meters from base position. The time histories of q_1 and q_2 reflect the payload applied at 5 seconds, removed at 13 seconds, and applied again at 19 seconds. The presence, magnitude, and location of the payload was determined indirectly as discussed in section 2.3; the information was used to adapt the control scheme by updating the sytem model. Figure 3 shows the trace of this payload estimation process, which is noteworthy for its accuracy. In this manner it was possible to adapt to large payloads, demonstrated experimentally with ease up to 3.2 kg, or 25% of the total system weight. A payload estimation record from ongoing balancing in the absence of payload (not shown) demonstrates a typical noise level of ± 26 gm, which is only 0.2% of the system mass.

Another experiment demonstrated the successful development and control of lateral force through the motion of the system masses. A chain connected the manipulator to a heavy mass on a rough table, and the manipulator was used to pull the mass against the force of friction through some target distance. The manipulator developed a lateral force through the movement of its mass center to a point behind its wheel axis; the system them maintained control through the motion ensuing as the lateral force exceeded the friction force, in much the same way that a human would pull a heavy weight accross a floor. Another experiment demonstrated the pickup of the 3.2 kg payload from the floor to an overhead height of 1.8m. The vertical force required to raise the mass was generated by placing the manipulator system masses at great eccentricity to the payload; this effect, and subsequent control of the system, closely resembled a weightlifter's *clean-and-jerk*.

5. APPLICATION OF MODEL BASED CONTROL TO SYSTEMS WITH FLEXIBLE LINKS

MBC has potential space tele-robotic application for manipulators with flexible links. In principle, information available from the on line system model can be utilized to adjust controller gains to the current manipulator configuration. We observe (but do not discuss further) that joint-flexible manipulators, in which flexibility effects are confined to revolute joints, would be controllable in all configurations. We direct our attention at manipulators characterized by linear elastic link bending effects, and presume in our discussion that lumped parameter modelling can apply. Such manipulators are difficult to control because there are many additional system degrees-of-freedom (the "deformation variables" introduced in modelling the flexibility effects) and because some flexural modes may be poorly coupled in the inputs. In this section we develop specialized equations of motion and discuss the potential for the application of MBC using modal decomposition.

Flexible manipulators undergo quasi-periodic oscillations due to elastic deformation. These vibrations develop in response to actuated motions and disturbances. Small vibrations of this type normally decompose into orthogonal modes. This holds true for a manipulator only if it is not undergoing gross motion. As a result of the nonlinear manipulator dynamics, oscillations in the structure exhibit cross terms which negate modal orthogonality. This effect can be deduced from the equations of motion. Equation 1, the manipulator equations of motion, can be expanded for a manipulator with flexibility; deleting summation symbols for purposes of clarity, it becomes:

$$\tau_i = D_{ij}\ddot{q}_j + C_{jk}(i)\,\dot{q}_j\dot{q}_k + g_i + K_{ij}q_j \qquad \textit{for } i = 1,\ldots,M. \tag{8}$$

where q also includes required deformation degrees of freedom. Consider a decomposition of the q into a vibration component, δq, plus an equilibrium trajectory component, q. Substituting into Equation 8 and segregating the terms for vibration yields:

$$\begin{aligned}\tau_i + \delta\tau_i = {} & \\ & D_{ij}\ddot{q}_j + C_{jk}(i)\,\dot{q}_j\dot{q}_k + g_i + K_{ij}q_j \\ & + D_{ij}\,\delta\ddot{q}_j + 2C_{jk}(i)\,\dot{q}_j\,\delta\dot{q}_k + C_{jk}(i)\,\delta\dot{q}_j\,\delta\dot{q}_k \\ & + K_{ij}\,\delta q_j \qquad \textit{for } i = 1,\ldots,M.\end{aligned} \tag{9}$$

Because the equilibrium trajectory portion of the equation by definition satisfies Equation 8, the remaining terms for the vibration component yield the governing equation of motion for vibrations:

$$\begin{aligned}\delta\tau_i = {} & D_{ij}\,\delta\ddot{q}_j + 2C_{jk}(i)\,\dot{q}_j\,\delta\dot{q}_k \\ & + C_{jk}(i)\,\delta\dot{q}_j\,\delta\dot{q}_k + K_{ij}\,\delta q_j \\ & \qquad \textit{for } i = 1,\ldots,M.\end{aligned} \tag{10}$$

We see that velocity cross terms exist if the manipulator is in motion. If the amplitude of vibration is small the equation linearizes. The D, C, and K are constant and the $C_{jk}(i)\,\delta\dot{q}_j\,\delta\dot{q}_k$ term is ignored. The free vibration (i.e. $\delta\tau_i = 0$) portion of the manipulator motion forms a linear dynamic system:

$$0 = D_{ij}\,\delta\ddot{q}_j + B_{ij}\,\delta\dot{q}_k + K_{ij}\,\delta q_j \qquad \textit{for } i = 1, \ldots, M. \quad (11)$$

$$\textit{where } B_{ij} = \sum_{j=1}^{M} 2\, C_{jk}(i)\, \dot{q}_j$$

The B matrix appears in the role of a damping term, however due to its form no vibrational energy is lost, only exchanged among the modes. If the manipulator is stationary, $\dot{q}_j = 0$, then it behaves like an undamped multiple degree of freedom elastic structure. To achieve stable control it is necessary to use the system inputs to add damping to the vibration equation.

In principle, such a manipulator would remain amenable to mathematical modeling. The computational burden of calculating a manipulator stiffness matrix is low compared to calculating the dynamic parameters, except that for the link flexible manipulator the entire system has more degrees of freedom. However, a valid control scheme which utilizes the model of the flexible manipulator is significantly more complex that for its rigid counterpart. Nonlinear decoupling such as achieved by the Computed-Torque method *cannot* be anticipated in most cases for flexible systems. Considering next modern control theory methods for pole placement in Multiple-Input Multiple-Output (MIMO) systems, since the system model in MBC can be continuously updated for the current manipulator configuration, MIMO pole placement control would have available at all times a model to linearize for control feedback gain calculation. However, preliminary evaluation of MIMO pole placement indicates that the methods involve numerous matrix inversions, and would not be suited to online implementation using current microprocessors.

An alternate control scheme to discuss is modal decomposition. Presumably, free vibration mode shapes can be calculated based on the system model. Once calculated, modal decomposition of the system dynamic equations and determination of input gains would be straightforward. It appears feasible to determine control gains by specifying the required modal damping matrix and calculating the resulting required actuator inputs. The calculation burden for this control scheme is high because of the eigenvector calculation, but appears to be within the capability of current technology. If proven feasible, this method represents an excellent solution to the flexible manipulator problem.

6. TRAJECTORY PLANNING FOR UTILIZATION OF INERTIAL EFFECTS

Trajectory planning utilizing inertial effects promises efficiencies of great significance to space applications. The payload experiments described at the conclusion of section 4 exemplify these efficiencies at an informal level. More broadly, in this category of trajectory planning one would find minimum energy paths, minimum energy-density paths, minimum time paths, minimum torque paths, and so on. One would also find paths which represent favorable matches between actuator capacities and task requirements. This work is the doctoral research objective of the first author [4] and is currently under investigation. Its pursuit is supported by the MBC capabilities described herein, but is not a direct extension of them. Therefore this brief section is less prescriptive and more descriptive than the discussion of MBC for flexible manipulator control.

Optimal control can solve certain of these problems, such as the minimum time path, and mathematical approaches exist which (under restrictions) can solve others, such as the geodesic for the minimum energy path. Our interest is in approximate approaches which can be framed more generally, and which can be calculated on-line, though not necessarily in real-time. A number of approaches are being studied, including evaluation of different abstractions for use as objective functions, and various mappings of inertial space from which approximate paths might be determined.

7. CONCLUSIONS

The feasibility of utilizing real time Model Based Control (MBC) for robotic manipulators has been demonstrated. The experimental results demonstrate the effectiveness of the control approach, balancing, and of the payload estimation/adaptation algorithm developed for this effort. The mathematical modeling of dynamics inherent in MBC permit the control system to perform functions that are impossible with conventional non-model based methods. These capabilities include:

- Stable control at all speeds of operation;
- Operations requiring dynamic stability such as balancing;
- Detection and monitoring of applied forces without the use of load sensors;
- Manipulator "safing" via detection of abnormal loads;
- Control of flexible manipulators.

This work directly demonstrates the first two capabilities and indicates the feasibility of the additional capabilities. The control of flexible manipulators is a particularly important potential application because this problem has proven very difficult to solve. This technology also supports our work on trajectory planning for favorable utilization of inertial forces.

8. REFERENCES

[1] Khosla, P.K.
Real-Time Control and Identification of Direct-Drive Manipulators.
PhD thesis, Carnegie-Mellon University, 1986.

[2] Markiewicz, B.R.
Analysis of the Computed-Torque Drive Method and Comparison with the Conventional Position Servo for a Computer-Controlled Manipulator.
Technical Memorandum 33-601, Jet Propulsion Laboratory, Pasadena, CA, March, 1973.

[3] Petrosky, L.J.
Problem Substructuring Applied to the Design of a Controller for the Stabilization of a Double Inverted Pendulum.
Master's thesis, Carnegie-Mellon University, 1986.

[4] Petrosky, L.J.
Utilization of Dynamic Forces in Robotic Manipulation (Dissertation Proposal).
PhD thesis, Carnegie-Mellon University, 1987.

[5] Petrosky, L.J., and Oppenheim, I.J.
Application of Model Based Control to Robotic Manipulators.
In S. Griffin (editor), *Second Annual Workshop on Space Operations Automation and Robotics (SOAR '88)*, pages 487-493. NASA Conference Publication 3019, July, 1988.

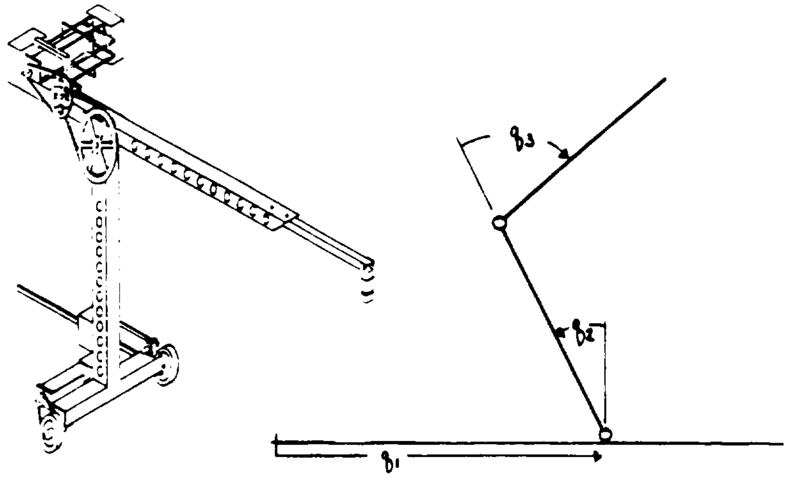


Figure 1. Experimental Manipulator

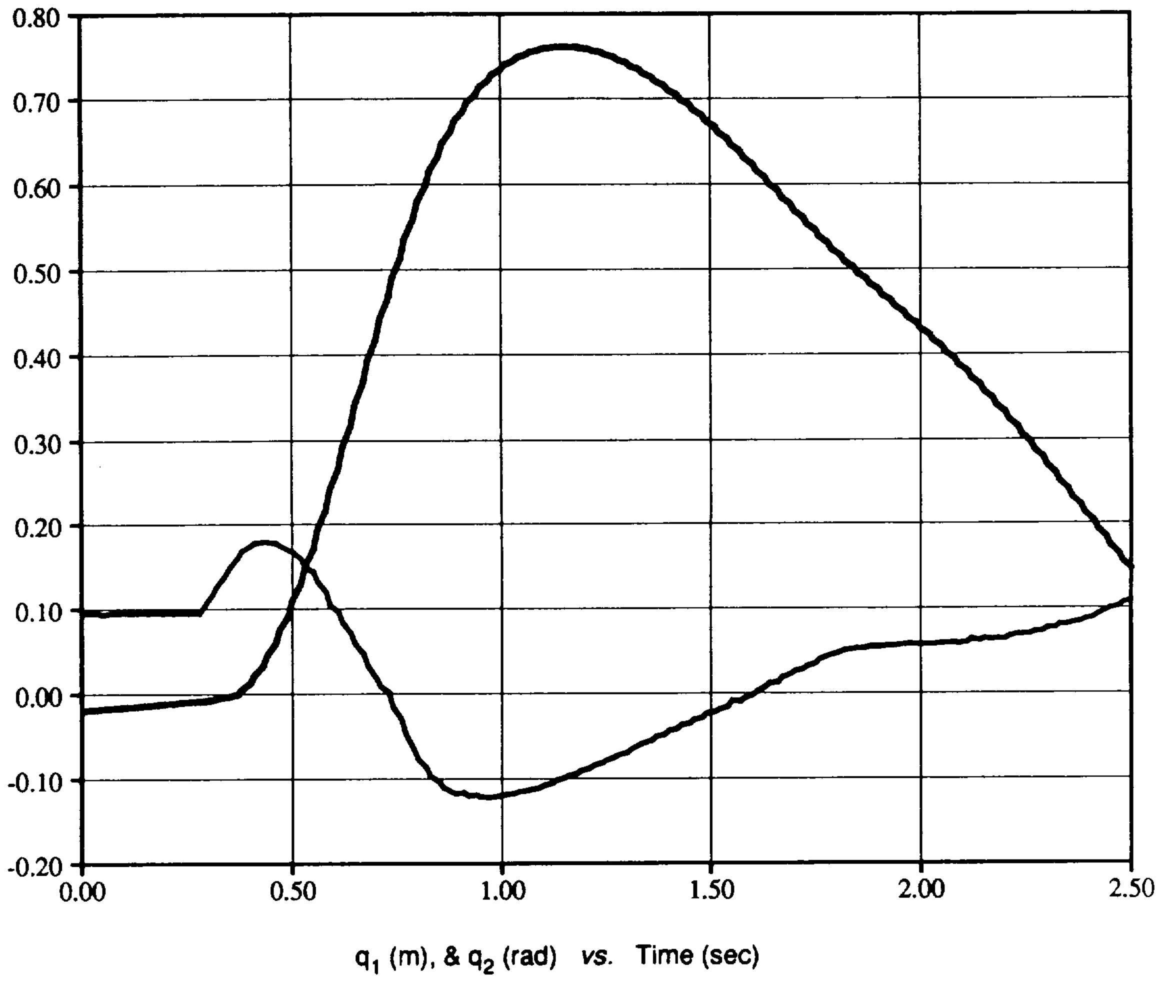
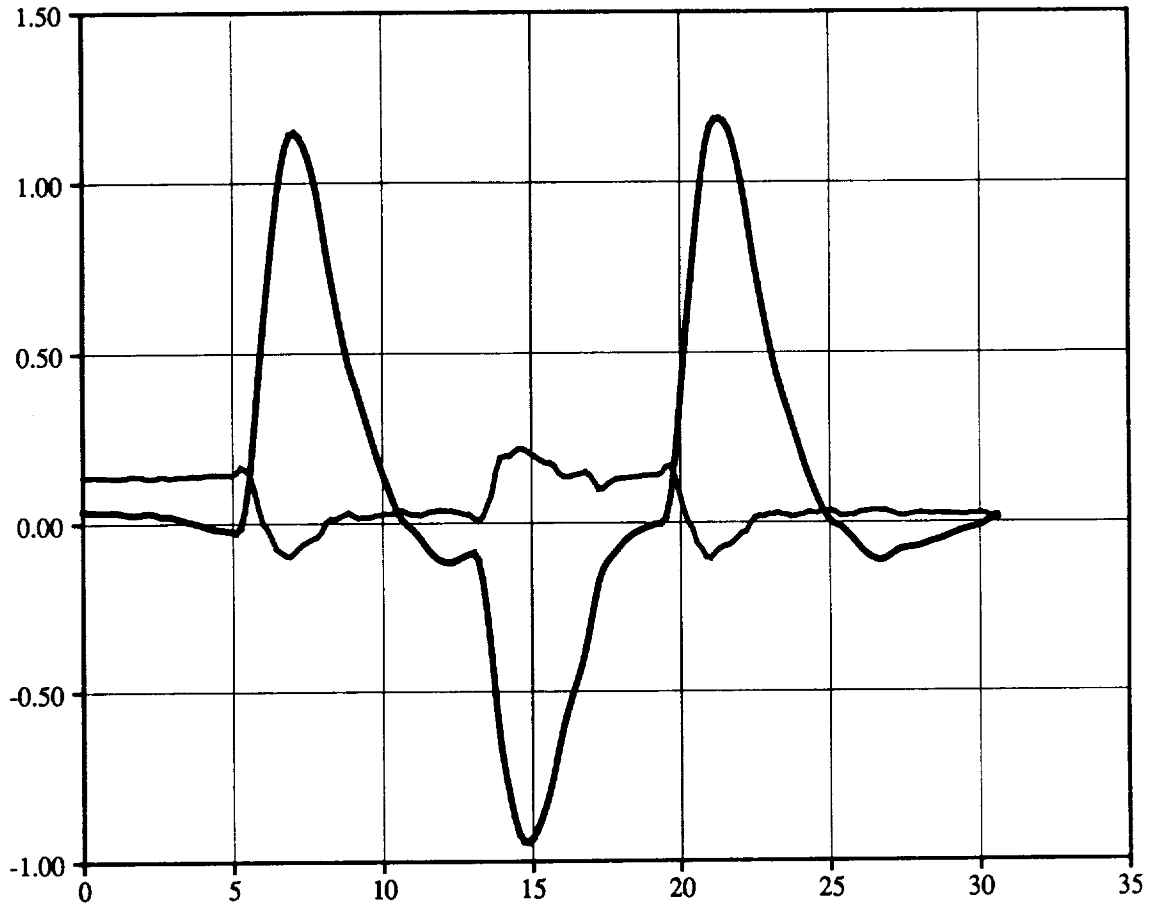
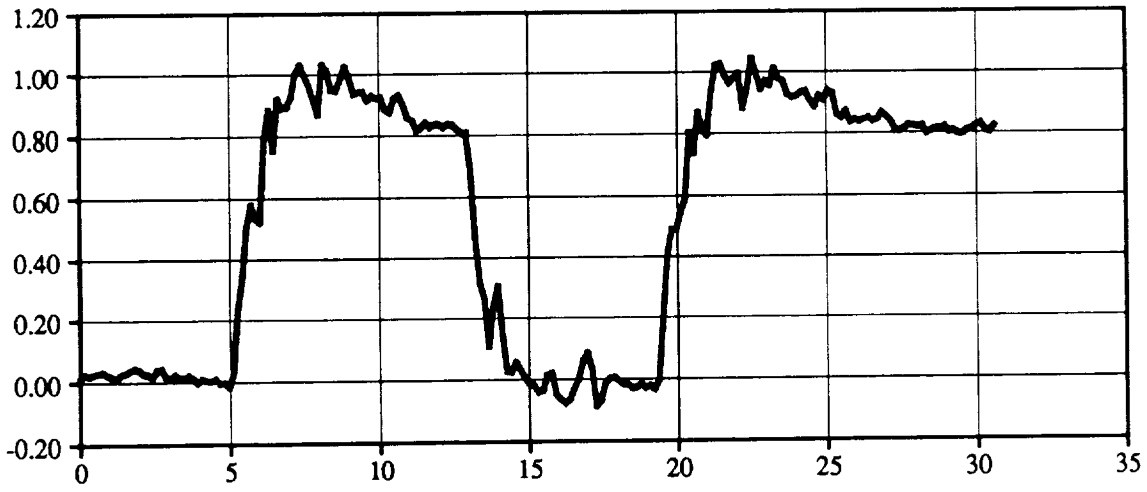


Figure 2. Manipulator Response to Impact



q_1 (m), & q_2 (rad) *vs.* Time (sec)



Est. Payload (kg) *vs.* Time (sec)

Figure 3. Manipulator Response to Application of Payload at Tip

N90-29834

DISCRETE-TIME ADAPTIVE CONTROL OF ROBOT MANIPULATORS

M. Tarokh
California Space Institute
University of California, San Diego
La Jolla, CA 92093

Abstract

A discrete-time model reference adaptive control scheme is developed for trajectory tracking of robot manipulators. Hyperstability theory is utilized to derive the adaptation laws for the controller gain matrices. It is shown that asymptotic trajectory tracking is achieved despite gross robot parameter variation and uncertainties. The method offers considerable design flexibility and enables the designer to improve the performance of the control system by adjusting free design parameters. The discrete-time adaptation algorithm is extremely simple and is therefore suitable for real-time implementation.

1. Introduction

It is recognized that adaptive schemes are effective means of robot control due to their ability to cope with the highly nonlinear, coupled and time-varying characteristics of robots. This is specially true in the case of direct drive robots and light weight manipulators where inertia changes and gravity effects are significant. Research efforts on adaptive control of manipulators have been concentrated on developing continuous-time control schemes [e.g. 1-9]. In practice however, robots are controlled by digital computers on discrete-time basis. Digital implementation of a solution based on continuous-time formulation can result in degradation of performance and the closed-loop system can even become unstable, especially when the sampling time is not small. Even if the sampling time could be made sufficiently small, digital implementation of a discrete-time adaptive scheme is more direct and straightforward.

In this paper, we develop a discrete-time model reference adaptive control scheme for trajectory tracking of robot manipulators. The present approach differs from the previously published results [e.g. 10-12] in that the discrete-time adaptive control is developed on the basis of a general coupled robot model, without linearizing the model or assuming negligible interactions among robot joints. Furthermore, instead of the conventional Lyapunov approach, hyperstability theory is utilized to obtain the adaptation laws. The use of hyperstability theory is more appealing than the Lyapunov approach since it is better suited to discrete-time systems and also offers more flexibility in design by providing additional free design parameters. These parameters can be adjusted by the designer to improve the response. Finally, the proposed discrete-time adaptive control algorithm is extremely simple and computationally fast, and is therefore suitable for real time digital control of robot manipulators.

2. Discrete-Time Robot Model

The equation of motion of an n-joint robot manipulator carrying a payload of mass m can be written as [4,9]

$$B_2(\theta,\dot{\theta},m)\,\ddot{\theta} + B_1(\theta,\dot{\theta},m)\,\dot{\theta} + B_0(\theta,\dot{\theta},m)\,\theta(t) = u(t) \tag{1}$$

where $\theta(t)$ and $u(t)$ are the $n\times 1$ joint angle and joint torque vectors respectively, and $B_2(.)$, $B_1(.)$ and $B_0(.)$ are $n\times n$ matrices whose elements are complex nonlinear functions of $\theta(t)$,$\dot{\theta}(t)$ and $m(t)$. Since $\theta(t)$,$\dot{\theta}(t)$ and $m(t)$ are functions of time, (1) can be expressed as

$$B_2(t)\,\ddot{\theta}(t) + B_1(t)\,\dot{\theta}(t) + B_0(t)\,\theta(t) = u(t) \tag{2}$$

where $B_2(t) \equiv B_2(\theta,\dot{\theta},m)$, $B_1(t) \equiv B_1(\theta,\dot{\theta},m)$ and $B_0(t) \equiv B_0(\theta,\dot{\theta},m)$ are $n\times n$ time- varying robot matrices.

Suppose that the robot is controlled by a digital controller. The inputs to the controller are the reference trajectory represented by the $n\times 1$ vector $\theta_r(k)$ and the actual joint angle vector $\theta(k)$, where $\theta_r(k)$ and $\theta(k)$ are obtained by sampling $\theta_r(t)$ and $\theta(t)$ at equally spaced time intervals T. The output of the digital controller is the vector $u(k)$, and is passed through a hold circuit to obtain the continuous-time signal $u(t)$ where $u(t)$ is constant over the time interval $(k-1)T\le t\le kT$. In order to obtain the equation relating $\theta(k)$ and $u(k)$, we must discretize the robot model (2). A simple method of discretization is by using the approximations

$$\dot{\theta}(t) \approx \frac{1}{T}\Big[\theta(k)-\theta(k-1)\Big] \;\; ; \;\; \ddot{\theta}(t) = \frac{d}{dt}\dot{\theta}(t) \approx \frac{1}{T^2}\Big[\theta(k)-2\theta(k-1)+\theta(k-2)\Big] \tag{3}$$

Substituting (3) into (2), we obtain

$$A_2(k,T)\,\theta(k-2) + A_1(k,T)\,\theta(k-1) + A_0(k,T)\,\theta(k) = u(k) \tag{4}$$

where $A_2(k,T) = \dfrac{B_2(k)}{T^2}$, $A_1(k,T) = \dfrac{B_1(k)}{T} - \dfrac{2B_2(k)}{T^2}$ and $A_0(k,T) = B_0(k) - \dfrac{B_1(k)}{T} + \dfrac{B_2(k)}{T^2}$ are $n\times n$ matrices, and $B_2(k)$,$B_1(k)$, $B_0(k)$ are the values of $B_2(t)$, $B_1(t)$, $B_0(t)$ respectively, evaluated at time $t{=}kT$. Note that $A_2(k,T)$ is a symmetric positive definite (SPD) matrix since $B_2(t)$ is always SPD [13]. Equation (4) is an accurate discrete-time representation of (1) provided that T is sufficiently small so that (3) can be used.

For exact discretization, we must find the response $\theta(t)$ of the continuous model (2) at time $t{=}kT$ and equate it with the response $\theta(k)$ of the discrete model (4), [14]. This will ensure that the two models describe the same robot motion at the sampling times $t{=}kT$, $k{=}0,1,2,\ldots$. Although this procedure provides structural information about the robot discrete-time model, it is extremely complex and will not be pursued here.

In the analysis to follow, we assume that the equation of the robot with a sampler in its output and a hold circuit in its input can be described by the discrete-time model (4), where $A_0(k,T)$ is invertible and the robot matrices are unknown. Since the sampling period is constant , we drop it for convenience and write (4) as

$$A_2(k)\,\theta(k-2) + A_1(k)\,\theta(k-1) + A_0(k)\,\theta(k) = u(k) \tag{5}$$

3. Adaptive Control Scheme

In this section, we describe a method for the design of discrete-time adaptive controllers for the robot model (5) such that the robot joint angle vector $\theta(k)$ tracks the reference trajectory vector $\theta_r(k)$ despite variations in the payload and unknown robot model parameters.

Let the $n \times 1$ joint angle error vector be defined as

$$\theta_e(k) = \theta_r(k) - \theta(k) \tag{6}$$

Substituting (6) into (5), we obtain the equation of the joint angle error as

$$\theta_e(k) = A_0^{-1}\Big[-u(k)-A_1(k)\,\theta_e(k-1)-A_2(k)\,\theta_e(k-2)+A_0(k)\,\theta_r(k)+A_1(k)\,\theta_r(k-1)+A_2(k)\,\theta_r(k-2)\Big] \tag{7}$$

Equation (7) suggests that in order to completely influence the joint angle error, we require a control law of the general form

$$u(k) = P_1(k)\,\theta_e(k-1)+P_2(k)\,\theta_e(k-2)+Q_0(k)\,\theta_r(k)+Q_1(k)\,\theta_r(k-1)+Q_2(k)\,\theta_r(k-2) \tag{8}$$

where $P_1(k), P_2(k)$ are time-varying feedback matrices acting on the joint angle error, and $Q_0(k), Q_1(k), Q_2(k)$ are time-varying feedforward matrices acting on the reference trajectory, all to be determined. Note that the discrete-time control law (8) is analogous to the continuous-time control law using position-velocity feedback and position-velocity-acceleration feedforward [9].
Substituting (8) into (7), we obtain the joint angle error equation for the closed-loop system

$$\begin{aligned}\theta_e(k) + A_0^{-1}\Big[P_1(k)+A_1(k)\Big]\,\theta_e(k-1) + A_0^{-1}\Big[P_2(k)+A_2(k)\Big]\,\theta_e(k-2) \\ = A_0^{-1}\Big[A_0(k)-Q_0(k)\Big]\,\theta_r(k) + A_0^{-1}\Big[A_1(k)-Q_1(k)\Big]\,\theta_r(k-1) + A_0^{-1}\Big[A_2(k)-Q_2(k)\Big]\,\theta_r(k-2)\end{aligned} \tag{9}$$

Suppose that the desired performance of the manipulator is represented by

$$\theta_{em}(k) + C_1\theta_{em}(k-1) + C_2\theta_{em}(k-2) = 0 \tag{10}$$

where $\theta_{em}(k)$ is the $n \times 1$ joint angle error vector of the reference model and C_1, C_2 are constant $n \times n$ matrices chosen such that joint angle errors are decoupled and decay with time. In the model reference adaptive control terminology [15], equations (9) and (10) describe the adjustable system and the reference model, respectively. For decoupling of the joint errors , we choose $C_1 = diag\{c_{1i}\}$ and $C_2 = diag\{c_{2i}\}$, $i=1,2,\ldots,n$. In order that the errors decay to zero, the roots λ_{1i} , λ_{2i} of the characteristic polynomial $\Delta(z)$ of the reference model (10) must lie inside the unit circle in the complex z-plane, where

$$\Delta(z) = \left| I_n z^2 + C_1 z + C_2 \right| = \prod_{i=1}^{n} \delta_i(z) \tag{11a}$$

and

$$\delta_i(z) = z^2 + c_{1i} z + c_{2i} = (z+\lambda_{1i})(z+\lambda_{2i}) \tag{11b}$$

Thus the diagonal elements of the matrices C_1 and C_2 are

$$c_{1i} = \lambda_{1i}+\lambda_{2i} \quad ; \quad c_{2i} = \lambda_{1i}\lambda_{2i} \quad , \quad i=1,2,\ldots,n \tag{11c}$$

where $|\lambda_{1i}|<1$, $|\lambda_{2i}|<1$ for the stability of the reference model.
The solution to (10) is

$$\theta_{em}(k) = \Phi_m(k)\,\theta_{em}(0) \tag{12}$$

where $\Phi_m(k)$ is the transition matrix of the reference model (10) and $\theta_{em}(0)$ is the initial value of the reference model. If $\theta_{em}(0)$ is chosen to be zero, $\theta_{em}(k)$ becomes identically equal to zero, i.e. $\theta_{em}(k) \equiv 0$ for all $k \geq 0$, due to the stability of the reference model. The objective is now to devise an adaptation scheme such that the robot joint angle error dynamics $\theta_e(k)$ governed by (9) approaches that of the reference model dynamics (10) in which $\theta_{em}(k) \equiv 0$. In order to achieve this objective, we define the deviation between the ideal and the actual errors as

$$\varepsilon(k) = \theta_{em}(k) - \theta_e(k) \tag{13}$$

Combining (9), (10) and (13), we obtain

$$\varepsilon(k) + C_1\,\varepsilon(k-1) + C_2\,\varepsilon(k-2) + w(k) = 0 \tag{14a}$$

where

$$\begin{aligned} w(k) &= \left[C_1 - A_0^{-1}(A_1(k)+P_1(k))\right]\theta_e(k-1) + \left[C_2 - A_0^{-1}(A_2(k)+P_2(k))\right]\theta_e(k-2) \\ &\quad + A_0^{-1}\left[A_0(k)-Q_0(k)\right]\theta_r(k) + A_0^{-1}\left[A_1(k)-Q_1(k)\right]\theta_r(k-1) + A_0^{-1}\left[A_2(k)-Q_2(k)\right]\theta_r(k-2) \\ &\equiv w_1(k) + w_2(k) + \cdots + w_5(k) \end{aligned} \tag{14b}$$

The adaptation problem is to find the feedback gain matrices $P_1(k), P_2(k)$ and the feedforward gain matrices $Q_0(k), Q_1(k), Q_2(k)$ such that the adaptation error dynamic (14) is stable, i.e. $\varepsilon(k)$ approaches zero asymptotically. If this is achieved, the joint angle error vector $\theta_e(k)$ becomes equal to the reference model error vector $\theta_{em}(k) \equiv 0$, implying that $\theta_e(k)=0$, hence $\theta(k)=\theta_r(k)$ and trajectory tracking occurs.

The state-space representation of (14) is

$$\begin{bmatrix} \varepsilon(k-1) \\ \varepsilon(k) \end{bmatrix} = \begin{bmatrix} 0 & I_n \\ -C_2 & -C_1 \end{bmatrix} \begin{bmatrix} \varepsilon(k-2) \\ \varepsilon(k-1) \end{bmatrix} - \begin{bmatrix} 0 \\ I_n \end{bmatrix} w(k) \tag{15}$$

Now consider the adaptation algorithm

$$v(k) = D \begin{bmatrix} \varepsilon(k-1) \\ \varepsilon(k) \end{bmatrix} \tag{16}$$

$$w(k)=\Psi_1(v,\theta_e)\theta_e(k-1)+\Psi_2(v,\theta_e)\theta_e(k-2)+\Psi_3(v,\theta_r)\theta_r(k)+\Psi_4(v,\theta_r)\theta_r(k-1)+\Psi_5(v,\theta_r)\theta_r(k-2) \tag{17}$$

where $v(k)$ is an $n\times1$ vector, D is a constant $n\times2n$ matrix to be determined, and $\Psi_1(v,\theta_e), \ldots, \Psi_5(v,\theta_r)$ are $n\times n$ matrices, also to be determined. In order to ensure that the adaptation dynamics described by (15)-(17) is stable so that the adaptation error approaches zero asymptotically, we utilize the Popov hyperstability theory. This theory requires that the dynamic equations of the adaptation process be arranged in a feedback configuration. The forward block must contain only linear time-invariant dynamic equations while the feedback block can contain nonlinear time-varying dynamic equations. In the robot control problem under consideration, the forward block has the input $w(k)$, the output $v(k)$ and is described by (15)-(16). The nonlinear feedback block is described by (17).

According to the hyperstability theory, the adaptation algorithm (15)-(17) is stable in the sense that $\lim_{k\to\infty} \begin{bmatrix} \varepsilon(k-1) \\ \varepsilon(k) \end{bmatrix} = 0$ if the following two conditions are satisfied:

Condition 1 : The transfer function matrix of the forward block $H(z) = z\,D\,(zI_{2n} - C)^{-1}B$ is strictly positive real (SPR), where $C = \begin{bmatrix} 0 & I_n \\ -C_2 & -C_1 \end{bmatrix}$ and $B = \begin{bmatrix} 0 \\ I_n \end{bmatrix}$.

Condition 2 : The input-output of the feedback block satisfies the inequality $\sum_{k=0}^{k_1} v^T(k)\, w(k) \geq -\gamma^2$ for all k_1, where γ is an arbitrary finite constant and the superscript T denotes the transposition.

Using proportional plus integral type adaptation for the gain matrices, it is shown in the appendix that the following algorithm that satisfies conditions 1 and 2

$$P_1(k) = P_1(k-1) + \hat{\theta}_e(k)\, \theta_e^T(k-1) E_{1P} + \hat{\theta}_e(k-1)\, \theta_e^T(k-2)[E_{1I} - E_{1P}] \tag{18a}$$

$$P_2(k) = P_2(k-1) + \hat{\theta}_e(k)\, \theta_e^T(k-2) E_{2P} + \hat{\theta}_e(k-1)\, \theta_e^T(k-3)[E_{2I} - E_{2P}] \tag{18b}$$

$$Q_0(k) = Q_0(k-1) + \hat{\theta}_e(k)\, \theta_r^T(k) F_{0P} + \hat{\theta}_e(k-1)\, \theta_r^T(k-1)[F_{0I} - F_{0P}] \tag{18c}$$

$$Q_1(k) = Q_1(k-1) + \hat{\theta}_e(k)\, \theta_r^T(k-1) F_{1P} + \hat{\theta}_e(k-1)\, \theta_r^T(k-2)[F_{1I} - F_{1P}] \tag{18d}$$

$$Q_2(k) = Q_2(k-1) + \hat{\theta}_e(k)\, \theta_r^T(k-2) F_{2P} + \hat{\theta}_e(k-1)\, \theta_r^T(k-3)[F_{2I} - F_{2P}] \tag{18e}$$

and

$$\hat{\theta}_e(k) = R_2\, \theta_e(k-1) + R_3\, \theta_e(k) \tag{18f}$$

where $E_{0P}, E_{0I}, \ldots, F_{2P}$ and F_{2I} are SPD adaptation gain matrices and the subscripts P and I denote proportional and integral parts, respectively. R_2 and R_3 are $n \times n$ diagonal matrices whose diagonal elements r_{2i} and r_{3i} are obtained from

$$r_{2i} = \alpha_i\, \lambda_{1i}\, \lambda_{2i}\, (\lambda_{1i} + \lambda_{2i}) \tag{19a}$$

$$r_{3i} = \alpha_i\, (1 + \lambda_{1i}\, \lambda_{2i}) \qquad i=1,2,\ldots,n \tag{19b}$$

where α_i are positive constants and $\lambda_{1i}, \lambda_{2i}$ are the eigenvalues of the error reference model chosen such that $|\lambda_{1i}|<1, |\lambda_{2i}|<1$, as explained before. Note that the feedback gains depend only on the joint angle error vector, whereas the feedforward gains depend both on the joint angle vector and the reference trajectory vector. A block diagram of the adaptive control scheme is shown in Figure 1.

Equations (8), (18) and (19) constitute the adaptation control algorithm. The SPD matrices $E_{0P}, E_{0I}, \ldots, F_{2P}, F_{2I}$, the positive scalars α_i and the eigenvalues $\lambda_{1i}, \lambda_{2i}$ must be specified by the designer. A simple structure for the above matrices is the diagonal structure. Furthermore, a particularly simple expression for $\hat{\theta}_e(k)$ is obtained if the eigenvalues of the reference model are $\lambda_{1i} = \lambda_{2i} = 0$, $i=1,2,\ldots,n$. This corresponds to the so called "dead-beat control", and in this case (18f) simplies to

$$\hat{\theta}_e(k) = R_3 \theta_e(k) \quad ; \quad R_3 = diag\,\{\alpha_i\} \tag{20}$$

Larger values of the elements of the matrices $E_{0P}, \ldots, F_{2I}$ and the scalars α_i correspond to higher adaptation gains and make the errors decay faster. However, if unmodeled dynamics are present, high adaptation gains can excite unmodeled dynamics, resulting in instability. Thus the design parameters must be selected based on a compromise between speed of adaptation and stability considerations.

It is seen that the adaptive control laws given by (8), (18) and (19) are extremely simple and are suitable for real time control. Furthermore the complex robot dynamics or robot parameters are not required for the generation of control torques. The adaptation algorithm ensures that the closed-loop system remains stable and that trajectory tracking occurs provided the rate of adaptation of controller gains is higher than the rate of change of robot matrices. For example, the matrices of many industrial robots do not change appreciably over time intervals of about ten milliseconds, in which case the controller adaptation time can be a few milliseconds.

5. Conclusions

An adaptive control scheme is developed using a general discrete-time model of robot manipulators. The control scheme utilizes only joint position-velocity measurements and the reference position, and does not require knowledge of the payload or the robot characteristics. The adaptation laws are derived using hyperstability theory which guarantees asymptotic trajectory tracking despite gross robot parameter variations. The controller gains are independent of the robot parameters provided that the gain adaptation is sufficiently fast.

The method offers considerable flexibility in design by providing many free design parameters. These parameters can be adjusted by the designer to improve the response and to increase the speed of adaptation. The discrete-time adaptive control algorithm is extremely simple and computationally fast, and is therefore suitable for real time digital control of robot manipulators. Extensive computer simulation studies using a model of a direct drive manipulator have shown that the discrete-time adaptive scheme performs satisfactorily despite gross payload variations and unknown robot parameters.

References

1. Dubowsky, S. and DesForges, D.T., "The application of model reference adaptive control to robot manipulators", ASME Journal of Dynamics Systems, Measurement and Control, Vol. 101, 193-200, 1979.

2. Takegaki, M., and Arimoto, S., "An adaptive trajectory control of manipulators", Int. Journal of Control, Vol. 34, No. 2, 219-230, 1981.

3. Kim, B.K., and Shin, K.G., "An adaptive model following control of industrial manipulators", IEEE Trans. Aerospace and Electronic Systems, Vol. 19, No. 6, 805-813, 1983.

4. Balestrino, A., DeMaria, G., and Sciavicco, L., "An adaptive model following control for robot manipulators", ASME Journal of Dynamic Systems, Measurement and Control, Vol. 105, 143-151, 1983.

5. Nicosia, S., and Tomei, P., "Model reference adaptive control algorithms for industrial robots", Automatica, Vol. 20, No. 5, 635-644, 1984.

6. Singh, S.N., "Adaptive model following control of nonlinear robotic systems", IEEE Trans. Auto. Control, Vol. 30, No. 11, 1099-1100, 1985.

7. Lim, K.Y., and Eslami, M., "Adaptive controller design for robot manipulators systems using Lyapunov direct method", IEEE Trans. Auro. Control, Vol. 30, No. 12, 1229-1233, 1985.

8. Craig, J.J., Hsu, P. and Sastry, S.S., "Adaptive control of mechanical manipulators", Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, 190-195, 1986.

9. Seraji, H. "A new approach to adaptive control of robot manipulators", ASME Journal of Dynamic Systems, Measurement and Control, Vol. 109, 193-202, 1987.

10. Horowitz, R. and Tomizuka, M. "Discrete-time model reference adaptive control of mechanical manipulator", Computer in Engineering, Vol. 2, Robot and robotics, ASME, 107-112, 1982.

11. Koivo, A.J. and Guo, T.H., "Adaptive linear controller for robotic manipulators", IEEE Trans. Auto. Control, Vol. 28, No. 2, 162-170, 1983.

12. Lee, C.S.G., and Chung, M.J., "An adaptive control strategy for mechanical manipulators", IEEE Trans. Auto. Control, Vol. 29, No. 9, 837-840, 198.

13. Craig, J.J., Robotics- Mechanics and Control, Addison-Wesley, MA, 1986.

14. M. Tarokh, "The effect of discretization on the performance of robot control systems", Proc. Second Int. Symposium on Robotics and Manufacturing, Albuquerque, NM, ASME press, 719-926, 1988.

15. Landau, Y.D., Adaptive Control- Model Reference Approach, Marcel Dekker, N.Y., 1979.

Appendix

In this appendix, we derive the gain adaptation algorithm (18). Consider Condition 1 and write $H(z)$ as

$$H(z) = z\ D\ (zI_{2n} - C)^{-1}B = DB + DC\ (zI_{2n} - C)^{-1}B \tag{21}$$

Now, $H(z)$ is SPR if the exists $2n \times 2n$ symmetric positive definite (SPD) matrices R and M, $n \times n$ matrix K and $2n \times n$ matrix L such that [15, Lemma B.4-2]

$$C^T RC - R = -LL^T - M \tag{22}$$

$$B^T RC + K^T L^T = DC \tag{23}$$

$$K^T K = (DB) + (DB)^T - B^T RB \tag{24}$$

The problem is to choose the matrices R, M, K, L and D to satisfy (22)-(24). Let

$$R = \begin{bmatrix} R_1 & R_2 \\ R_2 & R_3 \end{bmatrix} \quad ; \quad R_1 = diag\{r_{1i}\}\ ,\ R_2 = diag\{r_{2i}\}\ ,\ R_3 = diag\{r_{3i}\}$$

where R_1, R_2 and R_3 are diagonal $n \times n$ matrices whose diagonal elements are $r_{1i} > 0, r_{2i} > 0$ and $r_{3i} > 0$; $i = 1,2,\ldots,n$. This particular structure ensures that R is SPD and simplifies the derivations, as will be seen. Substituting R and B in (24), we have

$$K^T K = R_3^T = R_3$$

or $K = diag\ \{\sqrt{r_{3i}}\}$ and thus the matrix K is found to satisfy (24). Next we choose $D = (R_2 \quad R_3)$ and substitute for D, B, R, C and K in (23) to obtain $L = 0$. Thus L is also found and (23) is satisfied. Now we consider (22), and in order to obtain explicit relationship between the elements of R and the given matrices C and M, we select the following structures

$$M = \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \quad ; \quad M_1 = diag\{m_{1i}\}\ ,\ M_2 = diag\{m_{2i}\}\ \ ; m_{1i}\ , m_{2i} > 0$$

$$C = \begin{bmatrix} 0 & I_n \\ -C_2 & -C_1 \end{bmatrix} \quad ; \ C_2 = diag\{c_{2i}\}\ \ ,\ \ C_1 = diag\{c_{1i}\}$$

Substituting for R ,C and M in (22) and solving, we obtain

$$r_{1i} = m_{1i} + \frac{m_i}{c_i} c_{2i}^2 (1 + c_{2i})\ \ ,\ \ r_{2i} = \frac{m_i}{c_i} c_{2i} c_{1i}\ \ ,\ \ r_{3i} = \frac{m_i}{c_i}(1 + c_{2i}) \tag{25a}$$

where

$$m_i = (m_{1i} + m_{2i}) > 0\ \ ;\ \ c_i = (1 - c_{2i})\left[(1 + c_{2i})^2 - c_{1i}^2\right] \tag{25b}$$

The characteristic polynomial of the reference model (10) is

$$\Delta(z) = \left| z^2 I_n + C_1 z + C_2 \right| = \prod_{i=1}^{n} \delta_i(z) \tag{26a}$$

where

$$\delta_i(z) = z^2 + c_{1i} z + c_{2i} = (z+\lambda_{1i})(z+\lambda_{2i}) \tag{26b}$$

Since the reference model is stable, i.e. $|\lambda_{1i}| < 1$, $|\lambda_{2i}| < 1$, we must have

$$|\delta_i(0)|<1 \qquad \delta_i(1)>0 \qquad \delta_i(-1)>0$$

or

$$|c_{0i}|<1 , \quad (1+c_{1i}+c_{2i})>0 , \quad (1-c_{1i}-c_{2i})>0 \tag{27}$$

Inequalities (27) imply that c_i in (25b) is positive. Let $\alpha_i = \dfrac{m_{1i}+m_{2i}}{c_i}$ $, i=1,2,...,n$ where α_i are positive numbers, then using (25) and (26b) we have

$$r_{1i} = m_{1i}+\alpha_i(\lambda_{1i}\lambda_{2i})^2(1+\lambda_{1i}\lambda_{2i}) \tag{28a}$$

$$r_{2i} = \alpha_i\lambda_{1i}\lambda_{2i}(\lambda_{1i}+\lambda_{2i}) \tag{28b}$$

$$r_{3i} = \alpha_i(1+\lambda_{1i}\lambda_{2i}) \tag{28c}$$

Note that the acquired r_{1i} $,r_{2i}$ and r_{3i} are positive, and thus R_1, R_2, R_3 and consequently R are all SPD. We conclude that Condition 1 is satisfied by choosing D in the adaptation algorithm (16) as $D = (R_2 \quad R_3)$ where the elements of R_2 and R_3 are given by (28b) and (28c), respectively.

In order to satisfy Condition 2, we select the matrices $\Psi_1(v,\theta_e)$, . . . , $\Psi_5(v,\theta_r)$ in (17 according to the following proportional plus integral (summation) adaptation law

$$\Psi_1(v,\theta_e) = C_1 - A_0^{-1}\left[A_1(k)+P_1(k)\right] = Gv(k)\theta_e^T(k-1)E_{1P} + G\sum_{l=0}^{k-1} v(l)\theta_e^T(l-1)E_{1I} \tag{29a}$$

$$\Psi_2(v,\theta_e) = C_2 - A_0^{-1}\left[A_2(k)+P_2(k)\right] = Gv(k)\theta_e^T(k-2)E_{2P} + G\sum_{l=0}^{k-1} v(l)\theta_e^T(l-2)E_{2I} \tag{29b}$$

$$\Psi_3(v,\theta_r) = A_0^{-1}\left[A_0(k)-Q_0(k)\right] = Gv(k)\theta_r^T(k)F_{0P} + G\sum_{l=0}^{k-1} v(l)\theta_r^T(l)F_{0I} \tag{29c}$$

$$\Psi_4(v,\theta_r) = A_0^{-1}\left[A_1(k)-Q_1(k)\right] = Gv(k)\theta_r^T(k-1)F_{1P} + G\sum_{l=0}^{k-1} v(l)\theta_r^T(l-1)F_{1I} \tag{29d}$$

$$\Psi_5(v,\theta_r) = A_0^{-1}\left[A_2(k)-Q_2(k)\right] = Gv(k)\theta_r^T(k-2)F_{2P} + G\sum_{l=0}^{k-1} v(l)\theta_r^T(l-2)F_{2I} \tag{29e}$$

where G, E_{1P} , E_{1I} , . . . , F_{2P} , F_{2I} are SPD matrices, and the subscripts P and I denote proportional and integral terms, respectively.

Consider the first term in the expression for $w(k)$, i.e. $w_1(k)$ given in (14b). Using (29a), we have

$$\sum_{k=0}^{k_1} v^T(k)w_1(k) = \sum_{k=0}^{k_1}\left[v^T(k)Gv(k)\theta_e^T(k-1)E_{1P}\theta_e(k-1) + v^T(k)G\sum_{l=0}^{k-1} v(l)\theta_e^T(l-1)E_{1I}\theta_e(k-1)\right] \tag{30}$$

It is seen that the proportional term produces two quadratic forms $v^T(k)Gv(k)$ and

$\theta_e(k-1)^T E_{1P} \theta_e(k-1)$ which are both positive for all $k \geq 0$. Similarly, it can be shown [15, Appendix D] that the integral term produces quadratic forms and thus $\sum_{k=0}^{k_1} v^T(k) w_1(k) > 0$. Since $w_2(k), \ldots, w_5(k)$ in (14b) and (17) have structures similar to $w_1(k)$, we have $\sum_{k=0}^{k_1} v^T(k) w_j(k) > 0$, $j=1,2,\ldots,5$. We conclude that Condition 2 is satisfied due to the particular choices in (29).

Let us chose $G = A_0^{-1}$, define the change in the gain matrices due to adaptation at time k as $\Delta P_1(k) = P_1(k) - P_1(k-1), \ldots, \Delta Q_2(k) = Q_2(k) - Q_2(k-1)$, and denote the corresponding changes in the robot matrices by $\Delta A_0(k)$, $\Delta A_1(k)$ and $\Delta A_2(k)$. Then after simplifications, we obtain from (29)

$$\Delta P_1(k) + \Delta A_1(k) - \Delta A_0(k) C_1 = \left[v(k-1)\, \theta_e^T(k-2) - v(k)\, \theta_e^T(k-1) \right] E_{1P} - v(k-1)\, \theta_e^T(k-2) E_{1I} \quad (31a)$$

$$\Delta P_2(k) + \Delta A_2(k) - \Delta A_0(k) C_2 = \left[v(k-1)\, \theta_e^T(k-3) - v(k)\, \theta_e^T(k-2) \right] E_{2P} - v(k-1)\, \theta_e^T(k-3) E_{1I} \quad (31b)$$

$$\Delta Q_0(k) - \Delta A_0(k) = \left[v(k-1)\, \theta_r^T(k-1) - v(k)\, \theta_r^T(k) \right] F_{0P} - v(k-1)\, \theta_r^T(k-1) F_{0I} \quad (31c)$$

$$\Delta Q_1(k) - \Delta A_1(k) = \left[v(k-1)\, \theta_r^T(k-2) - v(k)\, \theta_r^T(k-1) \right] F_{1P} - v(k-1)\, \theta_r^T(k-2) F_{1I} \quad (31d)$$

$$\Delta Q_2(k) - \Delta A_2(k) = \left[v(k-1)\, \theta_r^T(k-3) - v(k)\, \theta_r^T(k-2) \right] F_{2P} - v(k-1)\, \theta_r^T(k-3) F_{2I} \quad (31e)$$

In order to make the controller gain matrices independent of the robot matrices, we assume that the changes in the robot matrices is much smaller than the corresponding changes in the gain matrices due to adaptation, i.e.

$$\Delta P_1(k) \gg \Delta A_1(k) - \Delta A_0(k) C_1, \ldots, \Delta Q_2(k) \gg \Delta A_2(k) \quad (32)$$

This assumption is valid if the adaptation rate is sufficiently fast or equivalently, if the robot matrices are slowly time-varying. The vector $v(k)$ in (31) is obtained from (16) as

$$v(k) = R_2\, \varepsilon(k-1) + R_3\, \varepsilon(k) \quad (33)$$

which in view of (13) with $\theta_{em}(k) \equiv 0$, is

$$v(k) = -(R_2 \theta_e(k-1) + R_3 \theta_e(k)) \equiv -\hat{\theta}_e(k) \quad (34)$$

Finally, using (31), (32) and (34), we obtain the gain adaptation laws given by (18).

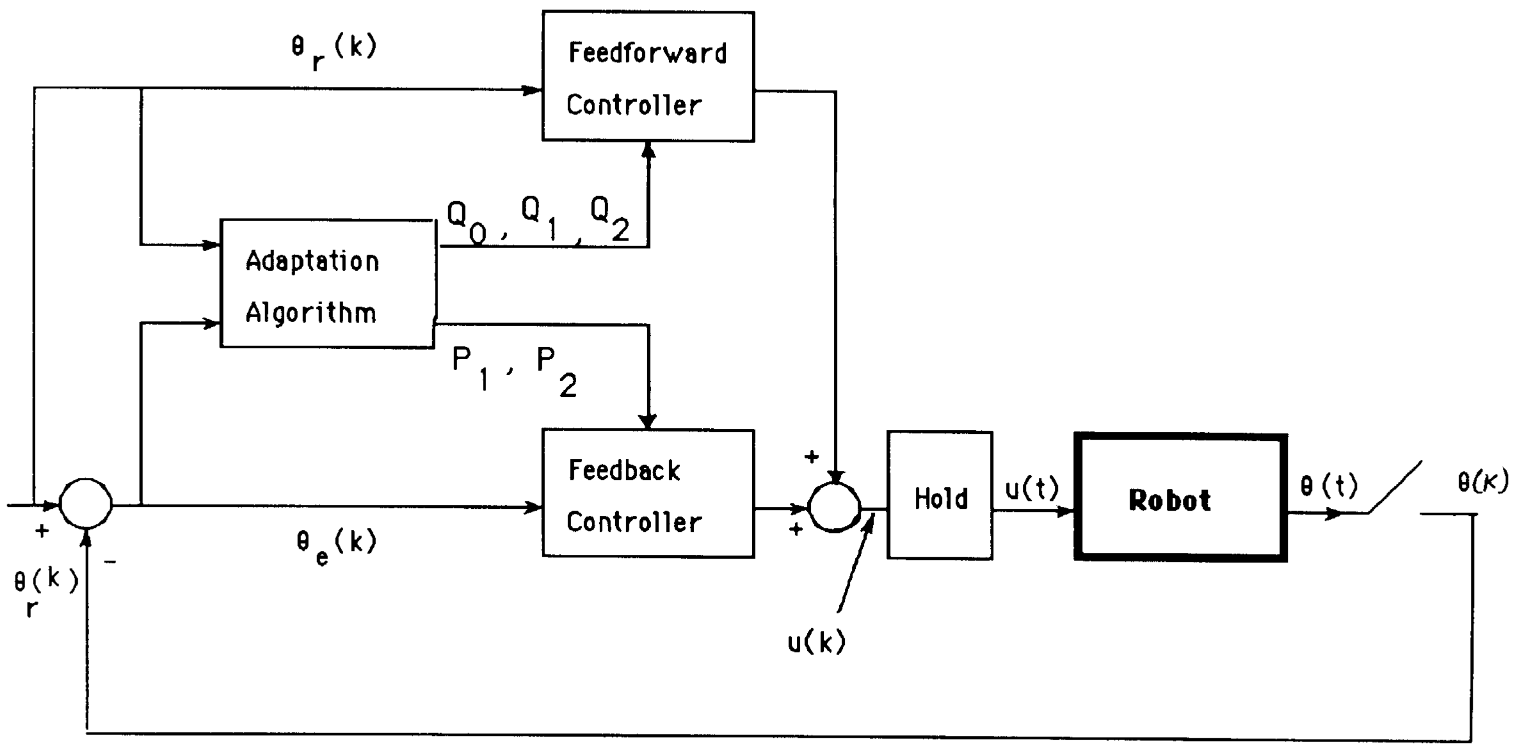


Figure 1 - Discrete-Time Adaptive Control Scheme

A DISCRETE DECENTRALIZED VARIABLE STRUCTURE ROBOTIC CONTROLLER

Zuheir S. Tumeh

School of Electrical Engineering
Georgia Institute of Technology
Atlanta , Georgia , 30332
Tel. (404) 894-3812

Abstract :

In this paper a decentralized trajectory controller for robotic manipulators is designed and tested using a multiprocessor architecture and a PUMA 560 robot arm. The controller is made up of a nominal model-based component and a correction component based on a variable structure suction control approach. The second control component is designed using bounds on the difference between the used and actual values of the model parameters. Since the continuous manipulator system is digitally controlled along a trajectory, a discretized equivalent model of the manipulator is used to derive the controller. The motivation for decentralized control is that the derived algorithms can be executed in parallel using a distributed, relatively inexpensive, architecture where each joint is assigned a microprocessor. Nonlinear interaction and coupling between joints is treated as a disturbance torque that is estimated and compensated for.

1. Introduction :

Strategies for designing manipulator controllers can generally be classified according to the degree of their dependence on the availability of reasonably accurate manipulator models. While some of these schemes, such as those based on systems with variable structure [1–10], model referenced [11,12], and self tuning controllers [13,14], are not necessarily model-based, others [15–22] depend to a varying extent on the availability of such models. Although controllers that belong to the first class are clearly robust to model inaccuracies, such schemes often disregard useful information embodied in the dynamic equations. Some of these approaches, however, have recently taken account of manipulator dynamics [2,4,5,11,12,22] in the form of additional nonlinear feedback. Model–based robot controllers, on the other hand, such as the computed torque control [15], are susceptible to deviations of the used model parameters from their actual values. More general nonlinear model-based control approaches [16,17] rely on using the complicated Lagrange-Euler inverse dynamic equations in real time. As a result, additional model inaccuracies are introduced if and when simplified versions of the L-E equations are used. Relatively few studies [17] have investigated the robustness of these control schemes to model parameter uncertainty.

It is generally agreed to in the literature that compensation for model inaccuracies is necessary to improve the robustness of model–based controllers. One form of such compensation, among others, is the use of the theory of systems of variable structure (VS) to compute auxiliary (or substitute) control signals. Many attempts in designing robotic VS controllers have relied on neglecting major components of the coupling torques between manipulator joints. Compensation for such torques is often left to the VS controller to achieve. The controller performance, however, can be significantly improved if estimates of these torques are also fed forward to compensate for them. While some efforts have ignored all components of the coupling torques (gravity as well as inertia and velocity coupling torques) and treated them as disturbances that can be compensated for by the VS controller [1,3,6,9,10], others have relied on direct computation of gravity torques [5] or have made use of the full set of dynamic equations [2,4,7,8]. The latter approach depends on the complexity of the manipulator dynamics and implementation of the control schemes of [2,4,7,8] for manipulators other than the used simple two and three link robots is computationally expensive. Most of the developed VS manipulator controllers have been tested by simulation using very simple (two or three link) robotic structures [1–3,6–10]. By contrast, few efforts have been tested experimentally using actual robot arms [5]. The majority of the reported analyses in this area use continuous time models [1,2,4,6–10] and ignore the effects of friction and damping encountered by the joint motors. Since the robot system is a continuous one that is digitally controlled along a trajectory (or towards a desired position), however, a discretized equivalent (or a sampled data) model of the manipulator is most relevant to this problem.

In this paper, a discretized equivalent model of the continuous robotic system is used at the joint level, taking into consideration all dynamic nonlinearities and sampling effects, to develop a decentralized linear time–varying controller. The motivation for decentralized control is that the developed control algorithms can be executed in parallel using a distributed, relatively inexpensive, architecture while avoiding the burden of computing a global nonlinear manipulator model in real time. Time schedules of the feedback gains and feedforward terms are computed off-line by computing the inverse dynamics along the desired trajectory. Due to uncertainty in some dynamic parameters, however, such as link inertial parameters, some coefficients of the discrete model are not exactly known. These coefficients also change as the robot configuration and load change. This is where the developed controller is modified using a variable structure suction control approach to compensate for model inaccuracies. The approach of this paper makes use of the knowledge of the model form and some of its poles and zeros. This results in a reduction of the number of unknown parameters and more accurate system representation. The developed controller is tested using a multiprocessor architecture and a six joint PUMA 560 robot arm. Each joint is assigned a microprocessor board based on an Intel 8086 processor. The parallel operation of the six processors is synchronized by a common clock. In section 2 of this paper the discrete manipulator model is presented along with the model-based controller. The VS-based controller is developed in section 3. Finally, section 4 presents the experimental testing of the developed controller.

2. A Discretized Equivalent Model and a Controller for a Manipulator Joint :

2.1. The Discretized Equivalent Model :

The discretized equivalent manipulator model developed in [21] is adopted in this paper since it is thought to account for nonlinear arm dynamics, joint motor electrical and mechanical characteristics, damping factors, friction, interference torque between joints, and sampling effects. A block diagram of this model with possible digital compensation and feedback filters is shown in Figure 1. The control voltage, V is output by each microprocessor joint controller to a digital–to–analog converter (DAC) which acts as a zero order hold (Z.O.H.) device. The DAC output voltage, V_{DAC}, is applied through a linear voltage amplifier to the joint motor input. V_a and V_m are the armature and motor voltages, R_a and L_a are the armature resistance and inductance, I_a is the armature current, k_v is the motor voltage constant, τ is the torque applied by the motor shaft, τ_d is the disturbance torque observed at the motor shaft which includes inertia and velocity coupling, gravity, friction, and other disturbance torques, J is the effective inertia at the motor shaft, B is the effective damping factor, n is the gear ratio, k' is a conversion constant, θ_a and θ_d are the actual and desired joint positions, and τ_d' is a feedforward compensation for τ_d. The system inside the dashed line is continuous while the one outside is described by the digital hardware and software used to control the joint.

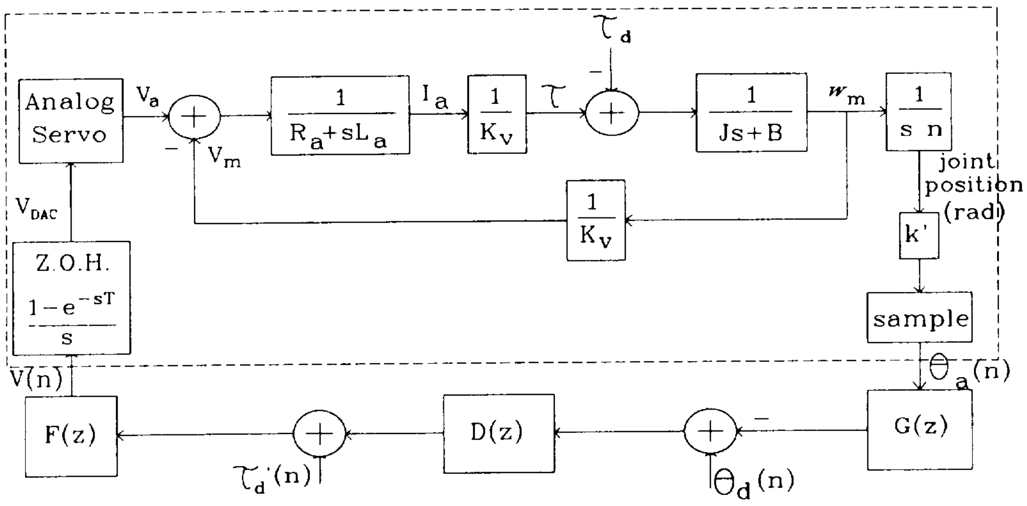


Figure 1. A Discretized Equivalent Joint Model

Assuming that J can be approximated by a constant within each sampling interval, the following transfer functions are obtained [21]

$$H(s) = \frac{\theta_a(s)}{V_{DAC}(s)} \approx \frac{kk'}{nk_vR_aJ}\frac{1}{s(s+s_1)} \quad , \quad F'(s) = \frac{\tau_d(s)}{V_{DAC}(s)} \approx \frac{k}{k_vR_a} \tag{1.a}$$

where
$$s_1 = \frac{1}{2}\left[\left(\frac{B}{J}+\frac{R_a}{L_a}\right) - \sqrt{\left(\frac{B}{J}-\frac{R_a}{L_a}\right)^2 - \frac{4}{k_v^2L_aJ}}\right] \tag{1.b}$$

where the inertia J_i and disturbance torque τ_{d_i} encountered by the rotor of joint i, are

$$J_i = \frac{D_{ii}}{n_i^2} + J_{r_i} \quad \text{and} \quad \tau_{d_i} = \frac{\tau_i - D_{ii}\ddot{\theta}_i}{n_i} + \tau_{f_i} \tag{2}$$

where D_{ii} and τ_i are the self inertia and torque of joint i computed using the inverse dynamic equations [15,22], J_{r_i}, τ_{f_i}, and n_i are the rotor inertia, friction torque, and gear ratio of joint i. Using an exact mapping of poles and zeros from the s plane into the z plane, ($z = e^{sT}$, where T is the sampling period), the discretized equivalents of the transfer functions (1) are [21]

$$H(z) = \frac{\theta_a(z)}{V_{DAC}(z)} = (1-z^{-1})Z\left(\frac{H(s)}{s}\right) \approx \frac{k_Hz^{-1}(1+z^{-1})}{(1-z^{-1})(1-p_1z^{-1})} \tag{3.a}$$

$$F'(z) = \frac{\tau_d(z)}{V_{DAC}(z)} = (1-z^{-1})Z\left(\frac{F'(s)}{s}\right) \approx k_{F'}z^{-1} \tag{3.b}$$

where
$$p_1 = e^{-s_1T} \quad , \quad k_H = \frac{Tkk'(1-p_1)}{2nk_vJR_as_1} \quad , \quad k_{F'} = \frac{k}{k_vR_a} \tag{3.c}$$

where the gains k_H and $k_{F'}$ are computed such that the steady state discrete system response is equal to the sampled steady state continuous system response.

2.2. The Digital Linear Time–Varying Controller :

Given the transfer functions (3) that represent a manipulator joint, the control task is to design the digital filters F(z), D(z), G(z), and a dynamics-based feedforward control signal such that θ_a tracks θ_d as closely as possible. To perform this task it is necessary here to note the timing of the model operation. At sample point n a desired position $\theta_d(n)$ and an estimated feedforward term $\tau_d'(n)$ that compensates for τ_d are input to the system while the actual joint position $\theta_a(n)$ is observed. It is desired that the actual position at the next sample time (n+1) be equal to the desired position input to the system at the current sample time i.e. $\theta_a(n+1) = \theta_d(n)$. To achieve this it is necessary to estimate a discrete input, $\tau_d'(n)$, that is equivalent to τ_d ahead of time to compensate for τ_d between the two sample times. This feedforward compensation signal is computed off-line using the robot inverse dynamic equations. Hence, it is desired first to have $\tau_d(z) = z^{-1}\tau_d'(z)$. As a result, one gets [21]

$$F(z) = \frac{V(z)}{\tau_d'(z)} = \frac{z^{-1}}{F'(z)} = \frac{1}{k_{F'}} \tag{4.a}$$

After simple manipulations, the filters D(z) and G(z) that result in the response $\theta_a(z) = z^{-1}\theta_d(z)$ are found to be [21]

$$G(z) = 1 \qquad \text{and} \qquad D(z) = \frac{k_{F'}}{k_H}\frac{1-p_1 z^{-1}}{1+z^{-1}} \tag{4.b}$$

The feedback control is then written as

$$V(z) = D(z)F(z)[\theta_d(z) - \theta_a(z)] + F(z)\tau_d'(z) \qquad \text{with} \qquad D(z)F(z) = \frac{1}{k_H}\frac{1-p_1 z^{-1}}{1+z^{-1}} \tag{5}$$

In the discrete time domain, the controller is written as

$$V(n) = \frac{1}{k_H^u(n)}[\Delta\theta(n) - p_1^u(n)\Delta\theta(n-1)] + \frac{1}{k_F'}[\tau_d'^u(n) + \tau_d'^u(n-1)] - V(n-1) \tag{6.a}$$

$$\text{where} \qquad \Delta\theta(n) = \theta_d(n) - \theta_a(n) \tag{6.b}$$

and superscript u denotes used (as opposed to actual) values. It remains, however, to compute the feedback gains $k_H^u(n)$, $p_1^u(n)$, and $\tau_d'^u(n)$. Since the only information available to a joint controller in a decentralized control environment is that generated off-line, estimates of these parameters are computed off-line, when the desired trajectory is generated, and later used in real time to implement the controller (6). This scheme is based on a computed torque approach and, as a result, is susceptible to the potential problems facing a computed torque approach. The most serious problem of these is the difference between the values of J and τ_d along the actual path and their nominal (used) values along the desired path. To address this problem, the controller (6) is modified using the theory of systems of variable structure (VS) to compensate for parameter uncertainty. The modified controller is developed in the next section.

3. Development of The Variable Structure Controller :

The purpose of the VS-based controller introduced in this section is to modify the controller (6) such that deviations of the used model parameters from their actual values are compensated for. The basic form of the controller (6) is, however, maintained since it is based on the actual model form which is known. An appropriate sliding surface and a switching variable are selected in terms of the joint tracking error and a suction control strategy [2,4] is used in the discrete time domain to design the controller such that the switching variable and tracking error converge to zero. First, the model (3) is written in the discrete time domain in terms of the actual model parameter values as

$$\begin{aligned}\theta_a(n) =& \theta_a(n-1) + p_1(n-1)[\theta_a(n-1) - \theta_a(n-2)] \\ &+ k_H(n-1)[V(n-1) + V(n-2)] - \frac{k_H(n-1)}{k_F'}[\tau_d'(n-1) + \tau_d'(n-2)]\end{aligned} \tag{7}$$

and the switching surface for each joint i is defined as

$$s(n) = e(n) + \lambda e(n-1) \qquad , \qquad e(n) = \theta_d(n-1) - \theta_a(n) \tag{8}$$

such that the trajectory tracking error decays exponentially when the switching variable is driven to zero. The switching variable s is "sucked" to zero using a discretized version of the suction control strategy outlined in the continuous time domain in [2,4]. Namely, the controller is designed so that

$$s(n-1)\ [s(n)-s(n-1)]<0 \tag{9}$$

such that $s(n) > s(n-1)$ if $s(n-1) < 0$ and $s(n) < s(n-1)$ if $s(n-1) > 0$. Condition (9) is not sufficient for the convergence of s(n) to zero. If $|s(n)-s(n-1)|$ can, however, be shown to be bounded by a small positive number δ then s(n) can also be easily shown to be bounded by a small positive number δ' using condition (9). To show this, we write

$$\begin{aligned}|s(n)-s(n-1)| =&|e(n)-e(n-1)+\lambda[e(n-1)-e(n-2)]|\\ =&|\theta_d(n-1)-\theta_d(n-2)-[\theta_a(n)-\theta_a(n-1)]\\ &+\lambda\{\theta_d(n-2)-\theta_d(n-3)-[\theta_a(n-1)-\theta_a(n-2)]\}|\\ <&|\theta_d(n-1)-\theta_d(n-2)|+|\theta_a(n)-\theta_a(n-1)|\\ &+|\lambda|[|\theta_d(n-2)-\theta_d(n-3)|+|\theta_a(n-1)-\theta_a(n-2)|]\\ <&\delta_1+\delta_2+|\lambda|(\delta_3+\delta_4)<\delta\end{aligned}$$

where the magnitude of the upper bound δ is determined by the speed of the desired trajectory, the manipulator mechanical time constants, and the sampling frequency. It is clear that δ decreases with increasing sampling frequency. For example, if the desired and actual joint speeds are bounded by 10 rad/s and the sampling frequency is 100 Hz, then $|s(n)-s(n-1)|$ is bounded by 0.25 radians for $\lambda = 0.25$. It is clear that as the sampling frequency increases infinitely, condition 10 tends to the suction control condition $s\dot{s} < 0$ of [4]. Although the condition

$$s(n)\ [s(n)-s(n-1)]<0 \tag{9)'$$

is more attractive since it ensures that $0 > s(n) > s(n-1)$ if $s(n-1) < 0$ and $0 < s(n) < s(n-1)$ if $s(n-1) > 0$, the design of a controller that would satisfy condition (9)' is quite complicated mathematically (as will be clear from the proof of Lemma 1.). Next, we proceed to design a controller that satisfies condition (9). First, the following set of upper bounds on model parameter deviations is defined

$$\alpha \geq \frac{k_H^u(n)}{k_H(n)} \geq \frac{1}{\alpha} \quad , \quad \alpha \geq 1 \quad , \quad \beta \geq |p_1^u(n)-p_1(n)| \quad , \quad \gamma \geq |r_d'^u(n)-r_d'(n)| \quad , \quad n \geq 0 \tag{10}$$

The variable structure controller that satisfies condition (9) is given by the following lemma.

__Lemma 1 :__ The control

$$\begin{aligned}V(n)=&\frac{1}{k_H^u(n)}\left[\frac{2}{\alpha+1/\alpha}u_1(n)+\alpha\left(\frac{\alpha-1/\alpha}{\alpha+1/\alpha}|u_1(n)|+\beta|\theta_a(n)-\theta_a(n-1)|\right)\ sgn\ s(n)\right]\\ &-V(n-1)+\frac{1}{k_F'}[r_d'^u(n)+r_d'^u(n-1)+2\gamma\ sgn\ s(n)]\end{aligned} \tag{11.a}$$

$$\text{where}\quad u_1(n)=\theta_d(n)-\theta_a(n)-p_1^u(n)[\theta_a(n)-\theta_a(n-1)]-(1-\lambda)e(n)-\lambda e(n-1) \tag{11.b}$$

Satisfies the convergence condition (9).

__Proof :__ Using the controller (11), the closed loop system response (7) is rewritten as

$$\begin{aligned}\theta_a(n)=&\frac{k_H(n-1)}{k_H^u(n-1)}\frac{2}{\alpha+1/\alpha}[\theta_d(n-1)-(1-\lambda)e(n-1)-\lambda e(n-2)]\\ &+\left\{1+p_1(n-1)-\frac{k_H(n-1)}{k_H^u(n-1)}\frac{2}{\alpha+1/\alpha}[1+p_1^u(n-1)]\right\}\theta_a(n-1)\\ &-\left[p_1(n-1)-\frac{k_H(n-1)}{k_H^u(n-1)}\frac{2}{\alpha+1/\alpha}p_1^u(n-1)\right]\theta_a(n-2)\end{aligned}$$

$$+\alpha\frac{k_H(n-1)}{k_H^u(n-1)}\left(\frac{\alpha-1/\alpha}{\alpha+1/\alpha}|u_1(n-1)|+\beta|\theta_a(n-1)-\theta_a(n-2)|\right)\ sgn\ s(n-1)$$
$$-\frac{k_H(n-1)}{k_F'}[r_d'(n-1)-r_d'^u(n-1)+r_d'(n-2)-r_d'^u(n-2)-2\gamma\ sgn\ s(n-1)] \tag{12}$$

Using equation (12) and defining the following quantities,

$$\varsigma(n)=\frac{k_H(n)}{k_H^u(n)}\quad,\quad \tilde{p}_1(n)=p_1(n)-p_1^u(n)\quad,\quad \tilde{r}_d'(n)=r_d'(n)-r_d'^u(n) \tag{13}$$

the increase in the value of the switching variable s between sample times n-1 and n is

$$\begin{aligned}
s(n)-s(n-1)=&e(n)-(1-\lambda)e(n-1)-\lambda e(n-2)\\
=&\theta_d(n-1)-\theta_a(n)-(1-\lambda)e(n-1)-\lambda e(n-2)\\
=&\left(1-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}\right)[\theta_d(n-1)-(1-\lambda)e(n-1)-\lambda e(n-2)]\\
&-\left[1-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}+p_1(n-1)-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}p_1^u(n-1)\right]\theta_a(n-1)\\
&+\left[p_1(n-1)-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}p_1^u(n-1)\right]\theta_a(n-2)\\
&-\alpha\varsigma(n-1)\left[\frac{\alpha-1/\alpha}{\alpha+1/\alpha}|u_1(n-1)|+\beta|\theta_a(n-1)-\theta_a(n-2)|\right]\ sgn\ s(n-1)\\
&+\frac{k_H(n-1)}{k_F'}[\tilde{r}_d'(n-1)+\tilde{r}_d'(n-2)-2\gamma\ sgn\ s(n-1)]\\
=&\left(1-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}\right)u_1(n-1)-\tilde{p}_1(n-1)[\theta_a(n-1)-\theta_a(n-2)]\\
&-\alpha\varsigma(n-1)\left[\frac{\alpha-1/\alpha}{\alpha+1/\alpha}|u_1(n-1)|+\beta|\theta_a(n-1)-\theta_a(n-2)|\right]\ sgn\ s(n-1)\\
&+\frac{k_H(n-1)}{k_F'}[\tilde{r}_d'(n-1)+\tilde{r}_d'(n-2)-2\gamma\ sgn\ s(n-1)]\\
=&\left(1-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}\right)u_1(n-1)-\alpha\varsigma(n-1)\frac{\alpha-1/\alpha}{\alpha+1/\alpha}|u_1(n-1)|\ sgn\ s(n-1)\\
&-\tilde{p}_1(n-1)[\theta_a(n-1)-\theta_a(n-2)]\\
&-\alpha\varsigma(n-1)\beta|\theta_a(n-1)-\theta_a(n-2)|\ sgn\ s(n-1)\\
&+\frac{k_H(n-1)}{k_F'}[\tilde{r}_d'(n-1)+\tilde{r}_d'(n-2)-2\gamma\ sgn\ s(n-1)]
\end{aligned} \tag{14}$$

Hence,

$$\begin{aligned}
\kappa(n)=&s(n-1)[s(n)-s(n-1)]\\
=&\left[\left(1-\frac{2\varsigma(n-1)}{\alpha+1/\alpha}\right)sgn\ [u_1(n-1)s(n-1)]-\alpha\varsigma(n-1)\frac{\alpha-1/\alpha}{\alpha+1/\alpha}\right]|u_1(n-1)s(n-1)|\\
&-[\tilde{p}_1(n-1)\ sgn\ \{s(n-1)[\theta_a(n-1)-\theta_a(n-2)]\}+\alpha\varsigma(n-1)\beta]\times\\
&\qquad\qquad|s(n-1)[\theta_a(n-1)-\theta_a(n-2)]|\\
&+\frac{k_H(n-1)}{k_F'}\{[\tilde{r}_d'(n-1)+\tilde{r}_d'(n-2)]\ sgn\ s(n-1)-2\gamma\}|s(n-1)|\\
=&a(n)|u_1(n-1)s(n-1)|+b(n)|s(n-1)[\theta_a(n-1)-\theta_a(n-2)]|+c(n)|s(n-1)|
\end{aligned} \tag{15}$$

and $\kappa(n)$, $n>0$, will be shown to be negative by showing that $a(n)$, $b(n)$, and $c(n)$, are negative.

1. To show that $a(n)$ is negative, we need to show that

$$|\alpha + 1/\alpha - 2\varsigma(n-1)| < |\alpha\varsigma(n-1)(\alpha - 1/\alpha)| \tag{16}$$

Since the right hand side of inequality (16) is positive, two cases are at hand

a) If $\alpha + 1/\alpha - 2\varsigma(n-1) > 0$, then we need to show that

$$\alpha + 1/\alpha - 2\varsigma(n-1) < \alpha^2\varsigma(n-1) - \varsigma(n-1)$$

or

$$1/\alpha - \varsigma(n-1) < \alpha[\alpha\varsigma(n-1) - 1]$$

or

$$1 - \alpha\varsigma(n-1) < \alpha^2[\alpha\varsigma(n-1) - 1]$$

This inequality is satisfied since $\alpha^2 > \pm 1$. Hence, $a(n)$ is negative.

b) If $\alpha + 1/\alpha - 2\varsigma(n-1) < 0$, then we need to show that

$$2\varsigma(n-1) - \alpha - 1/\alpha < \alpha^2\varsigma(n-1) - \varsigma(n-1)$$

or

$$\alpha^2[\alpha\varsigma(n-1) + 1] > 3\alpha\varsigma(n-1) - 1$$

or

$$\alpha^2 > \frac{3\alpha\varsigma(n-1) - 1}{\alpha\varsigma(n-1) + 1}$$

but

$$\frac{3\alpha\varsigma(n-1) - 1}{\alpha\varsigma(n-1) + 1} < \frac{3\alpha^2 - 1}{\alpha^2 + 1}$$

and

$$\frac{3\alpha^2 - 1}{\alpha^2 + 1} < \alpha^2$$

since

$$\alpha^4 + \alpha^2 - 3\alpha^2 + 1 = (\alpha^2 - 1)^2 > 0$$

Hence, $a(n)$ is negative.

2. To show that $b(n)$ is negative, we need to show that

$$|\tilde{p}_1(n-1)| < \alpha\varsigma(n-1)\beta$$

This inequality is satisfied since $|\tilde{p}_1(n-1)| < \beta$ and $\alpha\varsigma(n-1) > 1$ by definitions (10).

3. $c(n)$, $n > 0$, is negative since $|\tilde{r}_d'(n)| < \gamma$ by definition (10).

Hence, it is seen that $\kappa(n) < 0$, $n > 0$, and condition (9) is satisfied.

Q.E.D.

The next section presents the experimental testing of the developed controller.

4. An Example : A PUMA 560 Manipulator :

To obtain model parameters for a PUMA 560 arm, the motor and armature circuit parameters k_v, J_r, and L_a, were obtained from the manufacturer. R_a was measured for each joint by applying a DC voltage at the DAC output and measuring the armature current when no motion took place. To obtain the damping factor and friction torque for each joint a DC voltage was applied at the DAC output and the armature current and joint speed were measured. The resulting data points of current versus speed yielded B_i and τ_{f_i} using regression techniques. The linear voltage amplifier gain was set to 4. It was also necessary to adopt a set of link dynamic parameters. There are few reported efforts directed at identifying parameters not supplied by manufacturers such as inertial parameters and centroid coordinates. While some of these efforts adopt direct geometric approaches [22], others rely on experimental identification of these parameters [23–25]. Many approximations are made in [22] about link mass distribution and component shapes. Identification techniques require acceleration, torque, and force sensors and the results often bear a lot of noise [24]. In this paper, link masses were obtained from the manufacturer. Most of the centroid coordinates and inertial parameters reported in [22] were thought to be reasonably accurate and were used. All of the used model parameters for the used PUMA 560 arm are listed in [21]. α, β, and λ were set to

2, 0.2, and 0.25 respectively for all joints. γ was set to 0.5 N.m for the first three joints and 0.05 N.m for the last three joints.

The desired trajectory for the arm was specified by a cartesian path and a desired velocity profile for the end–effector. The path, sampled every 30 ms, consisted of three curves defined in the arm base frame. The first curve was a semicircle that started at (x,y,z)=(15,75,10)cm and ended at (5,65,-10)cm. The point (x,y,z)=(10,50,0)cm was in the motion plane and the semicircle followed was the one closer to this point. The end–effector accelerated from rest to a velocity of 0.35 m/s in the first 3 segments (90 ms), cruised at this speed for 34 segments, and decelerated to rest in the last 3 segments. The hand approach vector, a, was required to change from (0,0.9798,-0.2) to (0,0.9798,0.2) by requiring the angle

$$\phi = tan^{-1} \frac{a_z}{\sqrt{1 - a_z^2}}$$

to change from $-11.5°$ to $11.5°$ by accelerating in the first 3 segments, cruising at a constant speed in the middle 34 segments, and decelerating to rest in the last 3 segments. The second curve was a straight line that ended at (x,y,z)=(-5,85,-15)cm. The approach vector, a, changed to (0,0.9539,0.3). All velocity profiles were similar to those of the first curve except for the numbers of acceleration, constant speed, and deceleration segments which were 9, 7, and 9 respectively, and the end–effector constant speed which was 0.48 m/s. The third curve was a semicircle that ended at the initial arm configuration of the first curve. The point (x,y,z)=(5,60,-2.5)cm was in the motion plane and the semicircle followed was the one closer to this point. All velocity profiles were similar to those of the first curve except for the numbers of acceleration, constant speed, and deceleration segments which were 5, 45, and 5 respectively. The arm stayed at rest for 5 segments between each two curves. The corresponding desired joint trajectories are shown in Figure 2. The used PUMA arm was driven very close to its maximum speed.

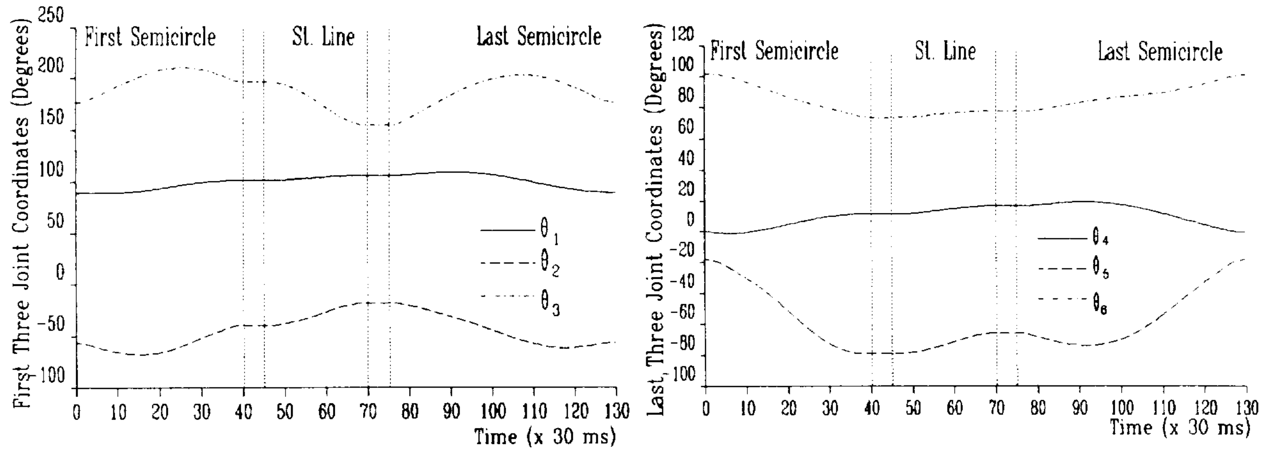
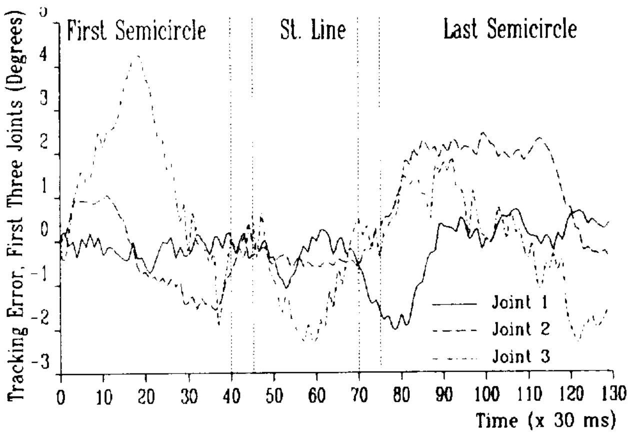


Figure 2. Desired Joint Trajectories

The arm was sampled every 10 ms and desired joint positions were generated every 10 ms by linear interpolation between their values stored for use every 30 ms. The feedback gains computed off–line were used three times within the 30 ms intervals each 10 ms (i.e. T=0.01 sec). The computation delay at each sampling interval was 1.00 ms. The joint trajectory tracking errors resulting from this experiment are shown in Figure 3. The tracking error is bounded by 2 degrees for joints 1 and 2, 4 degrees for joint 3, 0.5 degrees for joint 4, 1.5 degrees for joint 5, and 0.25 degrees for joint 6. This performance is slightly worse than that of the controller of [21] which is similar to the controller of this paper except for the absence of the VS–based compensation for model inaccuracy.

One probable cause for the increase in the tracking error, compared to the results of [21], is the chattering problem associated with the VS-based control. This chattering effect is clear in the high frequency behavior of the tracking error of Figure 3 and was felt clearly when the used manipulator exhibited noisy gittery motions during the performed experiments. Another probable cause for the increase in the tracking error is the increase in the computation delay (which is 0.55 ms for the controller of [21]). It does not appear that using a VS-based control, in the experimental context and setup described in this paper, to compensate for model parameter uncertainty has offered an advantage over using parameter estimates computed off-line.



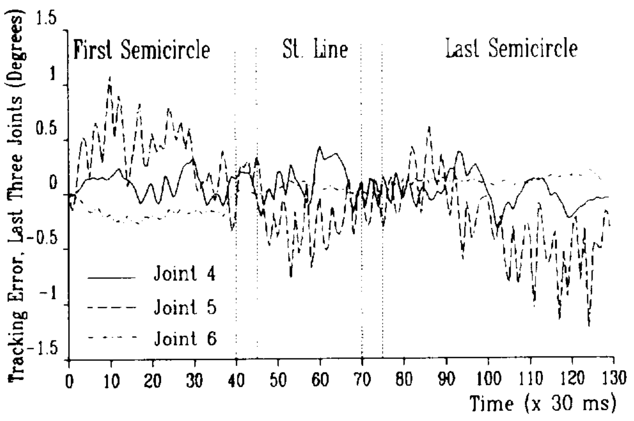


Figure 3. Joint Tracking Errors

5. Conclusion :

A decentralized digital linear time-varying variable structure trajectory controller for manipulator arms was developed and tested. A discretized equivalent model of the continuous manipulator system was used to design a nominal digital linear time-varying feedback. Time schedules of the estimated values of the feedback gains and feedforward terms were generated off-line. The feedback was modified using the theory of systems with variable structure to compensate for the difference between the used and actual values of the model parameters. The controller performs reasonably well considering that the used PUMA arm was driven along the trajectory at its maximum speed.

References :

[1] Young, K., "Controller Design for a Manipulator Using Theory of Variable Structure Systems", *IEEE Tran. Sys. Man & Cyb.*, Vol. SMC-8, No. 2, pp. 210–218, Feb., 1978.

[2] Slotine, J. and Sastry, S., "Tracking Control of Nonlinear Systems Using Sliding Surfaces, With Application to Robot Manipulators", *Int. J. Con.*, Vol. 38, No. 2, pp. 465–492, 1983.

[3] Morgan, R. and Özgüner, Ü, "A Decentralized Variable Structure Control Algorithm for Robotic Manipulators", *IEEE J. Rob. Aut.*, Vol. RA-1, No. 1, pp. 57–65, March, 1985.

[4] Slotine, J., "The Robust Control of Robot Manipulators", *Int. J. Rob. Res.*, Vol. 4, No. 2, pp. 49-64, Summer, 1985.

[5] Harashima, F. et. al. "Practical Robust Control of Robot Arm Using Variable Structure System", *Proc. IEEE Conf. Rob. & Aut.*, San Francisco, CA, pp. 532–539, April, 7–10, 1986.

[6] Young, K., "A Variable Structure Model Following Control Design for Robotics Applications", *Proc. IEEE Conf. Rob. & Aut.*, San Francisco, CA, pp. 540–545, April, 7–10, 1986.

[7] Slotine, J., "On Modeling and Adaptation in Robot Control", *Proc. IEEE Conf. Rob. & Aut.*,

San Francisco, CA, pp. 1387–1392, April, 7–10, 1986.

[8] Slotine, J. and Coetsee, J., "Adaptive Sliding Controller Design for Nonlinear Systems", *Int. J. Cont.*, Vol. 44, 1986.

[9] Özgüner, Ü et. al. "Decentralized Variable Structure Control of a Two–Arm Robotic System", *Proc. IEEE Conf. Rob. & Aut.*, Raleigh, NC, pp. 165-168, March, 31–April, 3, 1987.

[10] Pandian, S. et. al. "A Decentralized Variable Structure Model Following Controller for Robot Manipulators", *Proc. IEEE Conf. Rob. & Aut.*, Philadelphia, PA, 1988.

[11] Takegaki, M. and Arimoto, S., "An Adaptive Trajectory Control of Manipulators", *Int. J. Cont.*, Vol. 34, No. 2, pp. 291-239, 1981.

[12] Lim, K. and Eslami, M., "Robust Adaptive Controller Designs for Robot Manipulator Systems", *IEEE J. Rob. Aut.*, Vol. RA–3, No. 1, pp. 54-66, February, 1987.

[13] Koivo, A. and Guo, T., "Adaptive Linear Controller for Robotic Manipulators", *IEEE Tran. Aut. Con.*, Vol. 28, No. 2, pp. 233-242, 1983.

[14] Lee, C. and Chung, M., "An Adaptive Control Strategy for Mechanical Manipulators", *IEEE Tran. Aut. Con.*, Vol. 29, No. 9, pp. 837-840, 1984.

[15] Markiewicz, B., "Analysis of The Computed Torque Drive Method and Comparison with Conventional Position Servo for a Computer Controlled Manipulator", JPL Tech. Memo. 33-601, N73-21198, 1973.

[16] Freund, E., "Fast Nonlinear Control with Arbitrary Pole Placement for Industrial Robots and Manipulators", *Int. J. Rob. Res.*, Vol. 1, No. 1, pp. 65-78, 1982.

[17] Ha, I. and Gilbert, E., "Robust Tracking in Nonlinear Systems", *IEEE Tran. Aut. Con.*, Vol. 32, No. 9, pp. 763-771, 1987.

[18] Leahy, M., and Saridis, G., "The Effects of Unmodelled Forces on Robot Control", *Proc. IEEE* 25th *Conf. Dec. & Con.*, Athens, Greece, pp. 403-408, Dec., 1986.

[19] Khosla, P., "Some Experimental Results on Model–Based Control Schemes", *Proc. IEEE Int. Conf. Rob. Aut.*, Philadelphia, PA, pp. 1380–1385, April, 24–29, 1988.

[20] An, C. et. al. "Model–Based Control of a Direct Drive Arm, Part II: Control", *Proc. IEEE Int. Conf. Rob. Aut.*, Philadelphia, PA, pp. 1386–1391, April, 24–29, 1988.

[21] Tumeh, Z., "A Gain Scheduling Approach in Decentralized Manipulator Control Using Discretized Equivalent Joint Models", *Proc.* 27th *IEEE Conf. Dec. Con.*, Austin, TX, Dec., 7–9, 1988.

[22] Paul, R. and Rong, M., "The Dynamics of The PUMA Manipulator", Purdue Univ. Tech. Rep. EE82-34, 1982.

[23] Olsen, H., and Bekey, G., "Identification of Robot Dynamics", *Proc. IEEE Int. Conf. Rob. Aut.*, San Francisco, CA, pp. 1004–1010, April, 7–10, 1986.

[24] Atkenson, C., An, C., and Hollerbach, J., "Estimation of Inertial Parameters of Manipulator Loads and Links", *Int. J. Rob. Res.*, Vol. 5, No. 3, pp. 101–119, Fall, 1986.

[25] An, C. et. al. "Model–Based Control of a Direct Drive Arm, Part I: Building Models", *Proc. IEEE Int. Conf. Rob. Aut.*, Philadelphia, PA, pp. 1374–1379, April, 24–29, 1988.

TELEMANIPULATION

CONSTRUCTION AND DEMONSTRATION OF A 9-STRING 6 DOF FORCE REFLECTING JOYSTICK FOR TELEROBOTICS

Randel Lindemann

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91109

Delbert Tesar

The University of Texas at Austin
Austin, Texas 78705

Abstract

Confrontation with difficult manipulation tasks in hostile environments such as space, has led to the development of means to transport the human's senses, skills and cognition to the remote site. We examined the use of advanced Telerobotics to achieve this goal. A novel and universal hand controller based on a fully parallel mechanical architecture is discussed. The design and implementation of this 6 DOF force reflecting joystick is shown in relationship to the general philosophy of achieving telepresence in a man-machine system.

1. Introduction

This paper describes the work undertaken at the University of Texas at Austin to construct and implement a force reflecting universal hand controller in a microprocessor driven testbed with an industrial robot as discussed in [1]. The Texas 9-string kinesthetic joystick has been interfaced to a robotic manipulator and a microprocessor to realize a prototype telerobotic system. The system is a generalization of the industrial bilateral master-slave teleoperator. The man-machine interface is universal and therefore capable of positioning and orienting any 6 DOF manipulator once the suitable transformation changes are made in the controlling software. The 9 string kinesthetic joystick represents the extension of force reflection to the original 9 string unilateral joystick developed by Tesar and Lipkin as discussed in [2]. The design of the joystick has been based on maximizing its capability to convey telepresence through a novel parallel architecture which is actuated in antagonism. The Texas telerobotics system represents an experimental test facility for research into the engineering and human factors issues of man-machine interface. In the table below the design goals of the project are given.

	System Functional Attributes
1) decoupled interface:	no geometric similarity required between master and slave
2) motion projection:	projection of commanded motion
3) variable control point:	electronic control point selection
4) accommodation:	manipulator motion is altered by the end-effector
5) coordinated control:	operator directly controls end-effector motion
6) motion filtering:	jitters and jerks in input motion removed
7) positional scaling:	variable positional gain between interface and manipulator
8) indexing:	controller repositioning
9) reorientation:	compensation for operator perspective

2. Man-machine interface

Because of man's need to have control or an effective presence to do manipulation in remote or hostile environments the teleoperator system (TOS) has been developed. These TOS have given man the ability to extend his strength and dexterity along with his intelligence into the remote site. Historically, the TOS consisted of two manipulator arms which were geometrically identical. One arm, called the master served as the control input device positioned by the operator. The other arm, called the slave, could if servoed sense and feedback its load state to the master arm.

The importance of TOS is in its capacity to extend to a human operator the remote control of the full 6 degrees-of-freedom (DOF) of rigid body motion through the positioning of one hand. In an advanced TOS the man is only one component along with the computational base, remote manipulator, display facility, sensory hardware, communication system, and the control input device. In figure 1 below, a schematic of such a generic system is shown with the arrows indicating the flow of communication signals.

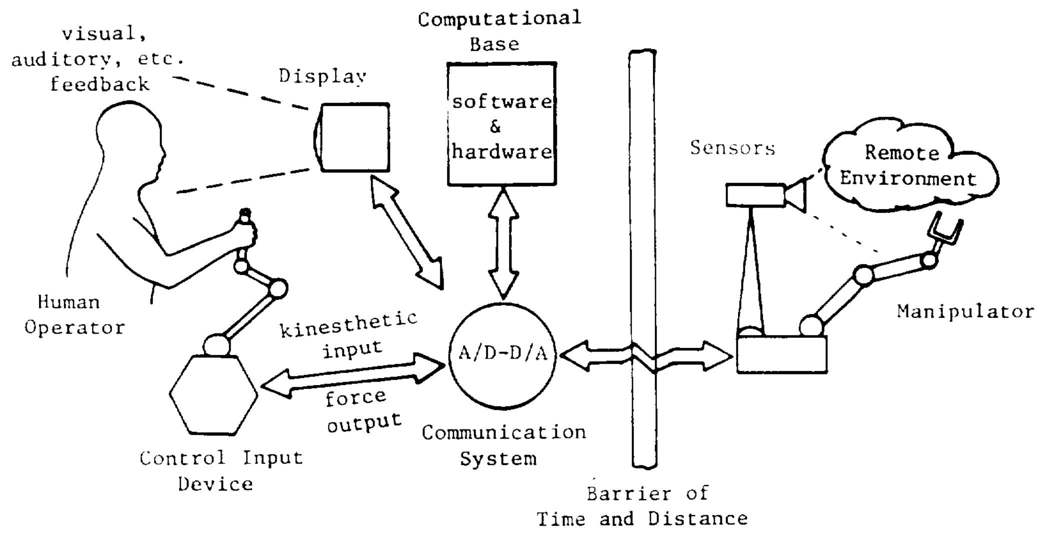


Figure 1: Advanced Generic Teleoperator System

An important characteristic of a TOS is the degree to which the operator is made to feel he is at the remote site actually performing the manipulation task. This illusive design feature is termed 'telepresence'. Studies conducted on advanced TOS indicate the need for force feedback to the operator from the remote manipulator. Thus, the TOS must condense the vast quantity of data that is echoed back from the environment of the manipulator into a form favorable to human perception and interpretation. Simultaneously, the human's limited output must contain sufficient information for unambiguous interpretation by the computational base. The result is a very comprehensive control input device. To achieve the most effective relationship between the operator and the manipulator, the control input device or man-machine interface should be effectively transparent to the information flowing through it.

In order to function efficiently, the TOS used in general, unstructured tasks will require specific slave manipulator geometries which may vary greatly in size. The man-machine interface must be constructed with respect to its utility as a control input and kinesthetic feedback device. The intersection of these two demands dictates the need for a universal manual controller. The universal controller is one which is fully software driven and requires a computational base or machine intelligence to drive two geometrically dissimilar manipulator arms. The inclusion of a machine intelligence into the TOS, frees the man-machine interface designer from the restrictions of kinematic replication and limited control in the development of a generalized master-slave TOS.

As indicated earlier, the state-of-the-art in TOS are replica (geometrically identical) master/slave systems, essentially a 30 year old technology that will not be adequate in difficult task environments such as orbital and interplanetary space. These systems lack transparency in the bilateral flow of communication that causes the operator to be between 2 and 20 times as slow as his functioning without a TOS, and generally precludes altogether complex tasks.

Our belief is that the most effective TOS will incorporate a universal man-machine interface optimized in its design to the relevant human factors involved in order to achieve telepresence. As a result the man-machine interface will have a geometry distinct and decoupled from that of the manipulator being controlled. The interface will then require a computer to augment the human intelligence as a computational base performing the needed geometric transformations between the man-machine interface and the manipulator. The form of this idealized controller is a universal bilateral position controller.

Advances in the last 25 years have also led to the development of programmable and autonomous manipulator arms called robots. A recent result has been their combination with TOS into a hybrid form of system called 'telerobotics'. The resulting system can be defined as a robotic system which in addition to its usual autonomous modes of operation can take control information directly from a human operator through a man-machine interface thus becoming teleoperated; or from a higher, supervisory level of executive control, thereby acting in a semi-autonomous manner. By making the universal master controller bilateral, the resulting system becomes conceptually two dissimilar cooperating robots, software coupled and running in real time.

3. Design and Analysis

In the past, problems with man-machine interfaces have included their inertia, backlash in their drive trains, friction, and limited or non-ergonomic motion capability. Transparency in the flow of communication signals requires that the inertial dynamics and friction effects of the man-machine interface be well below the intended feedback level in order to avoid operator confusion between signal and noise. In this project an isotropic controller has been sought with a constant (but programmable) joystick-to-end-effector position mapping and end-effector-to-joystick load state mapping.

Usually in a TOS or telerobotic system the hand controller is designed around the robot or remote manipulator arm which is designed around the tasks it is meant to perform. In contrast, our goal has been to design a universal man-machine interface around the human operator and use the necessary geometric software transformations in a computer. The forerunner of this project is the work of Lipkin (1983) in the design and construction of a unilateral 9-string joystick [3]. This work had then been followed by the initial configuration study for a 9-string bilateral joystick finished in 1986 by Agronin [4].

Therefore, we designed the joystick to minimize the interface dynamics and maximize the force feedback capability. In order to reduce the inertia associated with each of the air cylinders, an optimization has been performed and the point near the air cylinders closest to its moving centroid has been chosen as the pivot point in order to minimize the inertia. The air cylinder is connected to a universal joint by brackets. An additional benefit to this choice is that the air cylinder is supported near its center of gravity and most of the weight of the air cylinders is off loaded.

In order to maximize the force feedback a geometric optimization of the geometry of the 9-string joystick has been performed. The optimization has been used to design for the largest fixed minimum of maximum force feedback for use in the open loop control of the feedback signal. In order to assure an isotropic nature to the force feedback, the smallest maximum force that can be generated anywhere in the joystick workspace in any direction is the limiting factor. In order to maximize that quantity an analysis has been completed which relates the minimum force maximas to the geometry of the base triangles (where the cables emanate from the supporting structure of the joystick frame), the distance from the base triangle to the center point of the joystick workspace, and the air cylinder constant force.

The analysis approach (detailed in [1]) finds the algebraic rule that expresses the minimum of maximum force in a plane and then rotates the plane about the workspace center point and the line of action of air cylinder constant force. The calculus of minimization in one variable has then been used to find the minimum of force maximums. The technique has then been developed into a computer program which uses a global search technique to scan the joystick workspace. The program is interactive and the user inputed design factors in an adaptive fashion. The force feedback has been found to degrade as the volume of workspace increases. The final design chose an equal angle of 34.5 degrees between the strings and air cylinder shafts at the workspace center point, a pivotal offset of 0.0 inches, and an equilateral base triangle dimension of 20.83 inches.

As indicated earlier, the inertial dynamics of a kinesthetic controller is an important description of its quality of transparency or fidelity (signal-to-noise ratio). Therefore, a method of dynamic simulation has been performed based on the method of Tesar and Freeman [5]. The method uses dynamic equations based on influence coefficients which separate the purely position dependent functions from those which are time dependent (velocity, acceleration, etc.). An interactive program has been written and run simulating the Texas 9-string joystick undergoing a variety of path motions under representative velocities and accelerations.

The results of the simulation can only be summarized here (see [1]); but showed the inertial forces to remain at below 3% of the intended force feedback level even when the velocities and accelerations of the handgrip were at their peak representative values. The relatively small level of inertial force disturbance is to be expected as this along with high stiffness are representative properties of parallel mechanisms.

The choice of joystick working volume or that workspace the T-shaped handgrip can be moved within has been based on information found in the literature on other manual controllers which showed no debilitation using workspaces in the vicinity of a 12 inch cube [2]. Since an initial decision to use 18 inch stroke air cylinders as the compressive actuators had been made, the resulting approximate workspace of an 18 inch sphere has been deemed acceptable.

JOYSTICK DESIGN OBJECTIVES

joystick workspace	18 inch diameter sphere
dexterous workspace	10 inch diameter sphere
incremental translation	.13 inch
orientational range	180 degrees (3 axis)
incremental orientation	1.1 degrees
force feedback range	0 to 10 lbf
torque feedback range	0 to 24 inch-pound

In figure 2 below, we see the annotated schematic of the Texas 9-string kinesthetic joystick. The two upper (vertical) planes of the joystick frame are constructed of clear acrylic in order to not obscure the operators' vision as the robot work area is in front of the joystick and slightly to the right.

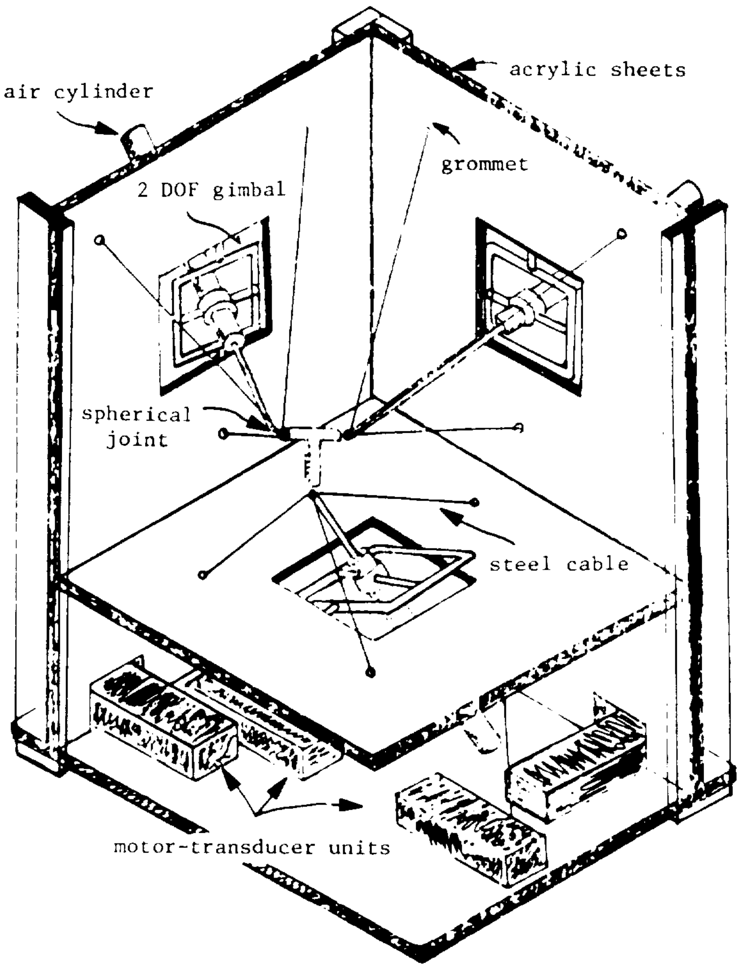


Figure 2: Texas 9-String Kinesthetic Joystick

4. Implementation

The handgrip position and orientation of the joystick is calculated in the dedicated microprocessor from the lengths of the 9 steel strings. The string lengths are measured by custom made rotary potentiometers. The microprocessor then maps the handgrip position and orientation to the end-effector of the robot. This mapping exists in software and can be scaled by the operator. Simultaneously, the robot end-effector load state is measured and mapped to the handgrip of the 9 string joystick.

A general transformation is used to map from the 6-dimensional force space of the robot end-effector to the 9-dimensional force space of the joystick. The 9-dimensional force space of the joystick is represented by the 9 independent servoed cable tensions, which can only act in tension, thus requiring the 3 constant forces of the compressive actuators in order to generate an arbitrary force at each connection of the 'T' shaped hand-grip. Force Feedback is accomplished by holding the three air-cylinders at constant pressure and then controlling the tensions in the strings via current controlled servo-motors. As the problem of determining the cable tensions is underconstrained, an optimization has been performed to minimize the sum of the squares of the cable tensions. In mathematical form this is known as the pseudo-inverse of a non-square matrix.

The dedicated microprocessor is a DEC Microvax II. The Microvax II computer represents the computational base for the application of the transformations, filtering, communications, and control activities present in this system. With a sufficiently fast protocol for communications to and from the robot controller, the Microvax II would represent real-time computing power for this level of task.

Each DC motor-transducer unit consists of a high-resolution .1% linearity rotary potentiometer attached to a spool wound with a steel cable. As the cable or 'string' is unwound from the spool, its length is proportional to the potentiometer resistance. Analog voltage measurement across the potentiometer can then be calibrated to the string length. The voltage readings from the transducers are continuous values which are converted via the DEC ADV11-C analog-to-digital converter board to digital information for the computer. Each transducer is driven by a brushless DC servo-motor to control the tension in the string. The Harowe motors are DC permanent magnet and brushless servo-motors with a stall torque rating of 35 pound-in.

The force feedback control signals from the Microvax are converted by the DEC AAV11-C digital-to-analog converter to the continuous voltage signals needed for the Benton SC-10 servo controllers. The servo-controllers operate in a current regulating manner to drive the DC motors. The use of a current control scheme over that of a voltage controlled one is critical to the performance of the 9 string joystick. The motor torque is proportional to the applied current. If the motors are powered in a voltage controlled mode then a back EMF forms which reduces the motor armature current. This results in the reduction of motor torque due to the circuit dynamics. This effect is equivalent to a mechanical damping. The magnitude of system dynamics is large enough to interfere with the operator's sensing of force feedback. In figure 3 below, the complete U.T. telerobotic testbed is shown.

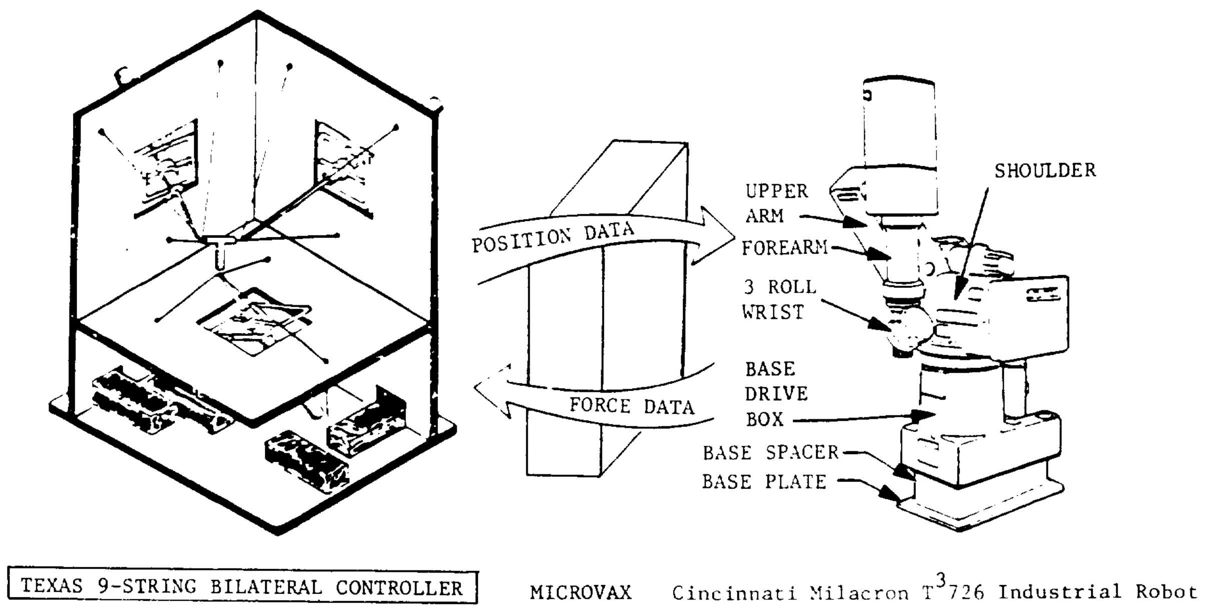


Figure 3: U.T. Telerobotic Test Facility

The three air cylinders represent the prismatic joints in the legs of the Stewart-platform parallel mechanism. The compressive actuators are Benton B-120 single ended, 18 inch stroke air cylinders. Each air cylinder is supported by an adjustable bracket to the center of a 2 DOF gimbal or hook's joint. The end of each air cylinder shaft is connected to a 3 DOF spherical joint, composed of a steel universal joint with a ball bearing at each end. One spherical joint is connected to the end of each one of the three arms of the 'T' shaped aluminum handgrip. At the connection of each spherical joint to an air cylinder shaft, three steel cables are attached. The intersection of the three strings with the air cylinder axis represents the point where the force at that arm of the handgrip is generated.

Consequently, an arbitrary force vector (magnitude bounded) can be applied to each arm of the 'T' shaped handgrip. Each force vector is limited by the applied maximum string tension. The three triad force vectors sum to produce the desired force and torque state at the center of the joystick grip.

Software has been provided by the manufacturer to interface the Microvax II to the controller of the Cincinnati-Milacron T_3-726. The Cincinnati-Milacron Inc. (CMI) host software is responsible for a time lag in the communications rate. The CMI software uses a non-real time protocol system known as DDCMP.

The load state at the robot end-effector is sensed by a commercially available force/torque sensor. The sensor is a Lord corporation model 15/50 load cell. The model 15/50 is mounted to the wrist of the robot, and a connection is provided to affix a Telerobotics International EP 100/30 robot gripper. The force sensor and the robot gripper are both driven by software implemented on the Microvax II. The robot end-effector is utilized by the telerobotic system operator via an on-off control button box. The button box is small and designed to be held in one hand by the operator to control the robot end-effector, while the other hand is in bilateral control with the robot arm.

The fully integrated telerobotic system is represented in figure 4 by a signal flow chart. After the system undergoes the startup procedures the T_3-726 is placed in a remote mode in which the Microvax computer becomes a peripheral to the robot's controller. The operator then controls the system at two levels. In the first level, he must enter instructions into a menu-driven interactive routine on the computer terminal. At this stage, the operator can determine which control mode is desired. The different options available include; position-only control, resolved motion rate control, and kinesthetic control. In addition, the operator has the ability via the menu-driven terminal display to modify the spatial correspondence between the robot and the 9-string joystick. The operator can rereference the fixed joystick workspace to a new region of the robots workspace, he can scale the position and force mappings between the robot and the joystick either up or down from unity, and he can perform a smoothing operation on the position data to remove jitter from the robot's motion.

In the second level, the operator has placed the system software into control mode. The telerobotic system is then active. The operator by moving the handgrip within the limitations of the joystick workspace performs either a proportional move or sets a proportional velocity into effect for the robot end-effector. If the kinesthetic control loop is active, when the wrist of the robot is loaded by forces and torques, a scaled equivalent force and torque state at the operator's hand is generated.

Testing of the DC motors has shown that a stall force greater than 12 pounds for several minutes yields high motor temperatures and declining performance. Therefore, the system is operated in the kinesthetic mode with a maximum string force of 12 pounds. From our design optimization procedure we calculated the pnuematic system set point and the maximum available force reflection for each triad of the joystick. The air cylinder constant force has then been set to $|F|=14.83$ pounds or 12.06 psig. The result is a maximum force feedback signal of 3.25 pounds at each arm of the 'T' shaped handgrip without affecting the isotropic nature of the force reflection. This corresponds to a range of force/torque feedback for the handgrip from a pure maximum force capability of 9.75 pounds in any direction, to a pure maximum torque capability of 43 in-pounds about any axis.

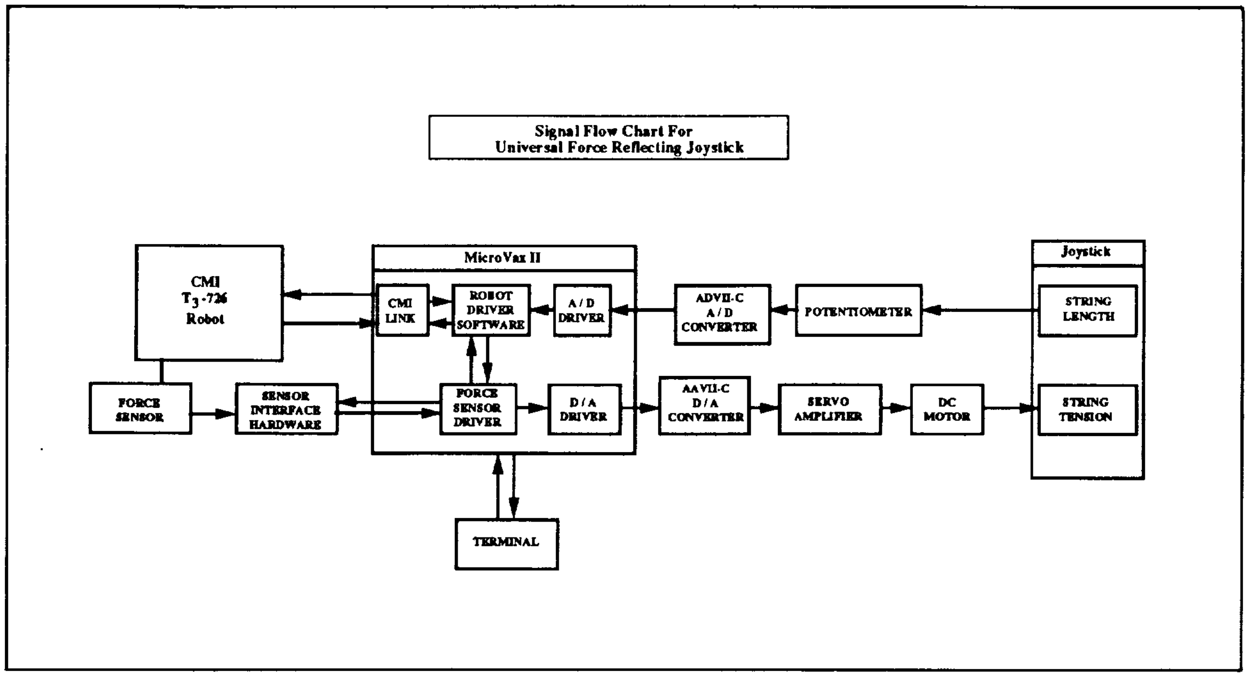


Figure 4: Testbed System Communications

5. Conclusions

Current methods of control use limited, corrupted, or inappropriately coded information to the human operator as well as hardware and software of insufficient power, generality, and dexterity to exploit the full capacity of telepresence.

The uniqueness of the 9 string joystick's geometry, the portability of its software, and the kinesthetic attractiveness of its operation make this man-machine interface a break with past engineering work in hand controllers and an excellent analysis tool for R&D.

The Texas telerobotics testbed after completion has been evaluated and found to be functional, yet showing significant detractions. Indicating the importance and difficulty of achieving real-time
telepresence in telemanipulation. The most crucial detraction to performance is the existence of a high level of coulomb friction in the joystick mechanism. The effect is concentrated in the sliding joint of the air cylinders. The implemention of a pnuematic system resulted in a masked force feedback, which blocks the joystick's transparency to bilateral communication flow. The friction force also had the effect of making small precise motions difficult.

The high level of friction force in the pnuematic system also had the effect of obscuring the importance of friction from the motor-transducer units, and the inertial forces incurred in moving the joystick. A number of alternatives to a passive pnuematic system were considered such as motorized capstan, linear induction motor, and a linear mechanism employing a constant force spring. Another significant limitation to system performance is the slow update rate, or system cycle time. The protocol that allows information from the T_3-726 controller to be sent to, or received from the Microvax II computer is not sufficiently fast to fulfill our design goal of achieving a 30 Hz run-time bilateral mapping.

The resulting update rate of 9-10 Hz represents a significant reduction in performance. The operator becomes cognitive of this time delay during precise positioning. Also, the time delay produces increasingly jerky motion in the manipulator as the distance between subsequent sampling points grows.

In its present form the Texas 9-string kinesthetic joystick represents a proof-of-concept for a universal, parallel 6 DOF force reflecting manual controller for telerobotics. It does not yet achieve the demanding characteristics of transparency to information flow, and the system does not yet achieve the goal of telepresence. Currently, it is not expected to pursue improvement in the Texas 9-string joystick; but rather to use it for research into the issues necessary to design the next generation of man-machine interfaces. Current thinking for next generation interfaces include advanced hand-controllers based on redundant and hybrid (parallel and serial mechanical architecture) design as discussed by Sklar in [6].

Primarily, the research use for the Texas 9-string joystick is in such areas as human factors engineering. Results from that work would then push the design of man-machine interfaces based on a quantified understanding of issues such as joystick inertia, friction, cycle time, work volume, etc.

References

1. Lindemann, R., "Construction and Demonstration of a 9-String, 6 DOF Force Reflecting Joystick for Telemanipulation," Master's Thesis, The University of Texas at Austin; December, 1987.

2. Tesar, D. and Lipkin, H., "Assessment for the Man-Machine Interface between the Human Operator and the Robotic Manipulator," Report to DOE and NSF, University of Florida; October, 1979.

3. Lipkin, H.,"Kinematic Control of a Robotic Manipulator with a Unilateral Manual Controller," Master's Thesis, University of Florida; May, 1983.

4. Agronin, M., "The Design and Software Formulation of a 9-string, 6 DOF Joystick for Telemanipulation," Master's Thesis, The University of Texas at Austin; August, 1986.

5. Freeman, R.A. and Tesar, D., "Kinematic and Dynamic Modeling, Analysis and Control of Robotic Mechanisms (via Generalized Coordinate Transformation)," Ph.D. Dissertation, University of Florida; May, 1987.

6. Sklar, M., "The Analysis of Hybrid Parallel/Serial Robotic Manipulators," Master's Thesis, University of Florida; April, 1984.

N90-2[illegible]7

RESPONSE TO REFLECTED-FORCE FEEDBACK TO FINGERS IN TELEOPERATIONS

P. H. Sutter, J. C. Iatridis, N. V. Thakor*
Franklin and Marshall College
Lancaster, PA 17604-3003
Dept. of Biomedical Engineering
Johns Hopkins University
Baltimore, MD 21205

Abstract

Reflected-force feedback is an important aspect of teleoperations. Our objective is to determine the ability of the human operator to respond to that force. The present study simulates telerobotics operation by computer control of a motor-driven device with capabilities for programmable force feedback and force measurement. We have developed a computer-controlled motor drive that provides forces against the fingers as well as (angular) position control. A load cell moves in a circular arc as it is pushed by a finger and measures reaction forces on the finger. The force exerted by the finger on the load cell and the angular position are digitized and recorded as a function of time by the computer. We investigated flexure forces of the index, long and ring fingers of the human hand in opposition to the motor driven load cell. We present results of the following experiments: 1) Exertion of maximum finger force as a function of angle; 2) Exertion of target finger force against a computer controlled force; 3) Test of the ability to move to a target force against a force that is a function of position.

Averaged over ten individuals, the maximum force that could be exerted by the index or long finger is about 50 Newtons, while that of the ring finger is about 40 Newtons. From our tests of the ability of a subject to exert a target force, we conclude that reflected-force feedback can be achieved with the direct kinesthetic perception of force without the use of tactile or visual clues.

1. Introduction

Space telerobotic systems[1,2] have many aspects, ranging from quite direct control by an operator in a master-slave configuration to much more autonomous control, which may be particularly useful when signal transmission times are large. We are concerned with the former, particularly the response of the human operator. Such a system might be used for work on a space station. An operator of a telerobotic system must be supplied with information on the status of the controlled device, perhaps an arm or manipulator. This information can be in the form of visual displays, audible or tactile signals, or reflected force feedback. Since knowledge of the forces experienced by the driven telerobotic system, as well as the position information, are of fundamental importance to the operator, it is useful to develop systems that feed that information back to the operator in as natural a way as possible. Our study is on the ability of a subject to respond to simple force signals to the fingers.

Each finger[3] has a metacarpal bone, which is inside the hand , and proximal, middle and distal phalanges. In our experiments, the metacarpalphalangeal (MCP) joint rotated in flexure. At high angles of rotation of the finger with respect to the straight ahead direction, the proximal interphalangeal (PIP) joint also rotated. The flexure forces are transmitted by tendons from muscles in the anterior forearm. The physiological basis for the perception of position and muscular force is discussed in review articles[4,5] on proprioception and kinesthetic sensations.

Reflected-force feedback to the fingers of an operator will be effective, if the operator can sense force levels in a normal way. This not only provides a more natural mode of perception by the operator but also frees up the other senses, for example vision, for other information gathering. In our experiments on the ability of a subject to sense forces, the computer provides a functional dependence of force on position that simulates the forces that might be felt by the slave unit in teleoperations in space.

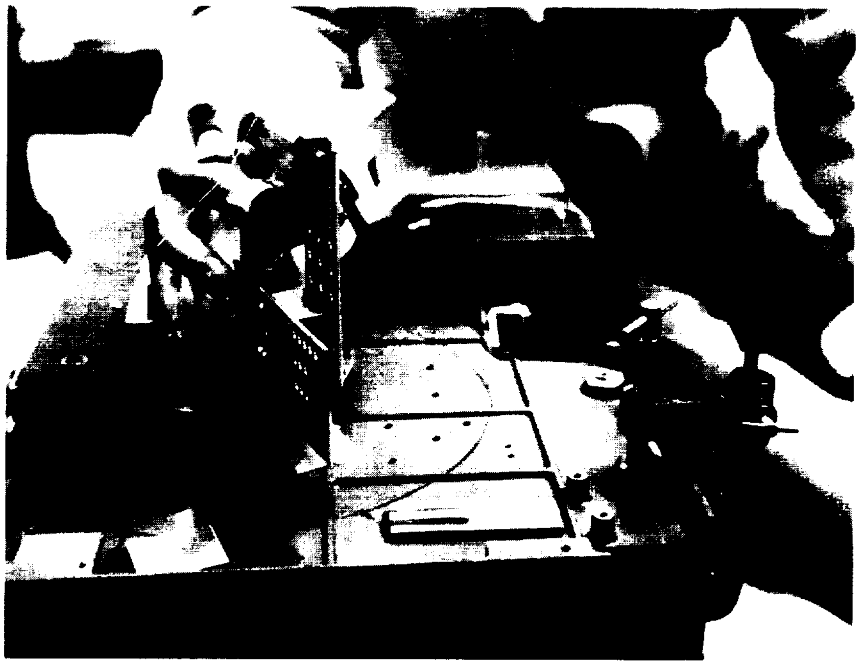
2. Instrument Design

This study simulated telerobotics operation with a computer (IBM AT) which controls a dc motor (Galil control board) to provide angular position and torque control. Metrabyte DAS8 and Tecmar LabMaster boards were used for data acquisition. A semiconductor load cell was used to measure the force exerted by a finger on a computer controlled motor drive system that carried the load cell along a circular arc in a horizontal plane. Angular position was measured with an optical encoder and a potentiometer. The digitized force and position information was recorded in the computer as a function of time. Flexure forces exerted individually by the index, ring and long fingers of the human hand were measured as the finger pushed against the motor driven load cell, with the subject's arm and wrist stationary. The lower arm was horizontal, with the wrist and hand extended straight ahead. The upper arm was vertical. The apparatus is shown in Figure 1.

A one inch diameter brass "pad" was placed on the distal (terminal) phalanx of the finger being tested. It was soft on the side facing the finger and had a semicircular cross-section groove on the side that mated with the load cell holder. The straight groove in the pad bore against a steel rod on the outside of the load cell holder. Thus a force could be applied to the load cell with some freedom of motion of the finger with respect to the load cell. The steel rod was attached to a pivoted plate (see Figure 1) that held a block of teflon that actually pushed on the semiconductor load cell and assured that the forces exerted on the cell surface were perpendicular and uniform. The voltage output of the load cell was a linear function of the force. The voltage was digitized and converted to the force in Newtons by use of a gravity calibration method. In some experiments, a two inch long brass "splint" was used to restrict rotation to the MCP joint only. This mated with the load cell in the same way as the circular "pad".

Figure 1: APPARATUS

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH



3. Experimental Results

3.1 Maximum finger flexion force:

We measured the maximum flexure force of the long, index and ring fingers of ten subjects. In one procedure, the motor drive swept the arm at a constant rate of 6.75 degrees per second for a time of 20.4 seconds. Data was obtained and recorded every 10 milliseconds. The subject exerted maximum force against the load cell with the pad on the terminal phalanx. This force deflected the arm slightly until the motor drive feedback torque became large enough so that the position was determined by the angle-sweep commands from the computer. The motor drive system was adequate to oppose the finger force and the angle increased as a linear function of time. Data from a single run is graphed in Figure 6 in the appendix. That data yields force and angle as a function of time after reduction based on the calibration of the apparatus. Figure 2 shows the maximum flexure force exerted by a finger as a function of angle for one individual. The three lines are for the index, long and ring fingers of this subject.

The finger angle is 0° when the finger is extended straight along a line through the forearm and hand. The subject's wrist was straight throughout the experiments . At first, only the metacarpophalangeal (MCP) joint rotates, but after about 70° the proximal interphalangeal (PIP) joint also rotates. Averaging over many runs and individuals yields a curve that is rather flat out to about 90° and then declines with increasing angle as shown in Figure 3.

FIGURE 2: MAXIMUM FINGER FORCE VS ANGLE

For One Individual

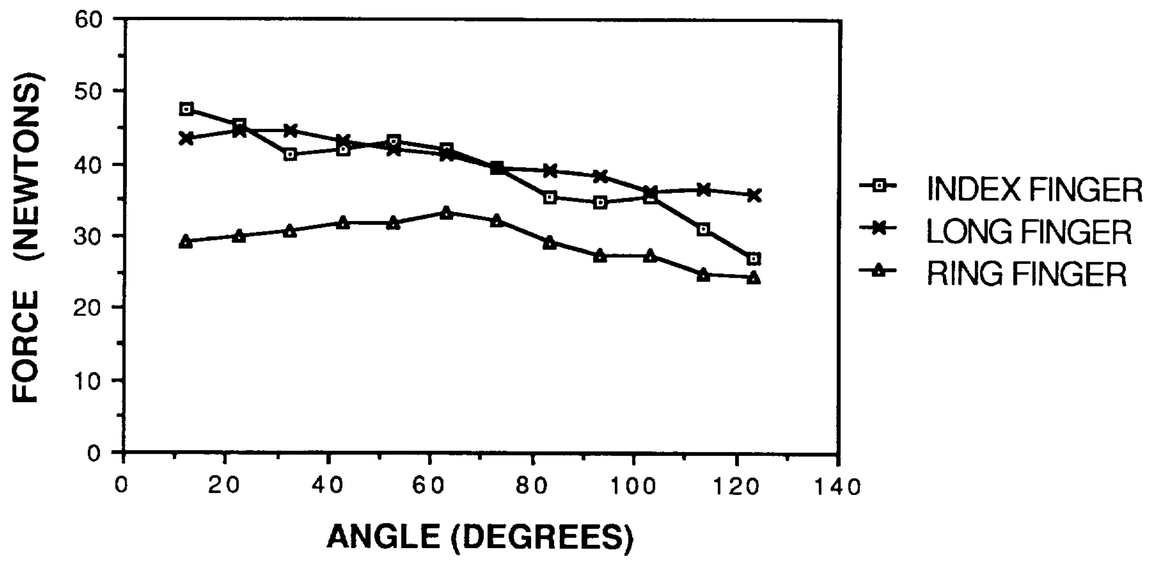
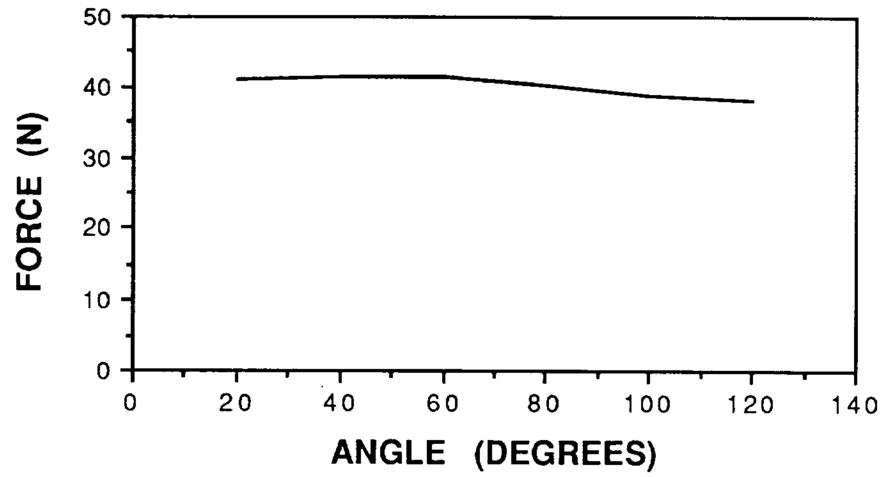


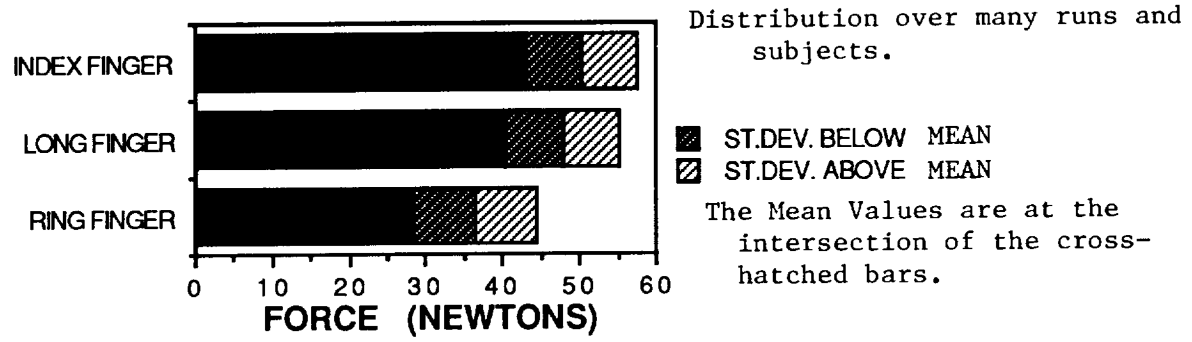
FIGURE 3: AVERAGE OF MAXIMUM FINGER FORCE

Averaging over many runs and subjects



The distribution over individuals is shown in Figure 4 in which the cross hatched segments of the bars represent a standard deviation above and below the mean, which is at the intersection of the cross hatched regions. For most individuals the index and ring fingers can exert approximately the same maximum force while the ring finger is weaker.

FIGURE 4: DISTRIBUTION OF MAXIMUM FORCE



Distribution over many runs and subjects.

The Mean Values are at the intersection of the cross-hatched bars.

3.2 Pulsed maximum force:

A second mode of testing was to have the computer sweep the angle range over a time interval of 40.8 seconds with the subject intermittently applying a maximum force. Data was obtained and recorded every 20 milliseconds. Typically, a signal tone was on for 3 seconds and then off for 3 seconds, etc.; and the subject exerted a maximum force when the tone was on and relaxed when it was off. This reduced the effect of muscle fatigue. The force versus angle was similar to the above graphs, with somewhat less decline at high angles. The higher angles occur at longer times so that part (but not all) of the decline at high angles in Section 3.1 may be due to fatigue.

In order to prevent rotation of the PIP joint, we also used a splint on the fingers of some subjects. The metal splint replaced the circular pad in bearing against the load cell device. The results were similar to those discussed above except that the finger rotation was limited to about 80°.

3.3 Target finger force:

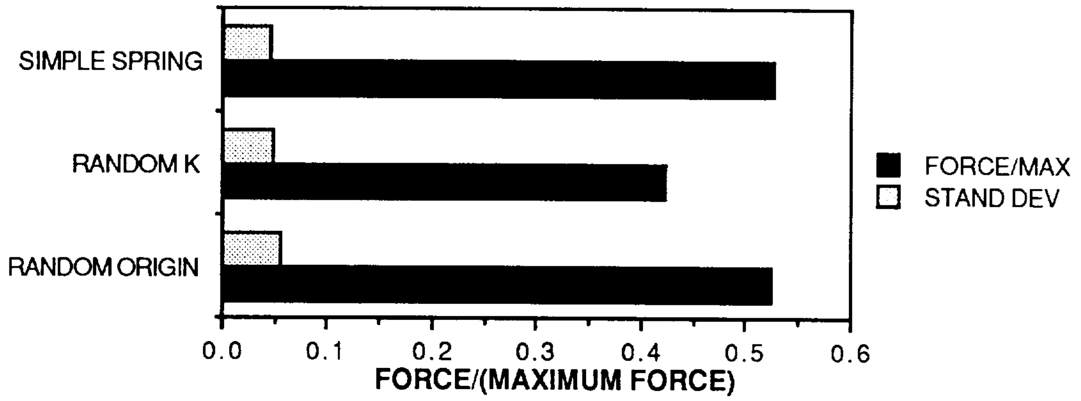
A series of experiments were designed to determine how well a subject could sense target force levels. In one set of experiments, the computer simulated a spring-like force. That is the restoring force was proportional to the distance from an origin. In this series, a subject heard a three second tone about once every six seconds and was told to push with the target force while the tone was on. The target force was half-maximum, quarter-maximum or eighth-maximum. A typical run consisted of the subject pushing at full maximum during the first tone and then, on the remaining tones, pushing at half maximum force. The repeatability of the force pulses was of primary concern, while the relationship to the maximum force was secondary. In the simple spring simulation, the computer selects a spring force constant and uses an origin near an angle of

0°. As the subject pushes against the load cell, the load cell moves through an angle while providing a restoring force that is proportional to the angular displacement from the origin. The finger might swing through an angle of forty degrees, for example, before the subject sensed the target force and held that force. The spring force constant and origin remain fixed during a single run. In each such run, the subject attempted to return to a target force 6 times. In a typical set of eight such runs, the ratio of the actual force to the maximum force was 0.46 ± .04 while the target was 0.50. The scatter of the 6 force pulses within a run was 7% ± 3%., which is .032± .014 with respect to the maximum force. For this simple spring simulation, the repeated return to the target force occurs at the same angular position during a run. Although the subject did not use vision to return to this repeated position, a proprioceptive sense could have been used rather than a perception of the force exerted by the finger.

3.4 Target finger force with separation of position and force sensing:

In order to separate the proprioception from the kinesthetic sense of force, we also simulated springs in which the spring constant or origin were randomized within a run. In these runs, the target force occurs at random angular positions during a run. Hence, the subject could not use position clues as a means of returning to the target force. Raw data for a run of this type is graphed in Figure 7 in the appendix. Since the vertical voltage scale is linearly related to the force, this data shows that the subject was able to closely return to the target force in successive attempts. The bar chart in Figure 5 shows the results of a series of runs with the index finger of a subject. Within each run, the computer simulated either a simple spring, a spring with random force constant or a spring with random origin. The long dark bar indicates the average ratio of the exerted force to the maximum force, with the target value being 0.50. The lighter bars show the average standard deviation of the scatter within a run, expressed as a fraction of the maximum force. We conclude that the subject is not relying on a position clue, but rather is correctly judging the force exerted by the finger.

FIGURE 5: HALF-MAX TARGET WITH SIMULATED SPRING



"RANDOM K" means random force constant or stiffness during a run.
"STAND DEV" means the standard deviation of the scatter within a run.

In order to study this force judgement at lower fractions of the maximum finger force, subjects attempted to repeat similar runs with a) a maximum force pulse followed by six attempts at a force of half-maximum; b) a first half-maximum force pulse followed by six quarter-maximum pulses; c) a first quarter-maximum pulse followed by eighth-maximum pulses. Table 1 shows the results of a series of these runs:

Table 1: Pulsed Target Forces Against Spring Simulations

		Mean successive/first		
First force	6 successive forces	Target	Observed	Scatter
Maximum	Half-maximum	0.50	0.48	± .05
Half-Max	Quarter-Max	0.50	0.44	± .05
Quarter-Max	Eighth-Max	0.50	0.34	± .05

The last column of Table 1 is the average scatter (standard deviation) within a run. This is expressed as a ratio to the force in the first pulse. The individual runs within a set included simple spring, random spring constant and random origin with results similar to those given above. The perception of the finger-force appears to be successful even at relatively low force levels.

3.5 Tactile clues:

The possibility that the force perception is simply due to tactile clues should be considered. For this purpose we repeated the runs with the use of the metal splint, which should reduce the pressure on the terminal phalanx and isolate the rotation to the PIP joint. The results were similar as long as the splint did not impede the motion with the least stiff spring simulations. These results are shown in Table 2:

Table 2. Pulsed Target Forces with Splint

		Mean successive/first		
First force	6 successive forces	Target	Observed	Scatter
Maximum	Half-maximum	0.50	0.42	± .05
Half-Max	Quarter-Max	0.50	0.46	± .05
Quarter-Max	Eighth-Max	0.50	0.45	± .04

As another way to reduce the tactile sensation of force, we immersed the terminal phalanx of the finger in ice for about ten minutes before repeating the runs. The finger was re-inserted in the ice for several minutes between runs, each of which took about 40 seconds. The results are given in Table 3:

Table 3: Pulsed Target Force with Numbed Finger
First pulse was maximum, successive six pulses were half-maximum.

	Mean of successive forces/first force			
Spring simulation	No Ice	Scatter	With Ice	Scatter
Fixed force constant	0.46	± .05	0.42	± .04
Random force constant	0.34	± .05	0.35	± .05

The results indicate that numbing the terminal phalanx of the finger, which is in the pad that pushes on the load cell, does not prevent the perception of force. We conclude that this force perception is not an artifact of tactile clues at the finger tips.

4. Conclusions

Based on the results in Section 3.1 and 3.2, it appears that a good design value for the maximum finger force to be exerted by an operator in a telerobotics system is about 40 Newtons for the index and long fingers and about 30 Newtons for the ring finger. These forces can be maintained out to angular excursions beyond 90 degrees. Averaging over runs and individuals, the maximum finger flexure force is nearly independent of angle in this range.

From the results in Section 3.3 and 3.4, we conclude that a subject can successfully exert a target force against a restoring force which is a function of position. The ability to repeatedly sense the target force level did not require visual or position information and did not require training.Furthermore, the perception of force was effective down to one-eighth of the maximum force. From the experiments described in Section 3.5, we conclude that the perception of the force exerted by a finger is not dependent on tactile clues.

This study is encouraging with respect to the use of direct operator perception of force in reflected-force feedback telerobotic devices. This can be useful in developing space telerobotics because it provides natural perception of force by the operator rather that a reliance on visual displays, for example, which might then be used for other information.

References

1. Antal K. Bejczy "Sensors, Controls, and Man-machine Interface for Advanced Teleoperation", Science 208, 1327 (1980)
2. R.L.Smith and D.J.Gillan, "Telerobotics in Space: Some Human Factor Considerations", ISA, 307-314, 1987
3. J.C.Becker, N.V.Thakor and K.G.Gruben, "A Study of Human Hand Tendon Kinematics with Applications to Robot Hand Design", IEEE Conference on Robotics and Automation, 1-6, 1986
4. W.J. Williams, "A Systems-Oriented Evaluation of the Role of Joint Receptors and other Afferents in Position and Motion Sense", CRC Critical Review in Biomedical Engineering, Dec.1981
5. P.B.C.Matthews, "Where Does Sherrington's 'Muscular Sense' Originate? Muscles, Joints, Corollary Discharges?",Ann. Rev. Neuroscience 5, 189, 1982.

Acknowledgement:

This work was supported by the Johns Hopkins University -Dana Foundation Sabbatical Program at the Department of Biomedical Engineering of Johns Hopkins University.

APPENDIX

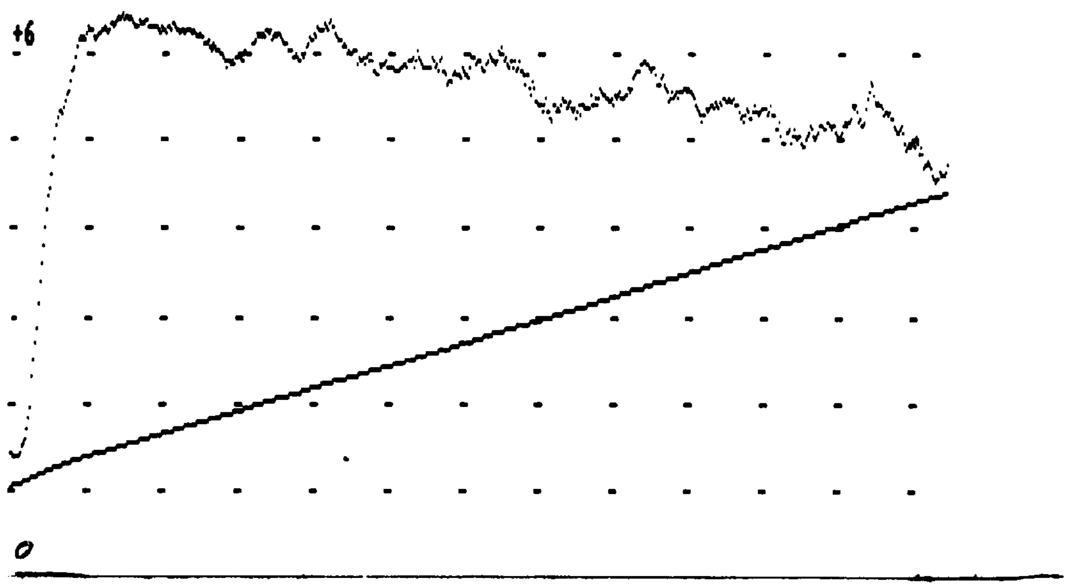


FIGURE 6 Raw Data for Force and Position versus Time.
The load cell voltage is plotted vertically with a scale of one volt per dot. 6 volts corresponds to a force of 42 Newtons. The dots are 1.5 seconds apart horizontally. The dotted line is position versus time with the motor drive providing a constant sweep rate under computer control at about 10^{o} per dot.

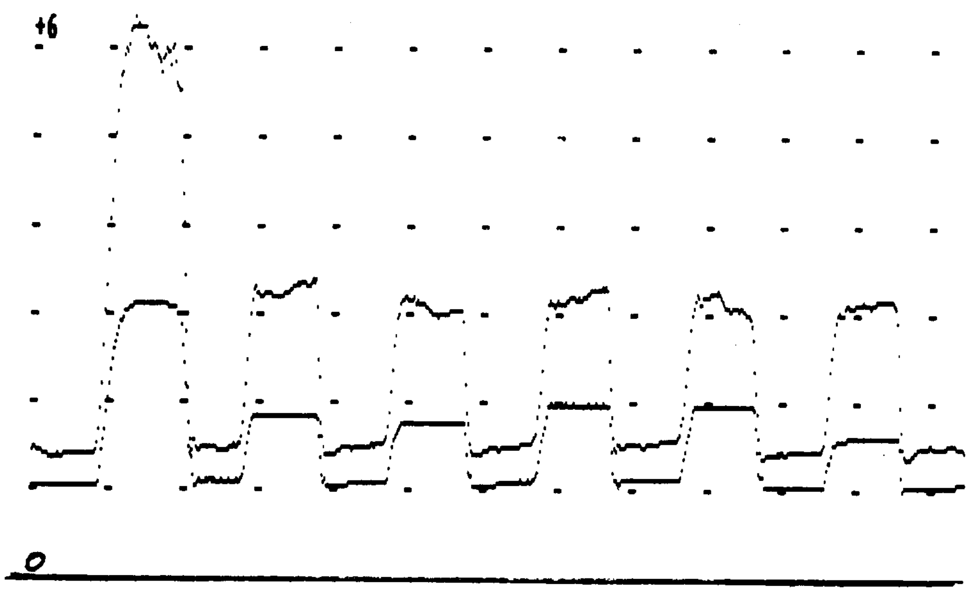


FIGURE 7 Raw Data for Target Finger Force Against a Simulated Spring. This simulation is for a spring with a random spring constant so that the stiffness of the spring changes randomly at each attempt. The dots are three seconds apart horizontally. The subject pushes for about three seconds and then relaxes for about three seconds. The first pulse target was maximum force and the load cell voltage reached about 6 volts which corresponds to 42 Newtons. The lower plateau at that time is the position data. On the following attempts the target force was half-maximum. Notice that the force levels are similar but that the lower plateau for position is variable. This is because the system behaved like a spring with a random stiffness. The subject pushes out to the same force level even though it occurs at different positions.

N90-29338

THE JAU-JPL ANTHROPOMORPHIC TELEROBOT

Bruno M. Jau

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

ABSTRACT

This paper describes work in progress on the new anthropomorphic telerobot that is currently being built at JPL. The initial robot configuration consists of a seven d.o.f. arm and a sixteen d.o.f. hand, having three fingers and a thumb. The robot has active compliance, enabling subsequent dual arm manipulations. To control the rather complex configuration of this robot, an exoskeleton master arm harness and a glove controller have been built. The controller will be used for teleoperational tasks and as a research tool to efficiently teach the computer controller advanced manipulation techniques.

INTRODUCTION

Current-day robots lack the manipulation capabilities and sensing skills of the human arm-hand system. The development of this telerobot has as its goal to provide limited man-equivalent dual arm manipulation capabilities for remote material handling, servicing and other manipulative tasks that require mechanical dexterity together with real time evaluations of a variety of sensory information.

The initial thrust of this research is to demonstrate advanced manipulation skills at never before achieved levels through teleoperation. The teleoperation mode is used for unstructured tasks in changing environments where preprogramming is almost impossible. It will be very useful for the early space station construction, assembly and contingency tasks: The dual arm robot will serve as dexterous front end to an RMS arm that is located on the space station or at the space shuttle and will perform tasks that previously required EVA astronauts. The robot will be controlled from a remotely located IVA crew station such as the Shuttle's aft flight deck where the operator controls the robot through the exoskeleton controller in a natural and user friendly way by just simply performing the desired manipulations.

Once the initial space station construction phase is accomplished by the late nineties, the station's tasks may become more routine while robot controllers will have become more sophisticated so that limited autonomy of the robot system can be considered. The system can then be upgraded to a telerobot with shared man-machine control due to the active compliance and a variety of sensors that are already built into this robot. This paper, however, focuses primarily on the soon to be completed first master and slave units in teleoperational control mode.

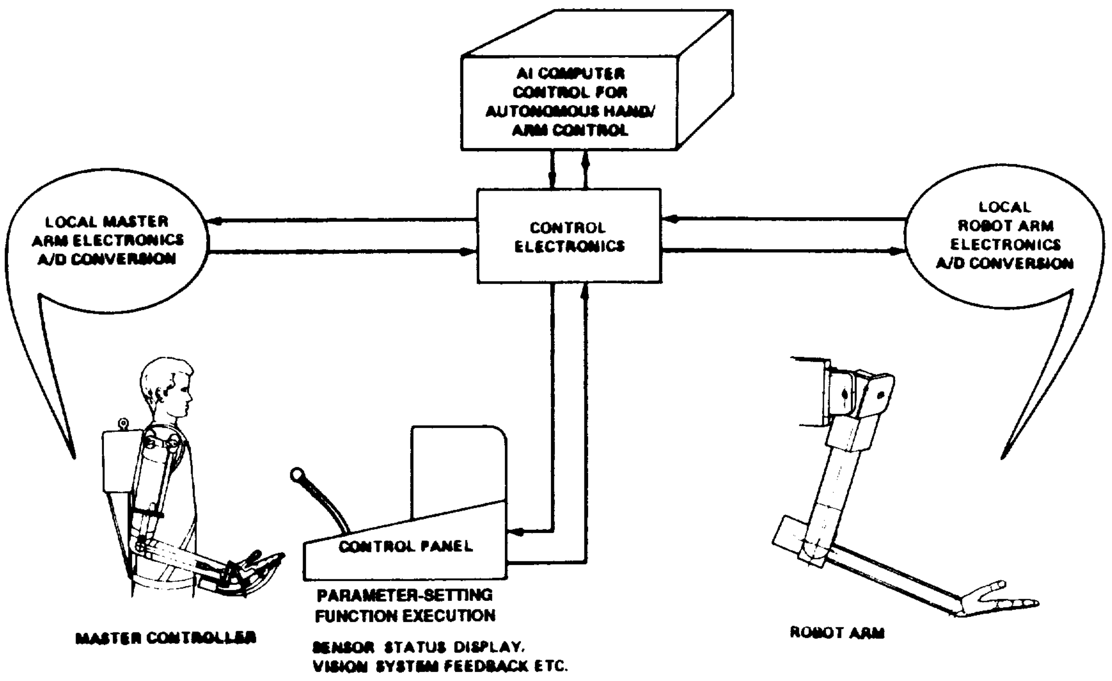

AI COMPUTER CONTROL FOR AUTONOMOUS HAND/ ARM CONTROL
LOCAL MASTER ARM ELECTRONICS A/D CONVERSION
CONTROL ELECTRONICS
LOCAL ROBOT ARM ELECTRONICS A/D CONVERSION
CONTROL PANEL
PARAMETER-SETTING FUNCTION EXECUTION
SENSOR STATUS DISPLAY, VISION SYSTEM FEEDBACK ETC.
MASTER CONTROLLER
ROBOT ARM

Fig.1 The Anthropomorphic Telerobot–Overall System

OVERALL SYSTEM

The initial system configuration is shown in Fig.1. It will consist of

o An anthropomorphic arm-hand robot

o An exoskeleton master arm harness and glove controller

o The computer control electronics

The second phase will be the addition of symmetrical master and slave arms and automation aids. Subsequent developments will incorporate auxiliary equipment and tools, emphasizing tool manipulation skills. Local and global mono and stereo vision and other sensory systems will also be installed. Graphics and predictive displays will be used to aid time delayed operations for robot control from a ground station.

THE ROBOT ARM-HAND

The Robot arm-hand is sketched in Fig. 2 without its auxiliary devices such as cameras. The first arm is currently being constructed. It features the following items:

o A mid-body section that has the two arm attachments on either side and an external attachment fixture at its back. The robot can be picked up by a carrying device such as the space shuttle's RMS arm.

o Seven d.o.f. arms in the exact kinematic configuration as the master arm. An exact one-to-one kinematic correspondence between master and slave arm joints exists.

- The hands have three fingers and a thumb, a palm, the three d.o.f. wrist and extend further through the forearm to the elbow. The finger drive mechanisms are located in the forearm and the wrist drives are located behind the elbow to counterbalance the arm.

- Most joints feature limber joint active compliance control which imitates the human muscle capability to loosen or stiffen the joint. Active compliance is activated through electromechanical devices that are built into the joint mechanisms. Active compliance enables dual arm manipulations, soft grappling where the hand can conform to the object's shape, and it opens the way for sensor-driven control.

- The mechanical hand also has the equivalent kinematic configuration as a glove controller so that the sensed human hand can provide direct teleoperational control, enabling the human hand to convey skilled hand manipulation techniques.

- Position, force and compliance sensors are built into the prototype. Other sensors will be added at later phases.

THE EXOSKELETON ARM-GLOVE CONTROLLER

The device is shown in Fig. 2. It consists of seven d.o.f. arm harnesses and sixteen d.o.f. glove controllers. All joints have force input sensors, position sensors, and are electromechanically backdrivable. One master arm has been built thus far. The arms are suspended on an overhead translation stage to relieve the operator from the weight of the structure. The operator can work in a standing or seated position and is free to move or walk around as far as the weight suspension system allows him to do so.

Arm Harness

- The base of the controller is a backframe that is being strapped to the operator's back. The backframe serves as reference position for subsequent arm joints. Any active or reaction forces are countered at the frame-human body interaction so that there are no external forces acting on the operator or on the controller. A recoil-free master controller is very important in the weightlessness of space.

- Each side of the frame has an attachment for one of the symmetrical arms. Both arms work completely independently. Linear slides are built in between frame and arm support, allowing passive shoulder motions in all three principal directions. They provide freedom for movements for the operator, enabling him, for instance, to raise his shoulder. Further, they provide size adjustments to accommodate different-size operators.

- The shoulder joint consists of three individually backdriven joints whose rotational axes intersect at the location which is concentric with the human shoulder ball joint.

- A two d.o.f. counterbalancing mechanism, mounted on the arm support structure and acting on two of the shoulder joints, effectively counteracts the mass-moment of the arm assembly at any arm position. The operator is thus not burdened by having to support the mechanical arm at extended arm positions and will not fatigue while working with extended arms for lengthy time periods.

- o Two of the shoulder joint drives have mechanical overload release mechanisms built into their gear trains. This feature, which automatically separates the motor drive from the joint in overload situations, may prevent operator injuries and protects the mechanical arm in case of unforeseen circumstances.

- o Telescopic arm sections are provided for both the upper and lower arms, providing adjustment capabilities for different human arm lengths. The arm sections partially encase the operator's arm without significantly limiting his arm motion capabilities.

- o Special strap-on features are provided at two locations each on the upper and the lower arm to provide good motion compliance between the mechanical and the human arm.

- o The wrist also consists of three individually backdriven joints with their motor drives located near the wrist.

Glove Controller

The hand controller is a slip-into device that the operator can wear on his hand, thus the name glove controller. Three fingers and the thumb are instrumented, each finger unit has four d.o.f. The little finger is not being used at present since the slave hand will be four fingered as well.

Metal plates ride on the backside of each finger section. They are connected by linkages which have common pivot points above the finger joints. Each pivot point is the center for a pulley that rotates at equal angles as the finger joint. The pulleys are backdriven according to the slave's actual displacements to provide identical hand configurations for master and slave hands. The finger actuators are located in compact finger-drive packages at and above the elbow, also serving as counterweight for the forearm. The finger actuators are linked to the finger by means of flex cables.

CONTROL

Due to its exoskeleton, anthropomorphic shape, the operator is able to wear the dual arm control harness. He then just simply performs the desired operation manually while monitoring the events at TV screens or by looking through the control station's windows. Position and force feedback are reflected at each joint in the arm and hand, thus providing a sense of actually operating out there. The master and slave arms are linked by high-speed optical data transmission lines with communication rates of over 1000 Hz. Due to the matching kinematic configuration between master and slave arms, no time-delaying coordinate transformations have to be performed, which results in excellent feedback and response rates.

Compliance control is automatic: If the operator exerts a slight force to move a joint, the slave will respond and feeds the actual motion back to the master controller. If the slave joint is prevented from moving, the master controller will not respond either. Should the operator continue to exert a force to the controller, the controller still senses the operator's force input and produces a proportional force at the corresponding slave joints while at the same time stiffening its internal joint stiffness mechanisms. Position and force control are thus automatically regulated in a human like fashion without the operator having to switch from one mode to another.

SPACE OPERATIONS

The system has been designed with space applications in mind and could be readied for an early space experiment prior to the space station's construction. Fig. 2 shows the artist's concept of the space robot configuration. Employing this anthropomorphic system in space has considerable advantages. Some of them are listed below:

- o It is designed as dexterous front end to the standard RMS arm.
- o The System Hardware can be stowed in the shuttle's cargo bay. The slave will be picked up by the RMS arm. The master could be brought through the tunnel to the flight deck.
- o The dexterous hand can plug in tethering devices and electronic connectors, including its own electrical connections to the RMS arm.
- o The master requires no independent fixed space or attachments in the aft flight deck.
- o The operator wears the master on his back, eliminating disturbing recoiling effects.
- o The operator retains his mobility and by freezing the slave is free to manipulate the RMS controls.
- o One operator can perform a variety of EVA manipulation tasks that previously required two EVA astronauts and one RMS operator.
- o No EVA life support system is needed. No decompression time is required for the astronauts. No mandatory three-day waiting period for EVA operations is required at the beginning of a mission.
- o The system can quickly be deployed in contingency and emergency situations.

SUMMARY

The new anthropomorphic telerobot system was discussed. It will consist of dual robot arm-hands and an exoskeleton dual arm-glove controller. The hands have a thumb and three fingers. The robot arm-hand has active compliance which is essential for advanced dexterous robot manipulations. The control harness allows the operator to command the task by simply performing the desired manipulations. Advantages of this system for space applications were mentioned.

GLOSSARY

EVA:	Extra Vehicular Activities:	Space Walking Astronauts
IVA:	Intra Vehicular Activities:	Astronauts commanding the events from an environmentally controlled command post
RMS:	Remote Manipulator System:	The space shuttle's arm

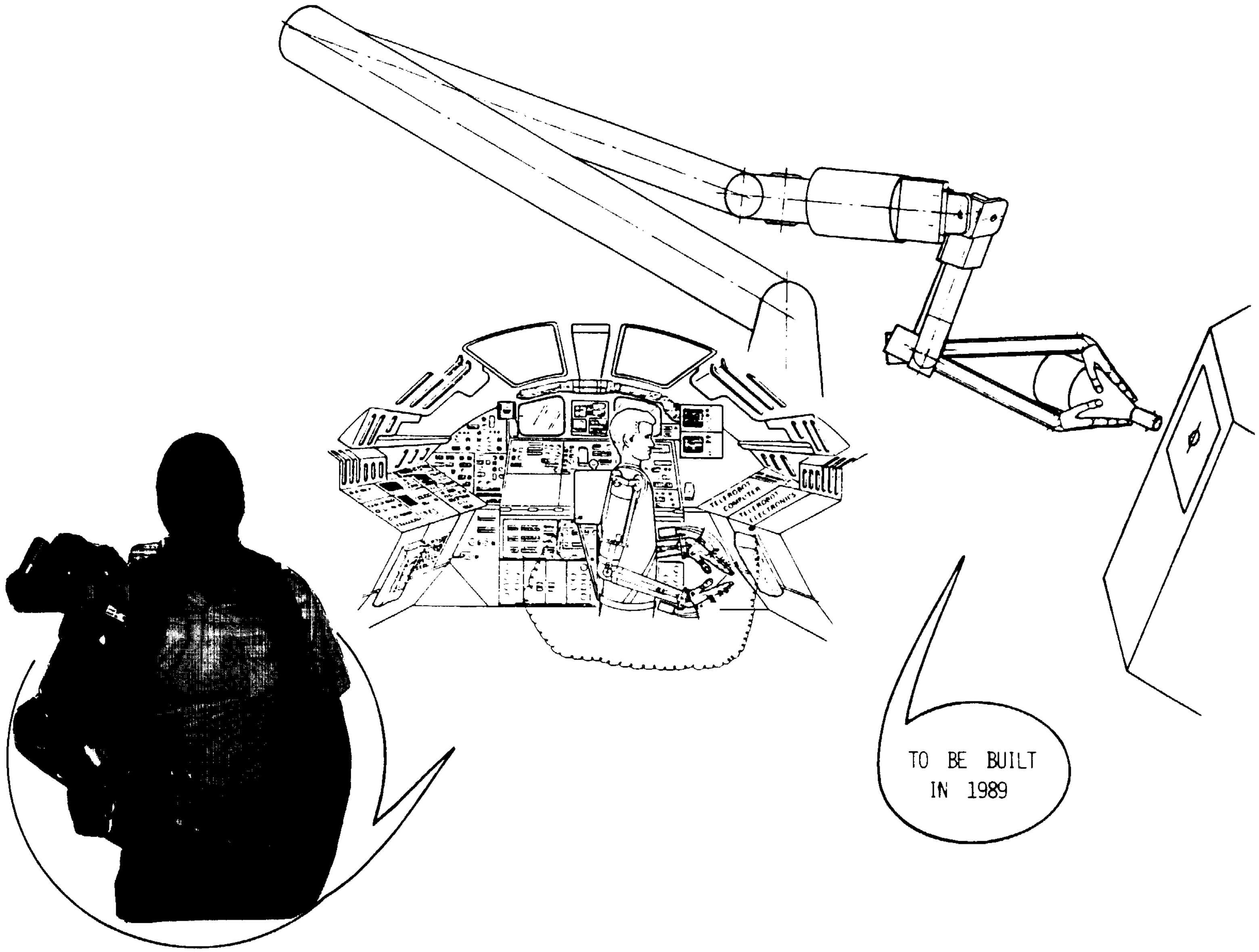


Fig. 2 Anthropomorphic Telerobot in Space Configuration

N90-29889

A PROCEDURE CONCEPT FOR LOCAL REFLEX CONTROL OF GRASPING

Paolo Fiorini & Jeffrey Chang
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91109

Abstract

This paper proposes an architecture for the control of robotic devices, and in particular of anthropomorphic hands, characterized by a hierarchical structure in which every level of the architecture contains data and control function with varying degree of abstraction. Bottom levels of the hierarchy interface directly with sensors and actuators, and process raw data and motor commands. Higher levels perform more symbolic types of tasks, such as application of boolean rules and general planning operations. Layers implementation has to be consistent with the type of operation and its requirements for real time control. In the paper we propose to implement the rule level with a Boolean Artificial Neural Network characterized by a response time sufficient for producing reflex corrective action at the actuator level.

1. Introduction

The set of tools available to robotics researchers to build grasping strategies includes path planners, many types of control algorithms, and sensor data fusion techniques. Many authors have proposed different architectures for the integration of these tools to assure a smooth flow of information from sensing to central computing and back to actuation. A proposed system is based on **Logical Sensor Specification** [6] and consists of software layers between physical sensors, actuators and the central processor, each one of them able to perform some local processing on sensor data and to directly modify commands from the top level to the actuators. A proposal also aimed at reducing the computational load of a controller is that of **Reflex Control** [1] in which a limited number of actions are carried on by the mechanical hardware to react to specified external situations. Other authors have proposed **Expert Systems** at various levels of a layered architecture for hand control. In [15] an expert system for configuring grasp postures is proposed, and this is integrated in [9] with an **Artificial Neural Network** for learning the relations between postures and object shapes. In [7] a **Learning Expert System** is presented for the discovery and refinement of control skills for fine manipulation tasks. These proposals represent efforts to improve performance and applicability of Control Theory and can be associated with the research in **Intelligent Control** [10],[13],[14], whose main effort is the integration of Control Theory, Artificial Intelligence and Operation Research in a homogeneous structure capable of autonomous reasoning and control.

From this brief overview, it is clear that architectures based on hierarchies of processing stages are undergoing extensive study, but present results do not take full advantage of the possibilities offered by this approach. For example, reflex control alone is not sufficient to handle a large number of operating conditions, and expert systems are too slow for a direct interface with a real time process. An efficient **Intelligent Controller**, must assure immediate reactions to unexpected external conditions and thus bypass the long processing time of the higher levels. A fast reaction would compensate for the different execution times of planning and control, and fill in possible voids in the command stream. This can be of great usefulness in the case of space servicing with time delay, when the remote operator cannot provide immediate feedback, and also to free some of astronauts time.

2. Hierarchical Controller

The needs described in the above paragraphs can be satisfied with hierarchical architectures and with circuital implementations of the various layers, that can assure better flexibility and performance than current systems. This proposed control architecture consists of three levels: a planner, a rule base and the actuator controller. Each level can be composed of different layers depending on the particular application, and on the requirement of data fusion. In the implementation that we propose, each level consists of a single layer, and the rule base is included between planner and actuator controller, to have response time and amount of knowledge intermediate between those of the other two. The purpose of this stage is to store expected relations among single feedback signals or subsets of them, and to use the results to understand the evolution of a grasping task. In particular this design allows for a flexible reflex control in various grasping tasks and permits the actuator controller to determine autonomously the best reaction to a given pattern of feedback signals. Both planner and actuator controllers use feedback signals and rules output to start the generation of a new plan and to modify actuator trajectory. When a new strategy is generated by the planner, the associated rules are blended with the current rule base, to assure a smooth transition between plans.

This architecture requires several additions to the standard implementations of planner and actuator controllers. The first one, in fact, has to generate the set of conditions that qualify the task evolution, while the second has to store a set of alternate trajectories. The rule base processes the feedback signals and generates a set of boolean variables, applies the rules supplied by the planner to these variables and determines the current evolution of the task. When a replanning occurs, it assures consistency in its rule base. In this implementation, two basic assumptions have been made: that a small set of actuator trajectories can cover most of grasping situations, and that the evolution of a grasping task can be represented by logical conditions. In this case, the rule base is an adaptive boolean network in which sensor conditions are stored as if-then rules expressed as boolean conjunctions. A range detector converts data from the sensors into logical levels, and the adaptation mechanism supervises the loading of a new plan.

The complete system will include two boolean networks, managed by an adaptation unit, to ensure that adaptation does not interfere with the correct processing of the feedback signals. When the backup network is successfully updated, the adaptation controller will switch it on line and will start updating the other one. Figure 1 is a block diagram of the proposed architecture of the complete system. In the following paragraphs, the justification for such an approach will be presented, together with the description of functions and implementation of the rule base.

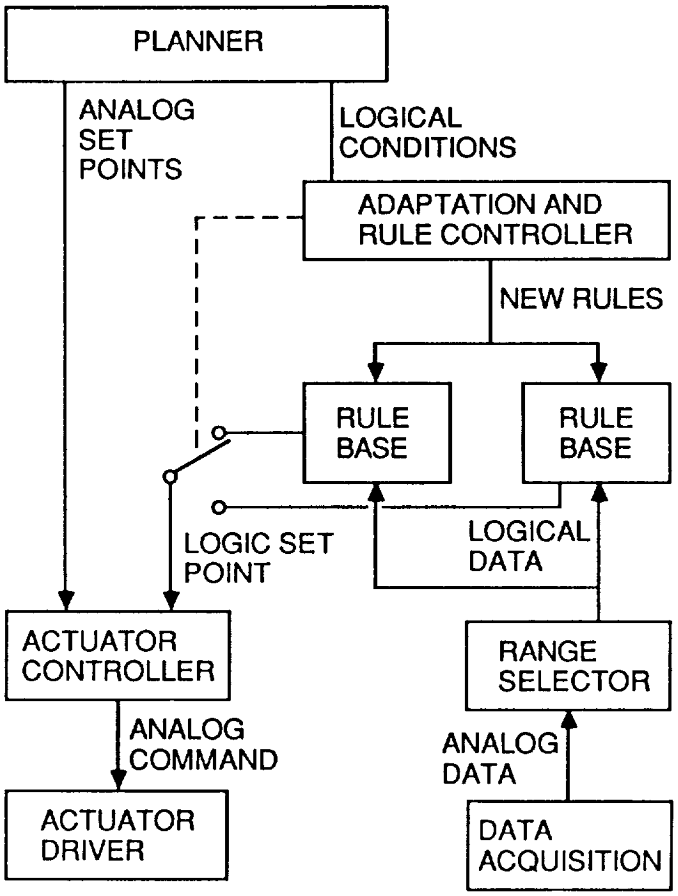


Fig. 1: Hierarchical architecture for robotic control

3. Design Approach to a Control Architecture

In defining a structure for a hand system, two main steps have to be taken: first the analysis of function allocation in the hierarchy, and second their match with candidate implementations to satisfy the needs of real time control. After the initial achievements in designing good mechanisms and control algorithms [3] [4], much effort is now directed towards the definition of an architecture that can include and organize all components of a multifingered hand system. In this situation, it comes quite natural to turn to the analysis of human behavior as a possible model for a control architecture.

In the functional analysis of [8],[11],[12] a causal hierarchy is defined, describing the different types of control actions of human operators supervising industrial plants. The layers of the hierarchy are based on increased level of symbolic representation of the information, from raw sensor readings to descriptions of plant states. At each level, the degree of complexity of the system representation is approximately kept constant. The plant operator is the processing power acting at each level of the hierarchy on different data abstractions. Depending on the case complexity, he/she can initiate a control action at every layer and, in particular, at the one whose data representation best describes the current situation. Such a system then has a capability for a particular reflex action, in the sense that commands to the plant can be generated at all levels of the hierarchy, without the need of reaching the top level of abstraction for deciding the next action. Learning of new skills is also necessary to fine tune the response of the system. A hierarchical structure has the potential for implementing a distributed supervised scheme, in which every layer can receive updates from the next higher level, and can modify the logic of the next lower level in the dual role of supervisor and learner.

The second step in designing a control architecture is the match of the functions assigned to a layer of the hierarchy with a specific implementation. In particular, the interaction between the actuator control and the strategy generated by the grasp planner is not well defined. During task execution , no provision is made for using sensor feedback to update in real time the initial plan, or for switching to alternate ones. The normal approach is to halt the operation when an error condition is recognized, and generate a new plan based on current sensor data. To answer the need for a fast activation of an alternate plan, qualitative conditions are mixed to the robot programs, defining the expected logical conditions of the task. To be more general, these qualitative or logical conditions should be generated by the task planner together with the quantitative information pertinent to the trajectory definition. In the same way that analog information is stored in the actuator controllers, logical conditions should be stored in qualitative controllers that would supervise the task evolution and would manage strategy changes. In this paper we present an example of this by assigning to one of the layers of the control hierarchy a particular class of artificial neural networks called **Dynamically Programmable Logical Arrays** [2], [16], and by implementing it in VLSI technology.

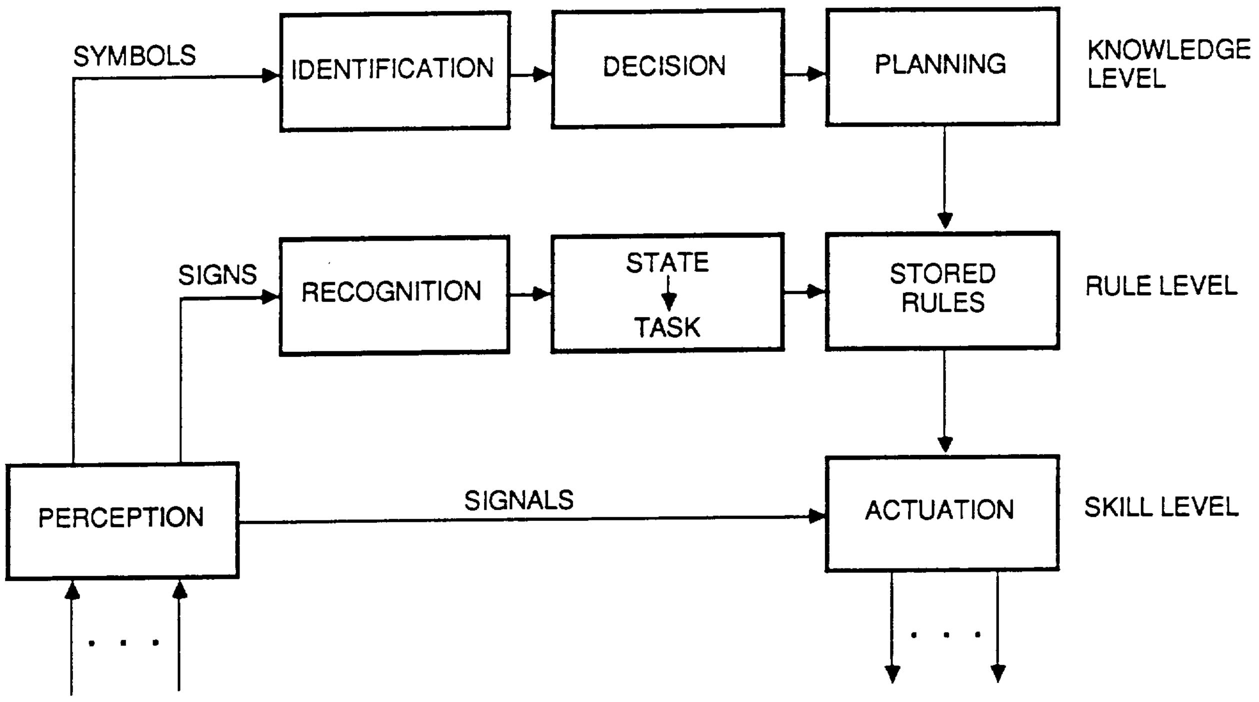


Fig. 2: The Rasmussen-Lind model

4. Functional Hierarchy

The model developed by Rasmussen and Lind and presented in [11] describes the operator's decision making process during the control of complex plants, such as nuclear or power facilities. The proposed data organization closely follows the needs of the human mind in terms of quantity and quality of information presented. Three parameters have been determined to play a major role in the modeling of a control action: causal relations, complexity of representation, and expectation on feedback data. Operators organize their mental model based upon the causal relations existing between elements of the plant, and generate several plant descriptions, each one with a different level of abstraction. In this way, the number of elements in each level, i.e. the level's complexity, can be kept approximately constant. Particular situations direct the operator's attention, and the corresponding control actions are determined by the presence or by the absence of specific feedback signals. Each hierarchical level corresponds to a different kind of mental process. At the bottom layers actions are immediately activated by important feedback signals, middle layers reasoning can recognize typical patterns in the feedback signals and command more complex reactions, and top layers mental process is dedicated to a symbolic type of reasoning in which sequences of patterns can be analyzed.

A schematic representation of this model is in Fig. 2, where three levels of this hierarchical structure are visualized. The bottom level is defined as the skill level,where no conscious action takes place, and feedback/reaction are governed by fixed and automatic relation, e.g. maintaining a level within ranges, or reacting to a particular alarm signal. At this level there is no abstraction on the feedback data, but each signal is considered alone for its particular meaning. In the middle level, some processing of the feedback information is necessary before a control action can take place. This data processing can be quite elementary, and can be visualized as a set of rules combining subsets of feedback signals. At this level, actions are decided on the basis of abstract entities derived from the rules stored at this level. The next higher level represents the knowledge level, where complex inferences have to be made. The type of processing at this level is not characterized by sets of rules, but by general planning functions. If we consider the above description as a possible model for a hierarchical control structure for a robotic system, we see that the decision process performed by the plant's operator can be used as a guide to identify both the degree of data aggregation needed at a certain level of the architecture and the type of processing most suitable for that level. This model can also be used to describe a typical sequence of actions in human manipulation.

The model can be mapped directly into a possible architecture of a control system for an anthropomorphic hand. In a simple three-level structure, the skill level corresponds to the actuator controllers, the rule level corresponds to an intermediate processing of feedback data and actuator commands, and the knowledge level corresponds to the grasp planner, such as the one presented in Figure 1.

5. Adaptive Boolean Networks

This type of network is built with node modules capable of manipulating small sets of input variables with logical operators that can be dynamically programmed to change the boolean function implemented in the node. The overall network is then a combinatorial circuit and its outputs are boolean functions of the inputs. These logical operations can be considered equivalent to propositional logic calculation, compiled into the network as logical rules relating symbolic inputs to symbolic outputs. Due to the nature of its boolean constituents, this processing is completely combinatorial, non-numeric and asynchronous. The architecture is regular with limited connectivity and modules can be easily structured by aggregating groups of functions. Adaptation is an additional feature of these networks that allows them to take an active role in configuring the connections of the logic gates, with the purpose of optimizing some performance index, such as minimality and consistency of the rule base. Dynamic programmability distinguishes these networks from conventional **Programmable Logic Arrays** which realize fixed functions after the initial programming step. They are also different from **User Programmable Gate Arrays** [5], in which the logic function embedded in the circuit can be altered by storing in the array a different set of connections for the logic gates. These arrays play no active role during reconfiguration, but they are reprogrammed on line by an external source.

The underlying concept for this class of combinatorial dataflow architectures is the same than that of **Artificial Neural Networks** (ANN); it is useful to make a brief comparison of the two structures. The prototype structure of an ANN element is a weighted sum of inputs modified by a nonlinear threshold function, while these type of networks have fully input, output and processing boolean. The main consequence of this is the simpler design of both the computing device and its control circuits. In an ANN, learning is achieved by a process of convergence following a gradient path in a multidimensional state space. In adaptive boolean networks the learning process is accomplished by adapting the current rule base to new rules presented to the network by an outside source. This can be viewed as a type of supervised learning in which the network takes the active role of blending the new rule with the old ones to assure consistency and minimality (Fig.3).

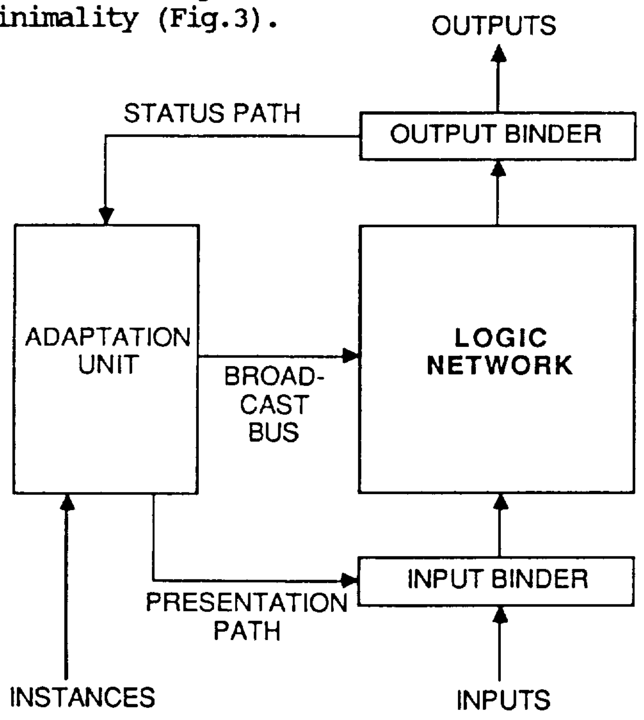


Fig. 3: Adaptive Boolean Network

Problem specification is done by incrementally entering into the system if-then rules expressed as boolean conjunctions. During processing, the network acts as a Programmable Logic Array. The network inputs are discrete variables supplied by the environment. The output of the processing is the truth values for the logic predicates which have been previously stored in the network as logic rules. During adaptation, the network structure changes to update the overall network functions. As new rules are added, the network automatically reconfigures to a logic circuit that seeks to maintain a minimal and consistent rule base. There is no explicit programming of the network, and the internal configuration of the network is not unique and depends on the initial state and on the history of the previous adaptations. The system accepts new rules that are sequentially presented to it by an external controller. This process allows each node in the network to determine its relation with the new rule and determine whether it should be involved in the adaptation process. The adaptation may involve addition or deletion of nodes, or compaction of subnetworks. A central controller is used for coordination, but the adaptive process itself is completely distributed in the network, and modifications to network are performed with considerable concurrency.

Rules consist of a conjunction of boolean variables as antecedent and of a single boolean variable as consequent, e.g.:

$$ABC \rightarrow Z$$

which means that if the antecedent is true then the consequent must be set to true. These rules differ from ordinary boolean functions, because of the characteristic of propositional logic of not specifying anything besides what the rule states. This means that no condition is set for the truth value of (not Z) and that Z can also be true in the absence of the given antecedent. Thus whenever the antecedent of a rule is not matched by the input environment, the instance provides no information about what the output of the system should be. Rule consistency is achieved by resolving all conflicts between two instances that contradict each other, i.e. if they are both eligible to fire for some state of the environment, and they have discordant output. Minimization occurs between concordant instances and only guarantees relative minimality. Minimal representation is achieved by heuristics methods, and not through procedures to derive a theoretical optimum. Minimality is said to have been achieved when no two concordant rules can be equivalently represented by one rule only, or by two rules with fewer variables.

This type of digital neural network can be implemented in several architectures [16], depending on the degree of connectivity among the nodes and the type of communication allowed between nodes and the external controller. A totally connected architecture has been extensively studied [2], and it consists of two function arrays, separated by a controller column, as in Figure 4. The first array is an AND plane, in which all nodes implement a logical AND function. The second array consists of nodes implementing the logical OR functions necessary to build the complete rule out of the minterms built in the AND plane. The central column consists of an array of nodes called D-nodes, which collect the output of a row of the AND plane, thus generating an output corresponding to a minterm of a rule. Variables are associated to nodes in the AND plane by setting a status bit in the node located at the intersection of the column, representing the variable and the row corresponding to the minterm. In a similar way, the OR plane collects the minterms forming a rule by using a chained OR to represent a logical function as a sum of products.

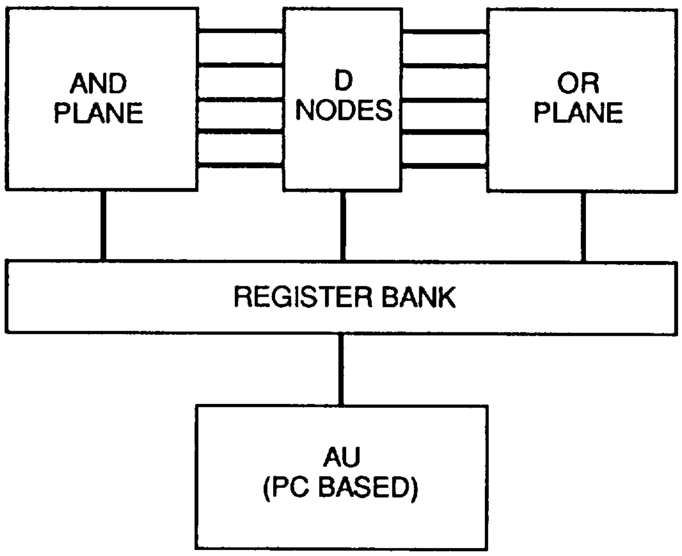


Fig. 4: System Diagram of the Prototype

The adaptation process in this architecture is a two step process. First the new rule is presented to the network by the Adaptation Unit, where each node performs a self classification to determine how it should be operated on, to bring about the correct adaptation of the network. Second, the Adaptation Unit guides the adaptation of the node pool so that a prioritized sequence of operations can perform the network adaptation. Rules are presented to the circuit as a sequence of component minterms, and they are assigned to a specific output, by enabling one of the columns of the OR plane. Complex operations, such as fusion of two rules, are done by the Adaptation Unit to which the network schedules the offending minterm for external minimization.

To experiment with integration of the rule base with an real actuator controller, the network has been implemented in VLSI technology so that it can be located with the controller and will not affect the communication bandwidth of the system. During processing, the network receives the output of range selectors that transform the analog output of the sensors into boolean variables, and then it processes them according to the memorized rules. During adaptation, the network structure changes to update the overall network function. In the present model, we use a recency law for resolving rule conflicts during the transition from a grasping strategy to the following one. The implemented prototype is a four input, four output, eight minterm network, calling for a 4 by 8 AND plane, an 8 node control column and an 8 by 8 node OR plane. The whole chip measures 7900 by 9200 microns and was designed in a 3 micron p-well CMOS technology. The chip is mounted in a 64 pin package, and is currently interfaced to a personal computer for functional testing. All input output lines are buffered to avoid racing conditions among the feedback signals. Figure 5 is a microphotograph of the chip.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

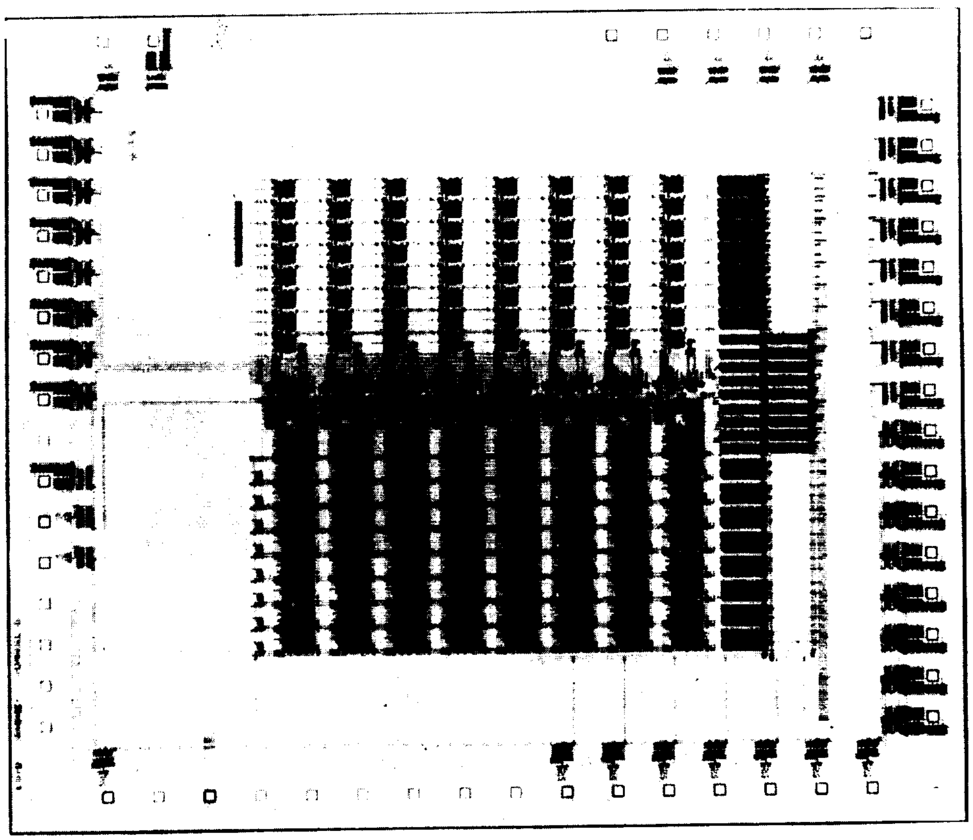


Fig. 5: Microphotograph of the Boolean Neural Network

6. Conclusion

In this paper we have presented the first steps of the design and implementation of a hierarchical architecture for the control of robotic devices such as mechanical hands. The justification for this approach is found in the analysis of human behavior during control functions, and it is an improvement over similar design proposed for intelligent controllers. The three fundamental criteria that this structure satisfies are: causal connections between layers, constant complexity of each layer, and directed focus of attention. The implementation of each layer must obey the same design criteria, and therefore it must change from one layer to the other, to fulfill the requirements of processing type and execution speed of that level. An implementation of a layer has been described, which will act as a rule base, or logical controller, in a three layer architecture, and the characteristics of this implementation as a boolean artificial neural network have been presented.

7. Acknowledgements

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] Bekey, G., Tomovic, R.,"Robot control by reflex action", Proc. Int. Conf. on Robotics and Automation 1986, pp. 240-247.

[2] Chang, J., Vidal, J. J.,"Inferencing in Hardware", MCC University Research Symposium, Austin, Tx, July 1987.

[3] Cutkosky, M.,"Robotic Grasping and fine manipulation", Kluwer Academic Publisher, 1985.

[4] Fiorini, P., "A versatile hand for manipulators", IEEE Control Magazine, October 1988.

[5] Freeman, R., "User-programmable Gate Arrays", IEEE Spectrum, December 1988, pp. 32-35.

[6] Henderson, T., Hansen, C., Bhanu, B.," The specification of distributed sensing and control", Journal of Robotic Systems, 2(4), 387-396(1985).

[7] Lee,S., Kim, H.M., "Learning Expert Systems for robot fine motion control", IEEE Intelligent Control Conference, Philadelphia, PA, 1988.

[8] Lind, M., "The use of flow models for automated plant diagnosis", 1981 NATO Symposium on Human Detection and diagnosis of system failures.

[9] Liu, H., Iberall, T., Bekey, G., "Building a generic architecture for robot hand control", Proc. IEEE Int. Conf. on Neural Network, July 24-27 1988, San Diego, CA, pp. II 567-574.

[10] Meystel, A., "Intelligent control in robotics", Journal of Robotic Systems, 5(4),269-308 (1988).

[11] Rasmussen, J., Lind, M.,"Model of human decision making in complex systems and its use for design of system- control strategies", American Control Conference, Arlington, VA, 14-16 June 1982.

[12] Rasmusen, J., Lind, M., "Coping with Complexity", RisO-M-2293, 1982, European Annual Conference on Human Decision and Manual Control, Deft 1981.

[13] Saridis, N. G., "Knowledge implementation: structures of intelligent control", Journal of Robotic Systems, 5(4), 255-268 (1988).

[14] Stephanou, H.E., Erkmen A.M.," Evidential classification of dexterous grasps for the integration of perception and action", Journal of Robotics Systems, 5(4), 309-336(1988)

[15] Tomovic, R., Bekey, G., Karplus, W., "A strategy for grasp synthesis with multifingered robot hands", Proc. Int. Conf. on Robotics and Automation 1987, pp. 83-89.

[16] Vidal, J. J., "Implementing Neural Nets with programmable logic", IEEE Trans. Acoustics, Speech and Signal Processing, Vol. 36, N. 7, July 1988.

N90-29340

Performance Limitations of Bilateral Force Reflection Imposed by Operator Dynamic Characteristics

Jim D. Chapel

Martin Marietta Astronautics Group
Denver, Colorado

Abstract

This paper presents a linearized, single-axis model for bilateral force reflection which facilitates investigation into the effects of manipulator, operator, and task dynamics, as well as time delay and gain scaling. Structural similarities are noted between this model and impedance control. Stability results based upon this model impose requirements upon operator dynamic characteristics as functions of system time delay and environmental stiffness. An experimental characterization reveals the limited capabilities of the human operator to meet these requirements. A procedure is presented for determining the force reflection gain scaling required to provide stability and acceptable operator workload. This procedure is applied to a system with dynamics typical of a space manipulator, and the required gain scaling is presented as a function of environmental stiffness.

1. Introduction

The development of NASA's Flight Telerobotic Servicer (FTS) provides the robotics research community many engineering challenges which must be addressed to provide a safe and effective space-based teleoperation system. One of the problems confronting this new initiative is the adaptation of control techniques widely used in nuclear and undersea teleoperation systems to a space-based system. The control techniques utilized in these existing systems have evolved over the past three decades to provide the operator with an interface that is comfortable and nearly transparent. However, the teleoperation technology base developed during this time period is not broad enough to facilitate engineering design and performance prediction for a system with significantly different characteristics. Digital sampling and communication delay inherent in a space teleoperation system have been shown to degrade the performance and stability of force-reflecting teleoperation systems [1,2]. Additionally, the dynamics of manipulators designed for use in space, as shown in Figure 1, are likely to be significantly different from those of manipulators designed for terrestrial applications. To facilitate a rational design process, a model-based analysis is needed that can predict performance of teleoperation control schemes when implemented in any specific system.

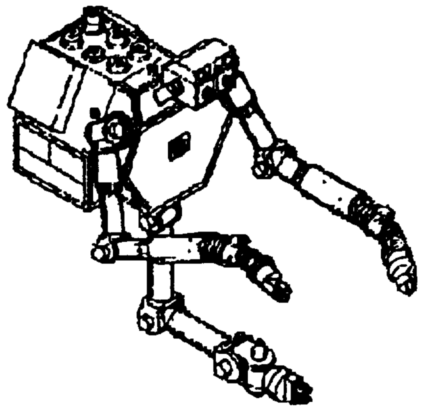


Figure 1 Early Flight Telerobotic Servicer Concept

This paper examines the dynamic characteristics of force-reflecting teleoperation systems, and investigates the effects of communication time delay, digital sampling, task dynamics, and operator dynamics on stability and performance. The technical approach presented here examines the dynamic characteristics of bilateral force reflection by exploiting similarities between bilateral force reflection and position-based impedance control. In both cases, a dynamic relationship is established between the measured environmental force and commanded manipulator position. In the case of impedance control, this relationship is realized by specifying parameters in a digital "impedance" filter, whereas in the case of bilateral force reflection, this relationship is determined by the dynamic characteristics of the human operator/hand controller combination. Unlike the impedance filter parameters, the human operator dynamic characteristics are nonlinear and time varying. Analysis of the linear time invariant problem generates stability boundaries for impedance filter design as functions of communication time delay, digital sampling and task dynamics [1,3,4]. Comparison of the impedance filter parameters to the physical parameters of the human operator/hand controller model reveals that this stability analysis imposes requirements on human operator stiffness and damping.

An experimental frequency response characterization of five test subjects is presented that quantifies the limited capabilities of human operators to provide the dynamic parameters needed to stabilize the system. The dominant features in the frequency range of interest are found to be captured using a second order model parameterized by stiffness, damping, and inertia. Test results are presented for minimum, moderate, and maximum exertion levels. Operator dynamic parameters are demonstrated to be closely coupled over the range of capability, i.e., operator stiffness cannot be generated independent of operator damping. The results of this characterization provide a basis for determining operational capabilities and limitations of force-reflecting teleoperation systems controlled by human operators.

The impedance control stability analysis imposes requirements on dynamic compensation in the force reflection loop, and the operator dynamic characterization provides the limits of human capabilities to provide this compensation. Comparison of human operator capabilities to the dynamic feedback requirements can be used to determine the stability and operator workload of bilateral force reflection. If the system is not stable, or the resulting workload is considered excessive, the analysis presented also shows how gains within the system, namely the force reflection and position gains, can be adjusted to provide acceptable performance.

2. Teleoperator Force Reflection Analysis

The free-space position response of the teleoperation system's manipulator is assumed to be accurately modeled by a second order transfer function. This assumption is valid for position controlled manipulators that do not exhibit flexible modes in the frequency range of interest, which is generally less than 10 Hz. Although the coefficients of this transfer function vary depending upon the manipulator's pose, the following analysis considers operation about a point in space so that the transfer function coefficients can be assumed constant. Stiffness is often the dominant effect of the environment, especially for assembly and maintenance tasks using metal parts. The environment can then be modeled by a single stiffness term, K_e. The transfer function relating position response to position command for the manipulator in contact with the environment is then given by

$$\frac{X}{X_c} = \frac{K_m}{J_m s^2 + B_m s + K_m + K_e} \qquad (1)$$

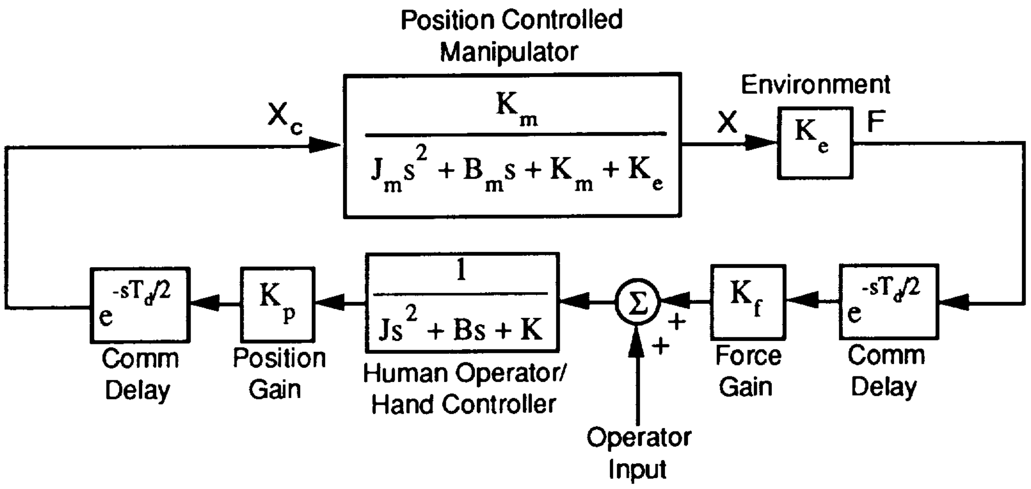
where K_m, B_m, J_m are the controlled manipulator stiffness, damping, and inertia, respectively. The forces seen at the manipulator are assumed reflected back to the hand controller through a communication link with time delay $T_d/2$. A feedback gain, K_f, is provided to scale the sensed environmental forces to a comfortable level for the operator. The human operator/hand controller system can be modeled as a second order dynamic system as well, but the transfer function coefficients are not constant because of varying levels of operator exertion. For this analysis, the operator dynamic parameters are assumed constant during the environmental interaction being studied. The resulting transfer function relating reflected force to hand controller position is given by

$$\frac{X_{hc}}{F} = \frac{1}{Js^2 + Bs + K} \tag{2}$$

where K, B, and J are the stiffness, damping, and inertia of the human operator/hand controller combination, respectively. A feedforward gain, K_p, is also provided to scale the hand controller motion to commanded manipulator motion. Finally, the commanded manipulator position is issued to the manipulator through a communication link with time delay $T_d/2$. The open loop transfer function of the system is given by

$$T(s) = \frac{K_p K_f K_e K_m e^{-sT_d}}{\left(J_m s^2 + B_m s + K_m + K_e\right)\left(Js^2 + Bs + K\right)} \tag{3}$$

Figure 2 presents the system structure in block diagram form. A reference input is introduced as representative of the operator inputs to the hand controller. Examining the block diagram of this system, we can see that bilateral force reflection forms a feedback control scheme with a dynamic compensator. The dynamics of the manipulator, the environment, and the operator/hand controller combination, as well as communication/computation time delay and control law gains, are all important in determining the stability of bilateral force reflection. Studying the magnitude and phase characteristics of the open loop transfer function given in Eq. (3) provides insight into the stability and performance of force-reflecting teleoperation systems that can be modeled as shown in Figure 2.



Position Controlled Manipulator
$\frac{K_m}{J_m s^2 + B_m s + K_m + K_e}$
X_c
X
Environment
K_e
F
$e^{-sT_d/2}$
Comm Delay
K_p
Position Gain
$\frac{1}{Js^2 + Bs + K}$
Human Operator/ Hand Controller
Σ
+
+
Operator Input
K_f
Force Gain
$e^{-sT_d/2}$
Comm Delay

Figure 2 Force-Reflecting Teleoperation Block Diagram

When the structure of this form of force reflection is compared to those of other forms of active force control, it is seen that the structure presented here is identical to the "impedance control" structure described in the robotic systems literature [5,6]. In both cases, the position command issued to the manipulator is modified by sensed environmental forces passed through a dynamic filter, thereby establishing a dynamic relationship between manipulator force and position. This relationship can be interpreted as a mechanical "impedance." Unlike the impedance control case where impedance filter parameters can be programmed arbitrarily, bilateral force reflection has a limited capability of providing filter parameters because of the human operator's physical limitations. Even so, the requirements on the physical dynamics of the human operator/hand controller are identical to those imposed on the impedance filter dynamics to ensure stability. This allows stability analysis results derived for impedance control to be applied to this problem. Details of the analysis have been previously published [3,4], and will not be repeated here. A summary of the approach and the results of this analysis are presented in the following discussion.

To find the stability boundaries of an impedance control system with a second order impedance filter described by Eq. (2), the parameters must be found for the open-loop transfer function in Eq. (3) that result in a marginally stable system. Because we are interested in impedance filter design requirements, we wish to find the parameters J, B, and K that produce a marginally stable system while the other parameters in the system remain fixed. If the characteristic equations of both the manipulator and impedance filter transfer functions in Eqs. (1) and (2) have damping ratios of at least 0.707, the magnitude response is a monotone decreasing function of frequency. This requirement implies that no resonant peaks be present in the magnitude response for either of these parts of the system. If the impedance filter represents the human operator/hand controller dynamics, this would normally be the case to provide acceptable performance and feel. The manipulator control system would typically be designed to exhibit this characteristic when not in contact with the environment, but the damping ratio of the characteristic equation of Eq. (2) decreases as the environmental stiffness, K_e, increases. For some range of environmental stiffnesses, the damping ratio of the manipulator's position response would still be greater than 0.707. The case where this assumption is not valid is discussed later in this paper. If the damping assumption holds, the simultaneous solution of the magnitude equation for a magnitude of unity and the phase equation for a phase angle of -180 degrees provides a unique solution for the impedance filter parameter K given J and B. Introduction of time delay into the system decreases the phase linearly with frequency and therefore does not affect the uniqueness of solution.

Numerically solving this nonlinear system of equations with various communication/computation time delays, we find stability boundaries on the K-B plane of the form shown in Figure 3. For the solution shown, the impedance filter parameter corresponding to the hand controller inertia, J, was set to zero. This assumption is equivalent to the inertia of the operator/hand controller combination being small compared with its stiffness and damping characteristics. Manipulator parameters used for the stability boundaries presented in Figure 3 are representative of dextrous space manipulators, corresponding to a manipulator weighing 150 lbs being controlled by a moderate performance position controller with a bandwidth of 1.5 Hz. Experimental studies using an impedance-controlled industrial manipulator have verified the general characteristics of these stability boundaries, and have also shown close agreement between analytical predictions and experimental measurements [3].

Interpreting the results shown in Figure 3 for the case of force-reflecting teleoperation, we see that the stiffness and damping that must be provided by the operator are quantified for the system with the given parameters. It is important to note that even with no time delay in the system, some operator stiffness and damping is required to stabilize the system. This phenomenon is caused by the phase lag or transport delay of the position-controlled manipulator. Not surprisingly, increasing the time delay within the system while keeping the control gains the same increases the requirements for operator stiffness and/or damping to retain stability. Because the operator provides stiffness and damping to the system by tightening his arm muscles, increased time delay sharply increases the physical workload of the operator. Additionally, the increased requirements may exceed the operator's capability to stabilize the system. Because the stiffness and damping parameters required are the ratios of operator stiffness and damping to the product of the control gains, decreasing the position gain, K_p, the force reflection gain, K_f, or both reduces the stiffness and damping requirements of the operator. In this way, a system with arbitrarily large time delay can be stabilized. Although it is certainly possible to decrease the force reflection gain to have a stable system with several seconds of time delay, the gain would be so small that force reflection would be effectively disabled. Smaller reductions in the control gains also degrade the performance of force reflection by reducing the "crispness" or "feel" of the system.

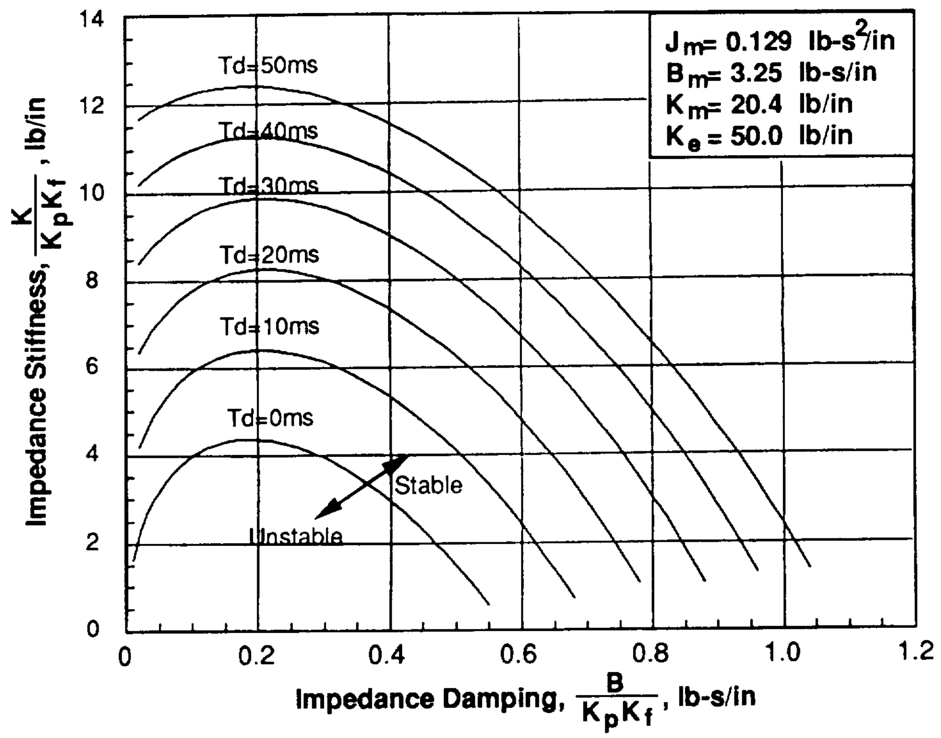


Figure 3 Stability Boundaries as a Function of Time Delay

Families of solutions can also be generated on the K-B plane for different values of the environmental stiffness. For systems with no time delays, the Routh-Hurwitz criterion can be used to determine stability conditions on K, B, and J [3]. If K and B are restricted to being positive real numbers and J is again set to zero, the constraint equation for K and B to retain stability is given by

$$K > -B\frac{B_m}{2J_m} + \sqrt{B^2\left(\frac{B_m^2}{4J_m^2} - \frac{K_e + K_m}{J_m}\right) + B\frac{K_pK_fK_eK_m}{B_m}} \qquad (4)$$

If the controlled manipulator dynamic characteristics, J_m, B_m and K_m, are held constant and K_e is allowed to vary, the relationship in Eq. (4) produces a family of stability boundaries parameterized by B, K, and K_e. Using the same manipulator dynamics that were used to generate the plot in Figure 3, we can find these stability boundaries in the K-B plane as shown in Figure 4. For a given set of control gains, larger environmental stiffnesses require higher values of stiffness and damping from the operator. Higher environmental stiffnesses therefore require higher operator workload and will generally produce a less stable system. As discussed before, the control gains K_p and K_f can be adjusted to guarantee that the system is stable, but at the cost of making the environment feel more spongy to the operator.

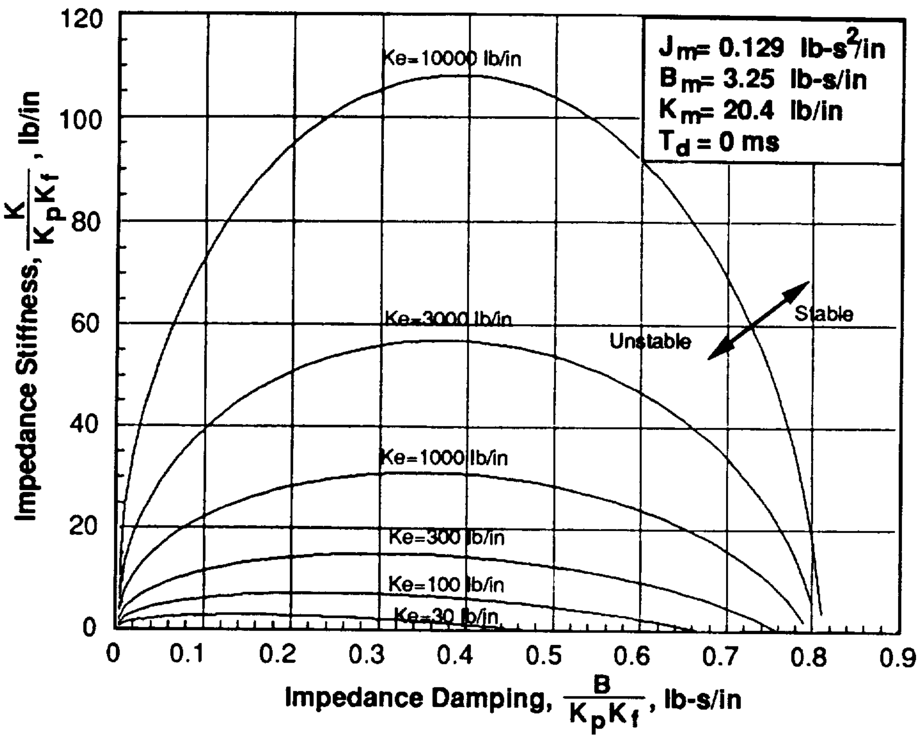


Figure 4 Stability Boundaries as a Function of Environmental Stiffness

If time delay is introduced in to the system, an analytical solution can no longer be obtained. As long as the environmental stiffness, Ke, is not large enough to produce a resonant peak in the magnitude response of Eq. (1), the same method of solution can be used to find stability boundaries for various environmental stiffnesses as was used to find the stability boundaries for various time delays. When a resonant peak is present in the magnitude response of the manipulator transfer function, the simultaneous solution of the magnitude and phase equations for a magnitude of unity and phase of -180 degrees no longer guarantees a marginally stable system. To find the values of K, B and J that produce a marginally stable system in this case, the Nyquist plot needs to be found that crosses the real axis at the -1 point and does not encircle the -1 point. Because the resonant mode produces a nearly circular contour in the Nyquist plane [7] and the phase changes rapidly near the resonant frequency, the real-axis intersection is approximately given by the real part of the open-loop transfer function, T(s), evaluated at the resonant frequency. Use of this approximation results in the following stability constraint:

$$Re\,(\,T(j\omega_r)\,) > -1 \qquad (5)$$

where ω_r is the resonant frequency of Eq. 1. This approximation is most accurate when the damping of the manipulator dynamics is small, or alternatively, when the environmental stiffness is large. The dynamics of the manipulator in contact with the environment are already known, so this constraint imposes limits on the impedance filter to maintain stability. The constraint given in Eq. (5) can then be used to determine the constraints upon impedance filter design to stabilize systems with time delay.

3. Human Operator Dynamic Characterization

The stability analysis presented above provides requirements for any dynamic force compensation implemented in the structure of Figure 1. Force-reflecting teleoperation is of this form and therefore requires specific human operator dynamic characteristics to retain stability. An experiment is presented here that determines operator dynamic characteristics over a range of test subjects from minimum to maximum exertion levels. Comparison of the experimentally demonstrated capabilities to the requirements derived from the stability analysis provides information about system stability and operator workload.

To investigate the operator dynamic characteristics, a one-DOF hand controller was set up as shown in Figure 5. The test setup consisted of an Inland brush motor attached to the input shaft of a harmonic drive with a 100:1 gear ratio. To obtain primarily translation motion, a hand grip with a 12 inch link was attached to the output shaft of the harmonic drive. The effects of drive friction and reflected inertia were minimized by implementing a moderate bandwidth analog torque loop on the harmonic drive. A conductive plastic potentiometer, mounted on the motor shaft, was used as the angular position sensor. The resulting one-DOF hand controller had the desirable characteristic of high torque levels yet was still responsive to extremely small operator force inputs.

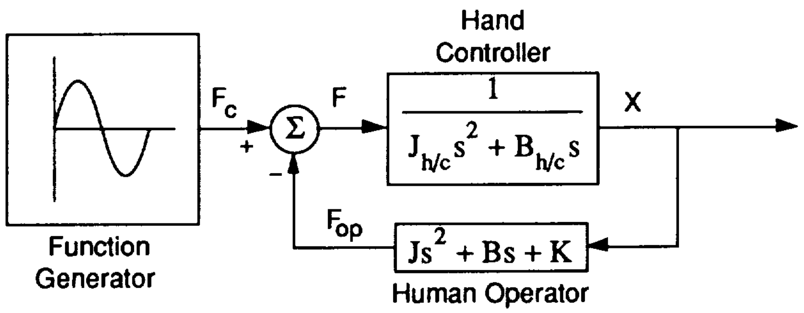


Figure 5 One-DOF Hand Controller Testbed

A set of five operators was tested using this experimental hand controller configuration. Motion in the direction forward from the body was chosen as representative of many teleoperation assembly and insertion tasks. Frequency response data was taken from 1.0 Hz to 6.0 Hz to obtain the transfer function between hand controller force and position. Each operator made three runs consisting of minimum operator exertion, moderate operator exertion, and maximum operator exertion. The torque input was set such that the operators could not stop hand controller motion even when exerting maximum force. Stiffness, damping, and inertia parameters were extracted from the frequency response data by first fitting a second order response to the frequency response data. The dc level was used to extract the operator stiffness, K. The second order curve fit and the parameter K were then used to determine the operator damping and inertia, B and J, respectively. These parameters represent the dynamics of the human operator/hand controller combination in Eq. (2). By subtracting off the hand controller damping and inertia terms, $B_{h/c}$ and $J_{h/c}$, determined from a separate open-loop frequency response on the hand controller alone, the human operator dynamic parameters can be determined.

The stiffness and damping characteristics of the human operators, determined by the method described above, are shown in Figure 6. For each operator, effective inertia displayed only a small variation between runs compared with the variation in the stiffness and damping parameters. The maximum inertia values were observed for the maximum exertion case where the operators attempted to resist hand controller motion by tensing the upper body muscles. Because more body mass is active during hand controller motion in this case, an increase in effective operator inertia of approximately a factor of two was observed for each operator. Measured operator inertias displayed negligible variation between minimum and moderate exertion runs.

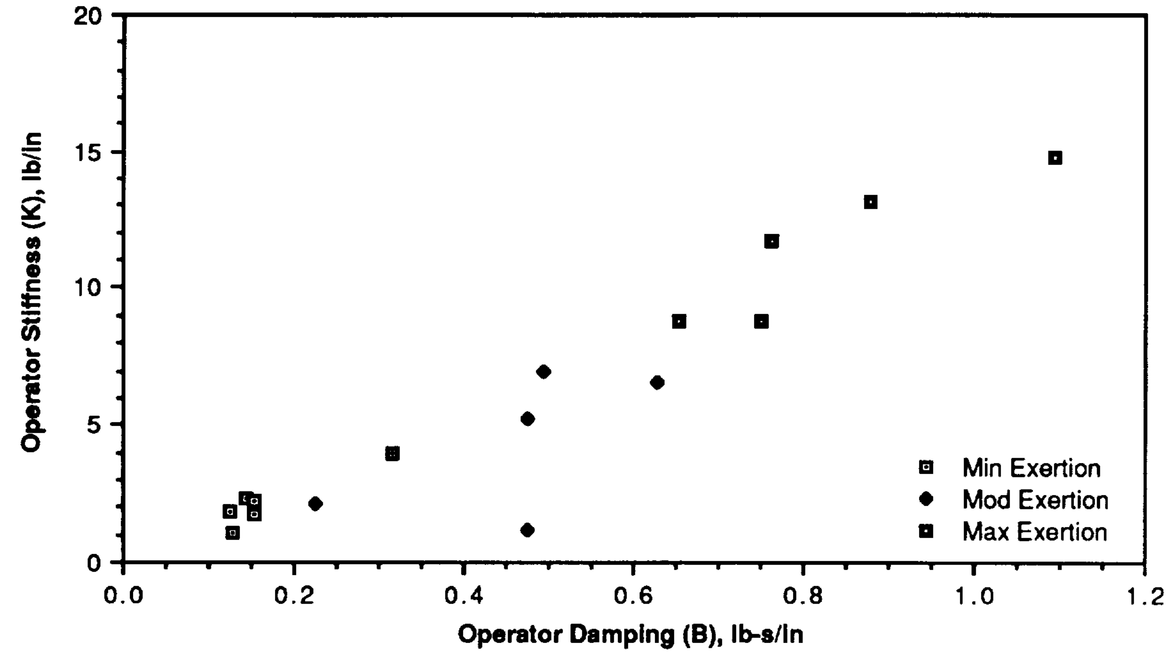


Figure 6 Operator Characterization Results

Two key results are observed from this testing. First, human operators clearly have limited capabilities for providing stiffness and damping to stabilize force-reflecting teleoperation systems. In the direction forward from the body, system designs requiring more than 10 lb/in stiffness or more than 0.7 lb-s/in damping would be unacceptable for prolonged operation. Systems requiring more than 15 lb/in stiffness or more than 1.1 lb-s/in damping could not be stabilized by human operators in this direction. Second, stiffness and damping apparently cannot be provided independently by human operators. In fact, the two parameters appear to be linearly related. This simplifies the required stability analysis because only a small range of the stability boundaries need to be examined to determine stability of the overall system and operator workload.

4. Teleoperation Performance and Stability Implications

The results of the stability analysis can be combined with the human operator characterization data to predict whether or not a force-reflecting teleoperation system can be stabilized by a human operator. If the operator is able to provide the required stiffness and damping characteristics, the operator workload can be predicted by examining the location of the required dynamic characteristics within the range of operator capabilities. Finally, if the system cannot be stabilized by the operator, or the workload is considered unacceptable, the required force reflection gain, K_f, and position gain, K_p, can be computed to ensure system stability and appropriate workload level. Examining the same system presented in the stability analysis above and considering the case where there is no time delay in the system, we can determine system stability, operator workload, and required gain scaling for the ideal situation. As shown in Figure 3, the addition of time delay into the system will increase operator dynamic requirements or, alternatively, will increase gain scaling requirements.

To compare the requirements of the force reflection compensator with the capabilities of the human operator for this ideal case, a linear curvefit is first made to the data shown in Figure 6. This linear equation is then scaled by K_p and K_f to account for non-unity control gains, and the resulting equation modeling the human operator can be written as

$$\frac{K}{K_p K_f} = 14.2 \frac{B}{K_p K_f} \qquad (6)$$

Overplotting Eq. (6) on Figure 4 provides a comparison between the human operator dynamic characteristics and the stability behavior of the force-reflecting system given in Figure 2 as the environmental stiffness is allowed to vary. By searching along the line given by Eq. (6), we can find environmental stiffness values that produce a marginally stable system for particular values of K/K_pK_f and B/K_pK_f. However, the human operator has clear limitations on providing stiffness, K, and damping, B. If the operational limitations of the operator are used for K and B in Eq. (6), the gain scaling required to provide a marginally stable system can be found as a function of the environmental stiffness. Examination of the data from Figure 6 shows that an operator stiffness of less than 8 lb/in was easily provided by the operators. Stiffness values above 10 lb/in required an excessive amount of effort and would not normally be considered as within the operators' capability range. Stiffness values between 8 lb/in and 10 lb/in required substantial effort but could be provided comfortably for short periods of time. Operator stiffness values of 8 lb/in and 10 lb/in, along with the corresponding operator damping values from Eq. (6), were used to find the required control gain, K_pK_f, to provide a marginally stable system as a function of the environmental stiffness, K_e. The resulting plot of K_pK_f as a function of K_e is shown in Figure 7.

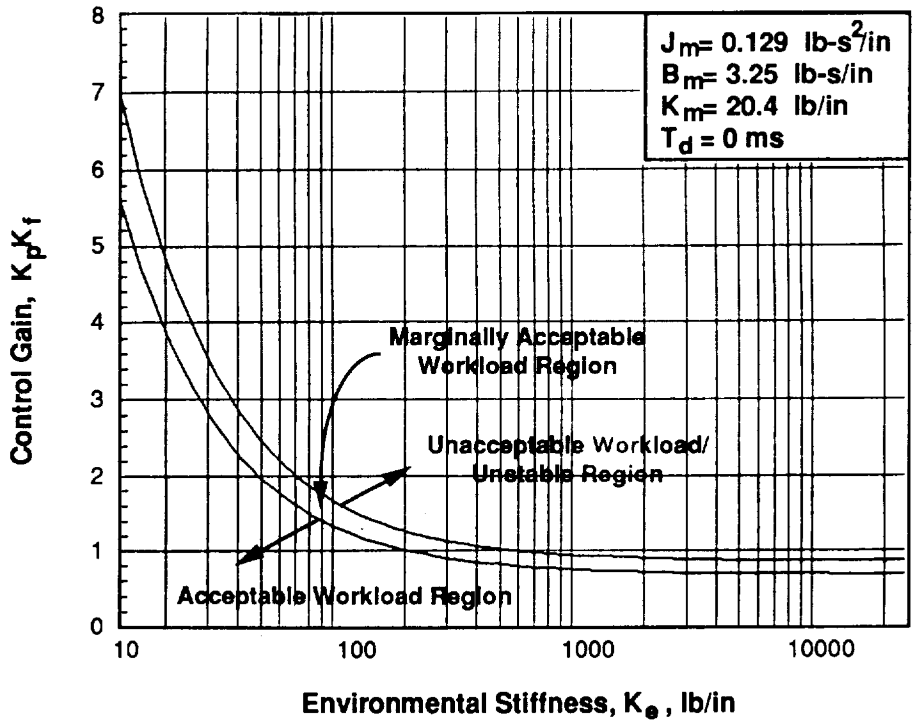


Figure 7 Gain Scaling Required to Stabilize Force Reflection

Examining the curves shown in Figure 7, we see that the product of the control gains, K_pK_f, can be greater than unity for small environmental stiffnesses. The allowable control gains decrease with increasing environmental stiffness. Even with no time delay, this system cannot be stabilized by the operator when the product of the control gains, K_pK_f, is unity and K_e is greater than 500 lb/in. Either the position gain, K_p, or the force reflection gain, K_f, or both must be reduced to operate this system in contact with stiff environments. The K_pK_f curve becomes flat as K_e increases above 1000 lb/in. Because of this, a single value of K_pK_f can be used for a large range of stiff environments.

5. Summary

The existing teleoperation technology base needs to be expanded to allow efficient design, development and deployment of the first generation of dexterous space robots. The initial work presented here has examined the stability and performance of the bilateral force reflection control scheme, commonly used in current teleoperation systems, when incorporated into a system with characteristics typical of space manipulators. Because of the similarities between this control scheme and the active force control scheme known as "impedance control," recent stability results from the analysis of impedance control can be applied to this system. These analyses show that time delays as small as 10 ms, such as those caused by communication delay or computation time, can significantly increase the stiffness and damping required from the operator to stabilize the system. The stiffness of the environment is also important in determining the stability of the system. Interaction with stiff work pieces requires large operator stiffness and damping parameters to retain stability. Experimental data has shown the operator stiffness and damping characteristics to be tightly coupled, and has shown the maximum operational values for operator stiffness and damping to be 10 lb/in and 0.7 lb-s/in, respectively, for the translational direction forward from the body. Scaling of the position and/or force control gains will allow any system to be stabilized, but at the cost of reducing the feel to the operator. A procedure was introduced that uses the operator model to find the required gain scaling as a function of the environmental stiffness. Even with no time delay, the example system representative of a space manipulator would require gain scaling when interacting with environmental stiffnesses larger than 500 lb/in. However, a large range of high stiffness values can be accommodated with little or no change in the gain scaling. Analysis of time delay effects indicates that more gain scaling would be required for a model including time delay. The further development of the understanding of how these critical system parameters affect overall stability and performance will allow standard engineering design techniques to be used to develop an effective control strategy for any specific telerobot system.

6. References

1. Chapel, J. D., and Lawrence, D. A., "Stability Analysis for Alternative Force Control Schemes as Applied to Remote Space Teleoperation," Proceedings of the 10th Annual AAS Guidance and Control Conference, February 1987.

2. Hannaford, B., and Anderson, A., "Experimental and Simulation Studies of Hard Contact in Force Reflecting Teleoperation," Proceedings of the 1988 IEEE International Conference on Robotics and Automation, April 1988.

3. Lawrence, D. A., and Stoughton, R. M., "Position-Based Impedance Control: Achieving Stability in Practice," Proceedings of the 1987 AIAA Guidance, Navigation, and Control Conference, August 1987.

4. Lawrence, D. A., "Impedance Control Stability Properties in Common Implementations," Proceedings of the 1988 IEEE International Conference on Robotics and Automation, April 1988.

5. Whitney, D. E., "Force Feedback Control of Manipulator Fine Motions," ASME Journal of Dynamic Systems, Measurement, and Control," June 1977, pp. 91-97.

6. Hogan, N., "Impedance Control: An Approach to Manipulation," ASME Journal of Dynamic Systems, Measurement, and Control," March 1985, pp. 1-24.

7. Ewins, D. J., Modal Testing: Theory and Practice, Research Studies Press Ltd, Letchworth England, 1984, pp. 158-168.

N90-29341

Sensor-based Fine Telemanipulation for Space Robotics

M. Andrenucci^, M. Bergamasco*°, P. Dario*°

^ Dipartimento di Ingegneria Aerospaziale, University of Pisa
* Scuola Superiore S. Anna, Pisa
° Centro "E. Piaggio", University of Pisa, Italy

Abstract

The control of a multifingered hand slave in order to accurately exert arbitrary forces and impart small movements to a grasped object is, at present, a knotty problem in teleoperation.

Although a number of articulated robotic hands have been proposed in the recent past for dexterous manipulation in autonomous robots, the possible use of such hands as slaves in teleoperated manipulation is hindered by the present lack of sensors in those hands, and (even if those sensors were available) by the inherent difficulty of transmitting to the master operator the complex sensations elicited by such sensors at the slave level.

In this paper an analysis of different problems related to sensor-based telemanipulation is presented. The general sensory systems requirements for dexterous slave manipulators are pointed out and the description of a practical sensory system set-up for the robotic system we have developed is presented.

The problem of feeding-back to the human master operator stimuli that can be interpreted by his central nervous system as originated during real dexterous manipulation is then considered. Finally, some preliminary work aimed at developing an instrumented glove designed purposely for commanding the master operation and incorporating Kevlar tendons and tension sensors, is discussed.

1. Introduction

A number of robotic tasks in space will involve operations inside narrow places such as small cells, tanks, platforms, or on special extravehicular structures. Some of those tasks will require high dexterity and complex sensorymotor control procedures. Environment conditions (e.g. zero g) will strongly affect the execution and the performance of the specific task accomplished by the robot. Nevertheless, all the other conditions of the particular task could vary according to the different procedure used and, in general, the same task will not be repeated in the same conditions. For the above reasons, a typical robot for space applications, even if it may possess almost the same hardware of a common industrial robot, most often requires remote human control (1).

We have elected to investigate in this paper a particular, though fundamental, function in which teleoperated robots are involved, that is telemanipulation. It is worth observing that telemanipulation, which is ultimately aimed at extending the sensing and manipulation capabilities of the human operator to the slave robotic system, requires not only a dexterous slave end-effector, but also a sensory system able to sense and transmit complex tactile and kinestetic sensations.

A number of articulated robotic hands have been proposed in the recent past in the field of autonomous robotics for dexterous manipulation (2) (3). The use of such hands as "slaves" in teleoperated manipulation is hindered primarily by the present lack of sensors in those hands. Furthermore, even if those sensors were available, it would be inherently difficult to convey to the operator the complex sensations elicited by such sensors at the slave level. This is a fundamental problem in telepresence: the telemanipulation system should allow the human operator not only to observe the manipulated objects, but even to feel the physical contact with them.

Current state of the art in telemanipulated end-effectors includes joysticks and handles, or grippers, incorporating some simple sensors. At the master device level, some additional sophistication has been achieved with the DataGlove (4), which incorporates fiberoptic position sensors located at the finger joints, and a 6-degree-of-freedom tracking device mounted at the wrist which provides information about position, orientation and whole configuration of the human operator hand in 3D space.

It is the objective of the research reported here to investigate the design principles and to identify the main problems involved in the development of a master-slave system which could be used for sensor-based telemicromanipulation experiments. In particular our ultimate goal is to render a human operator able to control a multifingered hand slave in a truly dexterous way, that is to accurately exert arbitrary forces and impart small movements to a grasped object belonging to a remote operational space.

In the following some basic considerations on the design of a telemanipulation system are discussed first. These considerations are related to the general system architecture and to the requirements for slave and master devices, as well as to the sensory systems which have to be integrated in their structures in order to achieve an active bilateral control of the manipulative operation. The following paragraphs deal with the description of the simple robotic system we are currently developing in order to investigate some basic issues in telemicromanipulation. The robotic system consists of a tendon actuated robotic slave finger with joint rotation and torque sensors and tactile sensors at the fingertip, and of an anthropomorphic glove-like exoskeleton incorporating actively controlled joints for reproducing kinesthetic sensations on the master human operator.

2. General design considerations for a telemanipulation robotic system

As a first step towards the development of a telemanipulation robotic system, we have attempted to define some general specifications both on the principles of operation (for example, the way in which the whole process could be performed) and on the specific hardware characteristics that a master-slave system should possess for carrying out telemicromanipulation procedures.

As far as the human control of the remote manipulative task is concerned, we assume that the operator (either the astronaut or a ground operator) will usually not supervise the operation just by giving commands to a computer and leaving the execution of semi-automated manipulation procedures to the robotic system. Rather, we assume a direct and continuous human control on the operation. In fact we have even conceived a strict isomorphic relation (**isomorphism**) between the robotic hand and the human master hand.

The isomorphic assumption leads to a clear emphasis in our approach for the concept of telepresence (or tele-existence) of which the telemanipulation task represents only one (although fundamental, because it is "active") aspect. For this reason, we have imagined a scenario conceptually rather similar to that already introduced in the virtual display-control interface for the DataGlove (4), where the human operator, wearing a video display in which the video image of the operational space is represented, feels himself as present in the remote working place. A pictorial representation of the possible scenario is given in Figure 1.

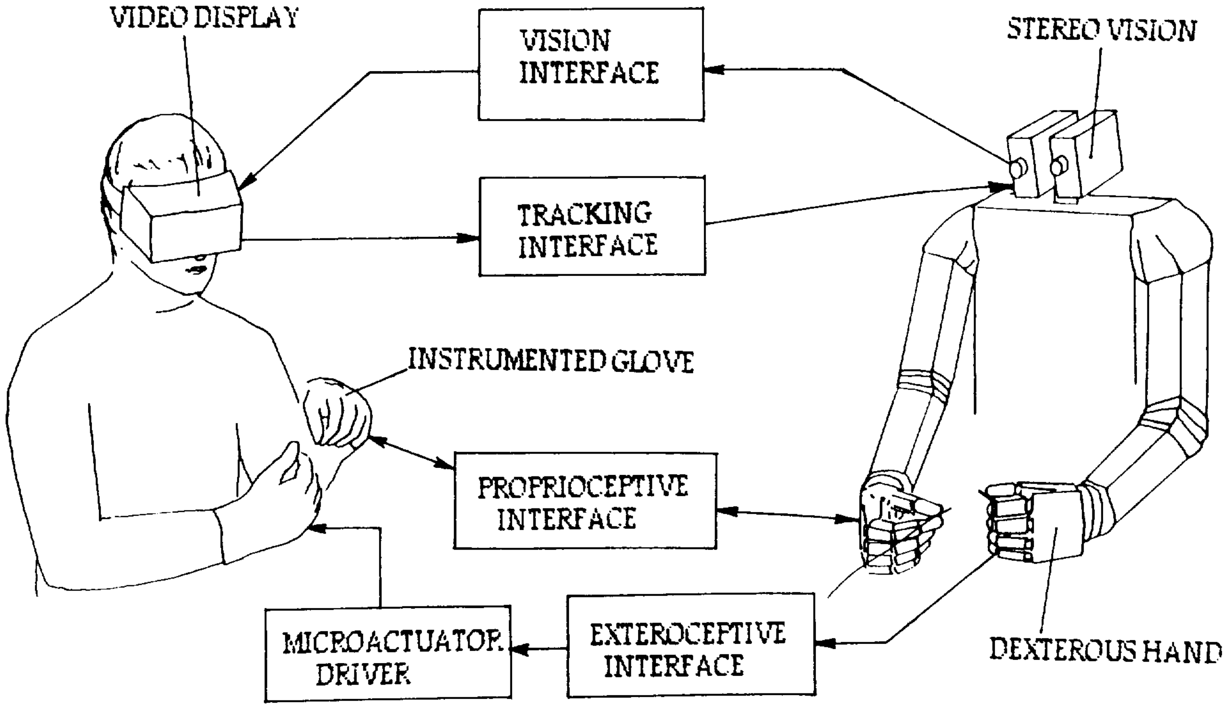


Figure 1. A scenario for telemicromanipulation.

In analogy with the expected performance of the visual feedback in the case of ideal telepresence, we assume that also the systems designed for feeding-back the contact information detected by the slave robotic hand during manipulation procedures will generate **adequate stimuli** (5) in the human operator hand. The term "adequate stimuli" means that the sensations evoked to the human brain cortex when the manipulation procedure is performed directly by the human hand should be similar to those evoked in the "artificial" situation in which the manipulaton procedure is actually performed by the artificial slave hand. This fact implies, that the contact information (i.e. that related to exteroceptors) should be conveied physically to the hand of the master operator, without any display interface, such as a computer-graphic display or other equivalent devices.

Another important consideration for the definition of a robotic system for telemanipulation refers to the availability of a dexterous robotic hand equipped with sensory systems of various kinds. This requirement originates from the very concept of dexterous manipulation, which requires an articulated effector equipped with proprio- and exteroceptive sensors, commanded through a hierarchy of sensory-motor control procedures. Only the availability of an appropriate set of sensors mounted on an artificial dexterous hand will allow the slave to perform "blind" (e.g. without direct visual feedback) tele-commanded explorations.

From a design point of view, it is important to note that the kinematics of the slave device could even be different from the master's one. The control of the slave in this case would be performed by introducing coordinate transformations. In the particular case the human control is obtained by using an instrumented glove, also the actuation system could be somewhat different from the slave actuation system: in this case a transformation between the master actuator space and the slave actuator space is needed.

3. Sensory requirements for a dexterous slave system

As discussed above, a primary need for a telemanipulation system is the presence of sensory systems located at the slave hand, and capable of extracting information about the contact conditions with the surrounding environment. These sensory systems, which allow the slave to be controlled during the execution of complex manipulation procedures, can be classified, according to the functional content of the information they extract, as:

a) **teleceptors,** which provide information about the remote place as a whole (artificial eyes and ears for long range action; proximity sensors for short range action, etc.);

b) **exteroceptors,** which detect information on the contact between the robot effector and the external environment (this category includes all the "skin" sensors);

c) **proprioceptors,** which sense position, orientation and relative movements of the various links of the robot effectors (angular joint rotation and internal force sensors);

d) **enteroceptors,** which monitor the functional conditions of the various mechanical and electronic components of the slave system.

That all these sensors are essential for the actual control of the whole telemanipulation procedure is easy to perceive by considering, for example, how fundamental is the skilled integration of visual and tactile/force sensing modalities for executing even simple manipulation procedures.

We intend to focus here our attention on categories b) and c) because these receptors are directly related to the hardware of the slave end-effector.

In general, although external and internal sensors (as exteroceptors and proprioceptors are commonly named in robotics) for space robotic end-effectors are based on the same principles of operation and on the same technologies as industrial robots, it must be taken into account that in the space environment some requirements on weight and size are critical.

Contact sensors (external force sensors and tactile sensors) play a very important role, among exteroceptors, on the slave hand. The ability to resolve the six components of the resultant forces and torques acting on the contact regions of the slave hand leads to a more accurate control of manipulative procedures (6). Force/torque resultant sensors can be positioned either at the wrist of the robotic hand or/and inside the distal phalanxes of each finger, being resolution improved while the sensor moves towards the fingertip. Besides determining contact force and torque, external force/torque resultant sensors provide also extremely useful information about the possible slippage of the manipulated object.

Tactile sensing can be considered as complementary to force sensing for the control of manipulation procedures. Although tactile sensing has been regarded so far in the field of robotics mostly as the artificial sensing modality devoted to determine pressure distribution over the contact regions of the end-effector, "tactile" sensing can actually provide a much wider and larger amount of information. Several technologies have been used to implement the former approach (6). At present, however, not only mechanical but also physical and chemical properties of the contact regions are considered as important and useful for perceptual purposes and for fine manipulation. Moreover, a "dynamic" approach to the analysis of tactile data is now being stressed, with particular emphasis to the control of exploratory and "blind" recognition procedures (7). Real dexterous behavior can result from a synthesis of force and tactile sensing. In fact external force measurements can be effectively combined with the detection of texture, local shape, roughness and "thermal properties" of the manipulated object in order to derive a more detailed description of the object and to more accurately control fine motion and force at the articulated slave hand.

Proprioceptive sensors have the function of indicating to the controller the relative position between the links of the slave hand. The knowledge, at any time, of the "joint vector" allows not only to implement pure position control procedure but also, in combination with internal force/torque information, the hybrid control of manipulation procedures.

In order to demonstrate the importance of providing a slave hand with an appropriate set of exteroceptive and proprioceptive sensory systems, we have implemented a set of simple exploratory procedures by utilizing a tendon actuated, anthropomorphic 4 degree-of-freedom finger equipped with joint rotation and torque internal sensors (8). External stimuli deriving from the operational space during contact between the finger and the environment are detected by an "epidermal" tactile sensor positioned at the fingertip. The same epidermal sensor, fabricated with a ferroelectric polymer film, could be also useful for detecting dynamic thermal properties of the contact regions.

4. Considerations on exteroceptive and proprioceptive feedback for the master hand.

Based on the assumption discussed in paragraph 2. of the isomorphic relation between the dexterous robotic slave hand and the human master hand, the problem of specifying the characteristics of the interfacing system has to be addressed. The functional operations required to this interface system are : a) to collect proprioceptive data from the master hand in order to command the slave operation , and b) to receive the exteroceptive information deriving from the slave and translating them into adequate feedback stimuli to the human hand.

Functions a) and b) must be performed by sensory and actuating systems positioned in contact with the human hand or with a deformable or rigid support wrapping the hand up. A clear example of such a structure is the already mentioned DataGlove (4), which incorporates joint angular rotation sensors but allows the hand to reach all possible kinematics configurations.

Manoeuvrability and ergonomic considerations are critical aspects in the design of the master telemanipulation system: these requirements are considerably emphasized in the case of telemicromanipulation, where the range of fine motion is very critical. For these reasons it is unlikely that the whole human master system could significantly differ morphologically and functionally from the human hand.

The system we have devised for the master hand consists of an instrumented glove possessing not only position sensors but also an actuating slave-commanded system for finger joints. The instrumented glove is depicted in Figure 2.

Kevlar tendons are routed along the back and the palm of the glove in order to actuate directly each phalanx according to a push-pull configuration. Tendon tension sensors, located at the wrist level, control the force-reflecting master-slave and slave-back-to-master procedures. Motors are also located remotely, in a structure beyond the wrist, in order to allow better hand manoeuvrability. An external glove protects the instrumented one and all Kevlar transmission tendons. Work is in progress for the realization of a prototype of the tendon-commanded glove. We must point out that, although the force-reflecting problem seems theoretically feasible, in practice several problems, derived from friction and real time coordination, could arise during the control phase.

An open problem for the realisation of a compact and compliant glove-like master device is the definition of the "actuating" or "stimulating" systems aimed at re-creating appropriate exteroceptive stimuli on the virtual contact regions of the human hand. A reasonable solution to this problem could be the use of local micro-actuators arrays capable of stimulating the human master's hand skin, according to a coherent spatio-temporal pattern. Other micro-actuators technologies (e.g. solenoid arrays, piezoelectric arrays or micromachined silicon active structures) have not been applied yet, owing probably to either volume or compliance constraints. Even feedback-to-master procedures for replicating "thermal" sensations could be implemented by available technology, should the dimensional vs. manoeuvrability problem find a practical solution.

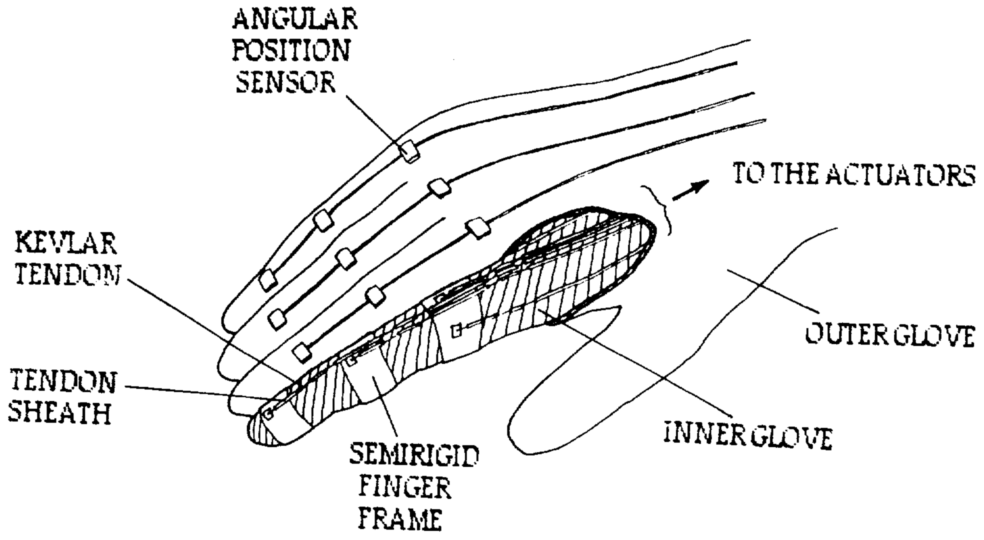


Figure 2. Scheme of the instrumented glove for fine telemanipulation

5. Conclusion

In this paper we have outlined the general problems of telemanipulation with emphasis on the particular case of fine manipulation tasks. In fact, we believe that this domain of applications, although extremely challenging, is going to be crucial to any wide diffusion of robotics teleoperation technology in space, and even elsewhere.

The problem of controlling very fine manipulation has to be addressed if, for instance, delicate assembly operations or remote handling of delicate samples have to be performed. In this class of operations, that will be increasingly important in space missions, a simple gripper will certainly not be sufficient, but even a multifingered hand will not be entirely useful if not equipped with adequate sensors.

An important aspect that we pointed out is that, although the use of joint rotation and torque sensors and of some contact sensors is an essential requirement for dexterous behavior, very fine manipulation requires, in addition, the use of true tactile sensors capable of discriminating very small surface indentation at the finger surface. For these operations the control of slippage will also be crucial; to this aim, perhaps even a sensitive force/torque fingertip sensor will not suffice, and skin-like distributed tactile sensors capable of sensing locally shear stress will be necessary.

Another important aspect of teleoperation, which to some extent comprises the same functional aspects of the problem, is tele-existence. In this field, sensing the tiny features of contact becomes a key part of the process of perceiving fully a remote reality. Measuring local indentation, and perceiving texture, thermal properties, compliance and other parameters of the touched object by dynamic exploration is a fundamental component of the process by which a human master operator can remotely "construct" a mental image of the environment which closely resembles the real one.

Based on the above considerations, we intend to address in depth in the near future the problem of "enriching" telesensations with information other than just vision. Teletactile sensing is a fundamental (even if not the only) part of the sensory information necessary to the master in order to "generate" a replica of the remote environment as faithful as possible. In this context, particular attention will be devoted to investigate issues of psychophysics, inherently associated with telesensation. Our approach to the problem of transmitting fine tactile sensations from the slave to the master has been outlined here. Further research, now in progress, will address this aspect more thoroughly, along with the key problem of using appropriate sensory-motor control techniques to extract dynamically tactile data by teleoperated exploratory procedures.

It may be worth pointing out, in conclusion, that telemicromanipulation can be very useful for a number of applications other than space. One of the applications we are investigating is, for instance, telemicrosurgery.

References

1. T. B. Sheridan :"Telerobotics", Proc. of the Workshop on Shared Autonomous and Teleoperated Manipulator Control (V. Hayward and A. Bejczy, Org.), 1988 IEEE Intern. Conf. on Robotics and Automation, Philadelphia, PA.

2. K. Salisbury :"Teleoperator Hand Design Issues", Proc. of 1986 IEEE Intern. Conf. on Robotics and Automation, San Francisco, CA.

3. S. E. Jacobsen et al. :"The Utah/MIT Dextrous Hand: Work in Progress", The Int. J. Robotics Res., Vol. 3, No. 4, 1984.

4. S. Fisher :"Telepresence master glove controller for dexterous robotic end-effectors", SPIE Vol. 726 Intelligent Robots and Computer Vision: Fifth in a Series, 1986.

5. R. F. Schmidt :"Fundamentals of Neurophysiology", Springer-Verlag New York Inc., 1978.

6. P. Dario, M. Bergamasco, A.S. Fiorillo :"Force and Tactile Sensing for Advanced Robots" in "Sensors and Sensory Systems for Advanced Robots", P. Dario (Ed.), Springer Verlag, Berlin, 1988.

7. M. R. Cutkosky, R. D. Howe :"Dynamic Tactile Sensing", Proc. of Ro.Man.Sy. Udine, Italy, 1988.

8. P. Dario, G. Buttazzo :"An anthropomorphic robotic finger for investigating artificial tactile perception", Int. J. Robotics Research, Vol. 6, Fall 1987.

FLIGHT EXPERIMENTS: SYSTEMS AND SIMULATORS

N90-29042

ROTEX—TRIIFEX: PROPOSAL FOR A JOINT FRG—USA TELEROBOTIC FLIGHT EXPERIMENT

G. Hirzinger
German Aerospace Research Establishment
DLR, Oberpfaffenhofen, FRG

A.K. Bejczy
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

ABSTRACT

The paper outlines the concepts and main elements of a RObot Technology EXperiment (ROTEX) proposed by DLR to fly with the next German spacelab mission, D2, in December 1991. It provides a 1-meter size, six-axis robot inside a spacelab rack, equipped with a multisensory gripper (force-torque sensors, an array of range finders, and mini stereo cameras). The robot will perform "assembly" and "servicing" tasks in a generic way, and will grasp a floating object. The paper focusses on the man-machine and supervisory control concepts for teleoperation from the spacelab and from ground, and explains the predictive estimation schemes for an extensive use of time-delay compensating 3D computer graphics.

A JPL-NASA proposal is underway to join ROTEX with a TeleRobotic Intelligent Interface Flight EXperiment (TRIIFEX), utilizing the functional and operational capabilities of ROTEX. The main objective of TRIIFEX is to extend performance and operation experience with hybrid position and force-reflecting control of telemanipulators to space telerobot missions, and to evaluate its human factors implications. JPL is planning to build a general-purpose computerized force-reflecting position control device backdriveable from robot hand sensors and a complementary graphics display of robot hand sensor data. The paper will include a brief description of the main elements of TRIIFEX, their interfaces to ROTEX, and the specific TRIIFEX objectives. TRIIFEX operation is planned from onboard the spacelab and from ground.

INTRODUCTION

Among the many areas important in space technology, automation and robotics (A&R) will become one of the most attractive ones for smaller countries like the Federal Republic of Germany, as well as for the big space nations. It will allow experiment-handling, material processing, assembly and servicing with a limited amount of manned missions, and it will provide an extensive technology transfer from space to earth applications. This is one of the main reasons why several activities towards space robotics have started in Germany with the long-term goal to make a major contribution to the space station, e.g., to the Man Tended Free Flyer (MTFF) subsystem.

In addition to study activities, DLR (the German Aerospace Research Establishment) made a proposal at the end of 1985 to fly a robot technology experiment ROTEX with the next "German" spacelab mission, D2, scheduled now for December 1991. Phase C/D is running now, with participation of two major German space technology companies, DORNIER and MBB, and including several of the leading German robotic research institutes. Thus ROTEX is a starting shot for a German participation in space automation and robotics, with a broad national basis.

A JPL-NASA proposal is underway to join ROTEX with TRIIFEX, utilizing the functional and operational capabilities of ROTEX. TRIIFEX employs hybrid position and force-reflecting master-slave control for telemanipulation. This control technique is the most efficient one for versatile telemanipulation in terrestrial applications; this control is standard in the nuclear industry. The reason for the efficiency of this control is twofold: (i) direct position control is inherent to these systems, and (ii) the operator's hand receives a genuine impression of acting forces and thereby is dynamically connected to the remote control task. However, system performance in this mode of control is closely coupled to the operator's body (manual) and mental (model reference) performance capabilities. The basic question TRIIFEX is asking is: how can this control technique be extended to space efficiently and safely? In particular: (i) How does microgravity affect *on-board* operator's performance? (ii) How will *ground* operator relate to control actions in micro-g from control inputs in normal-g, in particular, in the presence of a several-second R/T communication time delay? The first specific question is related to the operator's neuromotor response characteristics in micro-g. The second question is related to the operator's *psychomotor* response characteristics when control actions are across basically different dynamic environments and across time delay.

The first part of the paper is devoted to the description of ROTEX, and the second part briefly summarizes the main elements of TRIIFEX and interfaces to ROTEX.

THE ROTEX PROJECT

The ROTEX system contains several items:

- A small, six-axis robot (work space 1m) inside a space-lab rack (Fig. 1). The robot arm will be built by DORNIER company. Its gripper, built by DLR, will be provided with a number of sensors (Fig. 2): two six-axis force-torque wrist sensors, a tactile array in each finger for grasping force control, an array of nine laser-range finders, and a tiny stereo camera (smaller than a match-box) to provide a stereo image out of the gripper. In addition, a fixed pair of cameras will provide a stereo image of the robot's working area.
- The robot is able to perform automatic, preprogrammed motions as well as teleoperated motions via an astronaut on board or an operator on ground (Fig. 2).
- Two types of operational modes will be performed by the robot:
  a) Experiment handling. This is a slow or "micro-gravity (μg) mode" based on the execution of preprogrammed paths that may be reprogrammed from ground.

b) Servicing. This is a fast mode based on teleoperation on board and from ground, and on sensor-based learning of tasks on ground which are executed automatically on board.

- The main goals of the experiment are

  a) To verify joint control (including friction models) under zero gravity, as well as μg motion planning concepts, based on the requirement that the robot's accelerations while moving must not disturb any μg experiments nearby.

  b) To demonstrate and verify the use of advanced six dof hand controllers under zero gravity.

  c) To demonstrate the combination of a complex, multisensory robot system with powerful man-machine interfaces (such as 3D computer graphics, control balls, force-reflecting hand-controllers, stereo imaging, voice input-output) that also allow for teleoperation from ground.

In order to demonstrate servicing capabilities by teleoperation, three basic tasks are envisioned:

a) Assembling a mechanical grid structure (Fig. 3).

b) Connecting/disconnecting an electrical plug (which stands for replacement of an ORU).

c) Grasping a floating object.

For all these tasks, continuous or on-line sensory feedback is involved.

Multisensory Robot Gripper

Multiple sensing in the robot gripper and sensory feedforward in the man-machine interface are the key for the telepresence concepts envisioned. The gripper sensors involved belong to the new generation of DLR robot sensors with all analog preprocessing and digital computations performed inside the sensors or at least in the robot's wrist (Fig. 4). Using a high-speed serial bus, only two signal wires come out of the gripper (carrying signals of forces-torques and distances), augmented by two 20 kHz power supply wires from which the sensors themselves derive their DC power supply voltages via tiny transformers. The following sensor modules are provided:

a) An array of nine laser range finders based on triangulation: one "big" sensor (half the size of a match box) switchable into a scanning mode for a longer range of $\approx$ 3-50 cm, and four smaller ones in each finger for shorter ranges of 0-3 cm. The range finders are the result of more than five years' development aiming at a precise performance over a remarkable range and independent of the slant angle and surface of the measured object. One of the main problems to be solved in this development was the design of a nonlinear digital control system that adapts the light transmitter's intensity depending on the reflected light intensity. The signal control system now used varies the emitted power in a range of 1 to 10,000 within 10 μsec. This indeed enables the sensors to measure distances with respect to surfaces that show up strongly with quickly changing reflection characteristics.

b) A "stiff" six-axis force-torque sensor based on strain-gauge measurements and a "compliant" optical sensor (Fig. 5). Originally, it seemed necessary to make a selection between these two sensing principles. A solution was found that combines both principles in one compact sensor with the option to switch between them during operation. The "compliant" optical force-torque sensor consists of an inner and an outer part (Fig. 5). The basic measuring arrangement in the inner ring is composed of an LED, a slit and, perpendicular to it, a linear position sensitive detector (PSD) which is mobile against the remaining system. Six such arrangements (rotated by 60 degrees each) are mounted in a plane, whereby the slits alternately are vertical and parallel to the plane. The ring with PSDs is fixed inside the outer part and connected by springs to the LED slit basis. The springs bring the outer part back to neutral position when no forces/torques are exerted.

c) A tactile array of four by eight sensing dots in each finger using elastomeric rubber as transducer.

d) A pair of tiny stereo cameras, augmented with an additional pair of stereo cameras which is fixed in the rack, yielding a global view of the work space.

e) The sensor or control ball as a six dof hand controller. For a very natural six degree-of-freedom control of robots and of 3D computer graphic objects by using only one human hand, DLR developed different types of plastic hollow balls with six-axis force-torque sensors inside [3,4]. The latest and preferred version uses the compliant sensor (Fig. 5) inside the ball. The only difference between the wrist sensor and the control ball is that the outer ring in Fig. 5 is replaced by a plastic hollow ball.

Sensory Feedback Structures

The use of sensors in the feedback control is based on a sensor-based fine motion planning concept that has been outlined in different papers (e.g. [7]). Its main features are briefly as follows (see also Fig. 6). "Rudimentary" commands are derived either on-line from a human teacher operating the control ball or from a path-generator connecting preprogrammed points. They are interpreted in a dual way as force/torque or positional/orientational commands. When the robot moves in free space, the ball forces are transformed into translational commands; when the robot senses contact with the environment, it takes the ball inputs as nominal force values and, by closing the sensory loop at the robot's site (see Fig. 2), it always exerts only those forces which are given by the human operator [8]. Of course, any kind of shared control between robot and operator is feasible. Though the forces are not fed back to the human arm (as in "bilateral" force control), the operator is sure that the robot is fully under his control and he easily may lock up doors, assemble parts or plug in connectors. In other words, the human operator (via stereovision and 3D graphics) is enclosed in the feedback loop on a high level but with low bandwidth, while the low-level sensory loops are closed on-board at the robot directly with high bandwidth. Thus a supervisory control technique is envisioned that permits shifting more and more autonomy to the robot while always offering real-time human interference (Fig. 7).

Visual Feedback and Predictive Control

In teleoperation on-board the spacelab, the visual display is restricted to the use of a small colour TV monitor. In the present state of planning, the B/W stereo images produced by the gripper or the global cameras are displayed alternately to the operator, who will use shutter glasses (developed in German nuclear power facilities with only 15V power supply and switching frequencies up to 1 kHz) to obtain stereo perception of images. The sensory information will be added in simple bar-like form at the monitor's edges.

For teleoperation from ground the situation is different: much more powerful equipment is available there for visual feedback, but the communication link restrictions are obvious. Indeed, it turned out that the "normal" spacelab up-links as used until now are not at all adequate for telepresence ideas. They would create up-link delays of up to 15 seconds, partly caused by data checks in Houston. This seemed to destroy the ground teleoperation concept completely. The present base-line uses the Text And Graphics channel (TAG) for the up-link, eliminating these difficulties. This channel uses the TDRS satellite, and could not be tested until now. Using the TAG channel, the up-link command rates are in the range of 2 kbit/sec, assuming a sampling rate of 20 Hz. Nevertheless, we have to take into account an overall delay of four seconds in the loop closed at the ground station. In order to get exact knowledge about this delay, we will provide the ball commands with a code which, when arriving at the robot, are packed into the down-link information.

The down-link information comprises a sequential RGB video signal. The left and right black-and-white stereo images are packed into the red and green channel. They are superimposed and displayed on a polarized screen on ground. The down-link data channel also contains all internal (position encoders) and external sensory signals so that on a 3D graphics monitor the robot's position is displayed as well as all sensor data. Preferably, a stereo graphics system is used with real-time volume-shaded representation of the workcell.

The big problem for teleoperation from ground is the communication time delay. The only way to compensate for it is by using predictive computer graphics. Extensive use of them will be made in ROTEX. Fig. 8 shows that the human operator at the remote work-station handles the six dof hand controller by looking at a "predicted" graphics (e.g., wire frame) model of the robot. The control commands issued to this instantaneously reacting robot simulator are sent to the remote robot as well, using the time-delayed transmission links. Now the ground-station computer and simulation system contains a model of the up-link and down-link delay lines as well as a model of the actual states of the real robot and its environment. Note that we have several alternatives to superimpose the predicted robot model (augmented by predictions of any other moving parts) with other information representations:

a) The presently received (of course, delayed) TV stereo or mono image in case the globally fixed camera pair is active.

b) The "delayed" graphics image derived from this delayed TV image (including the case of hand-mounted cameras) and other sensory data.

c) The actual graphics image as derived from the state space model of robot and environment.

There is not yet a final conclusion on what the most efficient method of superposition would be. There is, of course, evidence that the most crucial problems lie in the derivation of "output data" (e.g., positions/orientations of moving objects) from stereo images and range finders. As real-time is required, this is an extremely challenging preprocessing problem solved by a parallel transputer system but not discussed in more detail here.

For the robot, we assume a linearized Cartesian state space model $\underline{x}_{k+1} = \underline{A}\underline{x}_k + \underline{b}\underline{u}_k$. In the case of grasping a floating object, this model in standard form of digital control theory not only describes the Cartesian robot dynamics, but also the dynamics of the free-flying part.

Thus the left part of Fig. 8 is just a prediction of the robot's present estimated state $\hat{\underline{x}}_k$ to the future state $\hat{\underline{x}}_{k+n_u}$; n_u is the up-link delay time expressed as a multiple of the sampling period, that makes up one delay d. This predicted state is the state to which the presently issued hand controller command has to refer. But the more interesting part is the estimator on the right half of Fig. 8. It compares the measured, but down-link-delayed output data $\underline{y}_{k-n_d}$ (the robot's positions and orientations) to the output data $\hat{\underline{y}}_{k-n_d}$ from the robot model running through the down-link-delay computer model (n_d is the number of sampling periods in the down-link delay). The estimator's detailed structure has been derived in [9]. For telemanipulation from ground in case of an assembly operation and for sensor-based task learning on ground, a realistic graphic simulation of the workcell and the robot's sensory perception is the crucial item. Fig. 9 shows the envisioned structure for telemanipulation from ground with simulated sensory path refinement.

TRIIFEX PROJECT

The key element in the TRIIFEX project is the use of a Force Reflecting Hand Controller (FRHC) to control the robot both from an on-board control stand and from a ground control station. The planned FRHC is not a geometric replica of the robot arm; it is a generalized position input and force feedback device, tailored to the operator and to the control station, and applicable to different manipulators. The generalized FRHC technique has been described in detail elsewhere [11]. The device planned for TRIIFEX is somewhat different from the one described in [11] for packaging reasons; it will have an elbow instead of a telescoping linear link.

The use of a generalized (non-replica) FRHC device also represents a new control configuration: (i) force feedback is referenced to wrist force-torque sensor information, and (ii) the control requires a computer for coordinate transformations and sensor data handling. The sensor data will be displayed on a dedicated graphics display. The planned on-board TRIIFEX system is shown schematically in Fig. 10. This figure emphasizes the on-board TRIIFEX electronics architecture and its interface to the ROTEX electronics.

The performance capabilities and characteristics of a generalized FRHC laboratory system at JPL are described elsewhere in this conference proceedings [12-13]. The on-board experiments are planned to be identical to the ROTEX experiments.

The TRIIFEX ground station system is schematically shown in Fig. 11. A key component in the TRIIFEX ground system is the use of a "Phantom Robot," which is a high-fidelity 3D graphics image of the real robot, superimposed on the 3D TV image of the real robot in the workcell. The operator interacts with the Phantom Robot in real time. Thus, the motion of the Phantom Robot on the TV monitor screen acts as a predictive display in a real work environment shown on the TV screen. The motion of the real robot image will follow the motion of the Phantom Robot image after some time delay. The contact closure actions will be referenced to local F/T sensor data and will be controlled locally through the F/T sensor data upon the operator's initialization commands. The operator's responsibility here is the verification of the status of the real robot versus the Phantom Robot before the closure action is initiated so that there is a certainty that the local control algorithm can complete the task. Again, the TRIIFEX ground experiments are planned to be identical to the ROTEX experiments.

The general objective of the TRIIFEX project is to validate and quantify force-reflecting position control technology for Earth-orbital space missions. The planned performance measurements are focussed on human operator's performance capabilities. They are aimed to evaluate (i) on-board operator's ability to use force-reflecting position control of a telemanipulator in microgravity, and (ii) ground operator's ability to use this technique for telemanipulation in microgravity from a normal gravity base under several-second R/T communication time delay.

An expected major benefit of the TRIIFEX project is the evaluation of the validity of ground simulation data of microgravity telemanipulation by comparing flight experiment data to data obtained through ground simulation of the same experiments.

CONCLUSION

The ROTEX proposal is a first step of Germany's engagement in space robotics aimed at the demonstration of a fairly complex system with a multisensory robot on board and human telerobotic interference that makes use of sensor-based six dof hand controllers, new concepts for predictive 3D computer graphic and stereo display. Teleoperation from ground is a very challenging technique that forces us to move even more strongly toward on-board autonomy. The planned control strategy is to move the human operator increasingly towards supervisory control without changing the control loop structures.

The TRIIFEX proposal complements the ROTEX proposal by providing alternative man-machine interface devices and techniques in order to broaden the knowledge base for human-control performance capabilities for space telemanipulation.

ACKNOWLEDGMENT

This work was partly carried out at the Jet Propulsion Laboratory of the California Institute of Technology under contract to the National Aeronautics and Space Administration.

REFERENCES

[1] Bejczy, A.K.: Smart sensors for smart hands. AIAA paper No. 78-1714, AIAA-NASA Conf. on "Smart" Remote Sensors, Hampton, VA, Nov. 1978, pp. 14-16.

[2] Lee, S., G. Bekey, and A.K. Bejczy: Computer control of space-borne teleoperators with sensory feedback. Proc. of IEEE Intl. Conf. on Robotics and Automation, St. Louis, MO, March 25-28, 1985, pp. 205-214.

[3] Heindl, J., and G. Hirzinger: Device for programming movements of a robot. U.S. Patent No. 4,589,810, May 20, 1986.

[4] Hirzinger, G., J. Dietrich: Multisensory robots and sensor-based path generation. IEEE Intl. Conf. on Robotics and Automation, San Francisco, April 7-10, 1986.

[5] Craig, J.J., and M.H. Raibert: Hybrid Position/Force Control of Manipulators. Transactions of the ASME, Journal of Dynamic Systems, Measurements and Control, Vol. 102, (6/1982), pp. 126-132.

[6] Mason, M.T.: Compliance and force control for computer controlled manipulators. IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-11, No. 6 (1981), pp. 418-432.

[7] Hirzinger, G., and K. Landzettel: Sensory feedback structures for robots with supervised learning. IEEE Intl. Conf. on Robotics and Automation, St. Louis, MO, March 1985.

[8] Hirzinger, G., and J. Heindl: Sensor programming, a new way for teaching a robot paths and force torques simultaneously. Third Intl. Conf. on Robot Vision and Sensory Controls, Cambridge, MA, Nov. 7-10, 1983.

[9] Hirzinger, G.: Predictive and estimation schemes in sensor-controlled telerobotics. NATO Advanced Research Workshop "Estimation Theory in Sensing Systems for Improved Performance," Playa d'Oro, Spain, Oct. 6-9, 1987.

[10] Sheridan, T.B.: Human supervisory control of robot systems. IEEE Intl. Conf. on Robotics and Automation, San Francisco, CA, April 7-10, 1986.

[11] Bejczy, A.K., and J.K. Salisbury, Jr.: Controlling remote manipulators through kinesthetic coupling. *Computers in Mechanical Engineering (CIME)*, Vol. 2, No. 1, July 1983, pp. 48-60.

[12] Szakaly, Z., W.S. Kim, and A.K. Bejczy: Force-reflective teleoperated system with shared and compliant control capabilities. Proc. of NASA Conf. on Space Telerobotics, Pasadena, CA, Jan. 31-Feb. 2, 1989.

[13] Hannaford, B., and L. Wood: Performance evaluation of a six-axis high fidelity generalized force-reflecting teleoperator. Proc. of NASA Conf. on Space Telerobotics, Pasadena, CA, Jan. 31-Feb. 2, 1989.

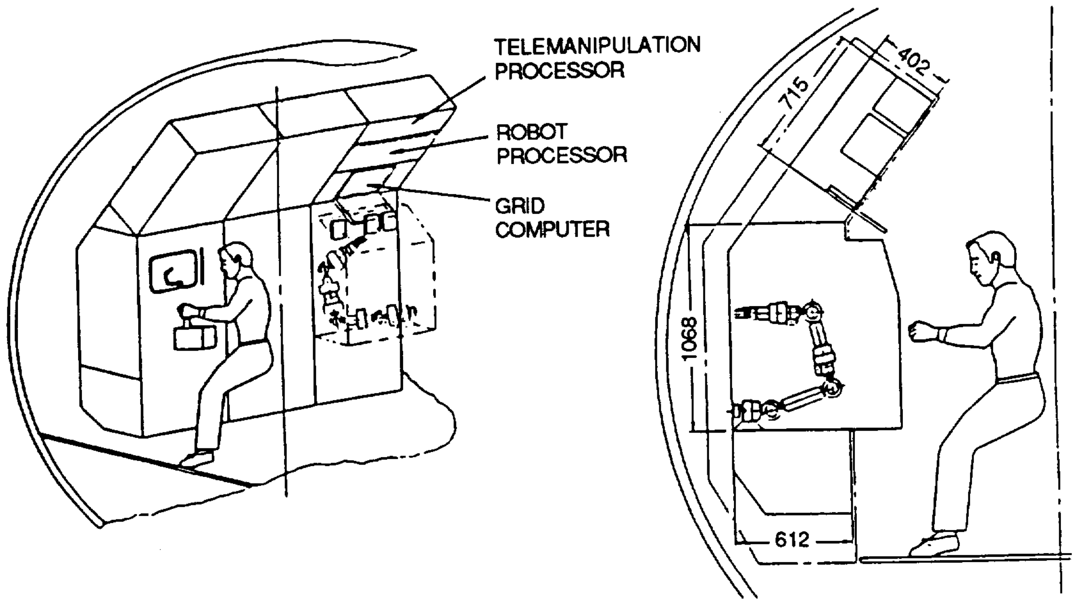


Fig. 1 Robot in the spacelab-rack

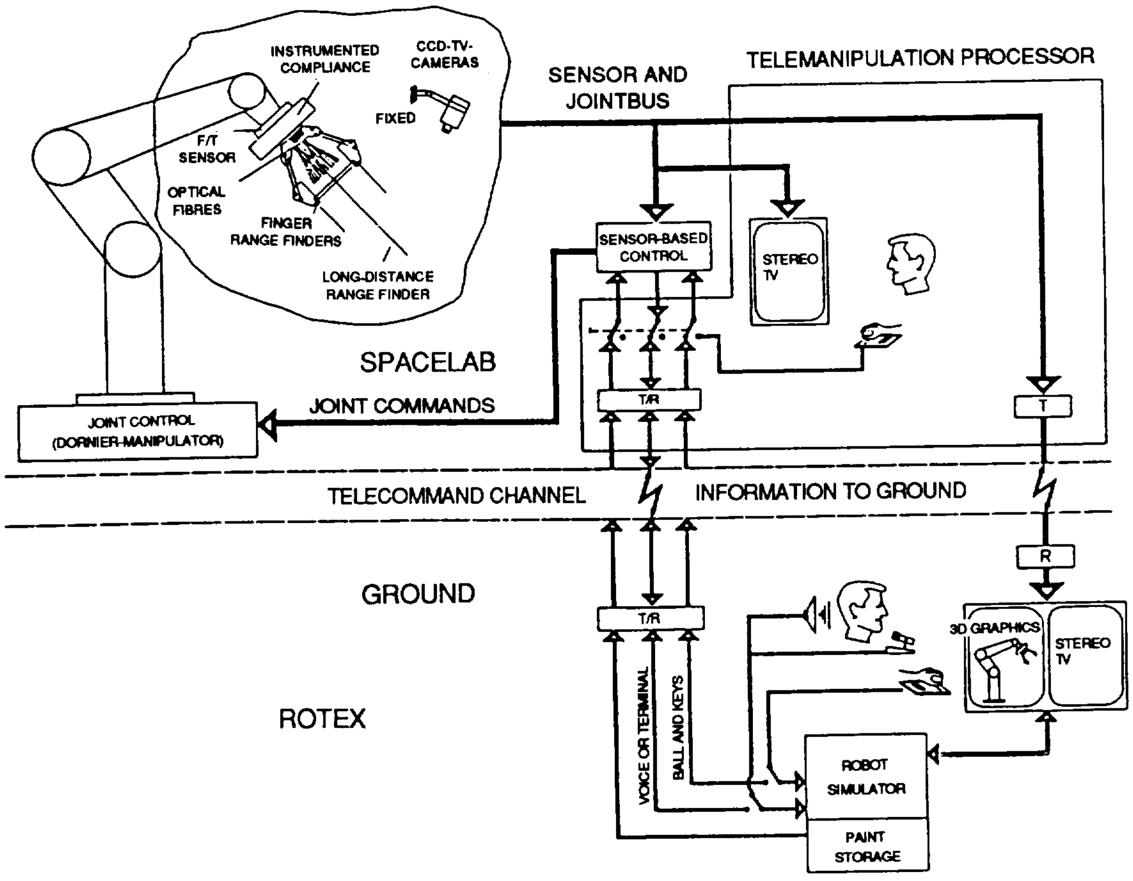


Fig. 2 Schematic Representation of ROTEX

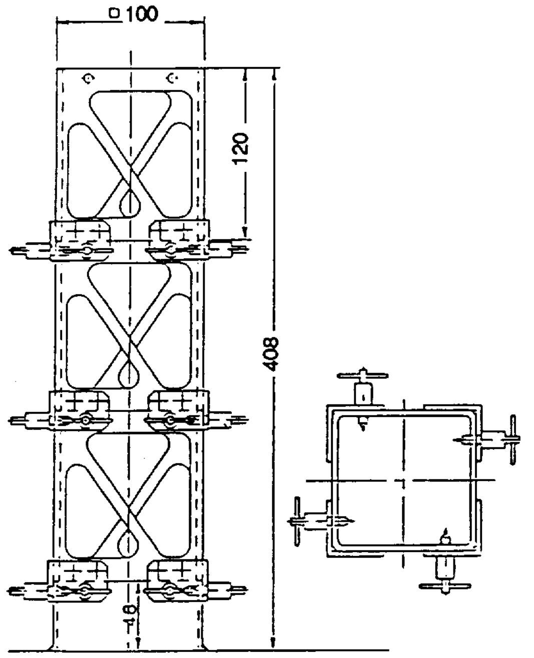


Fig. 3 Mechanical grid structure to be assembled by the robot

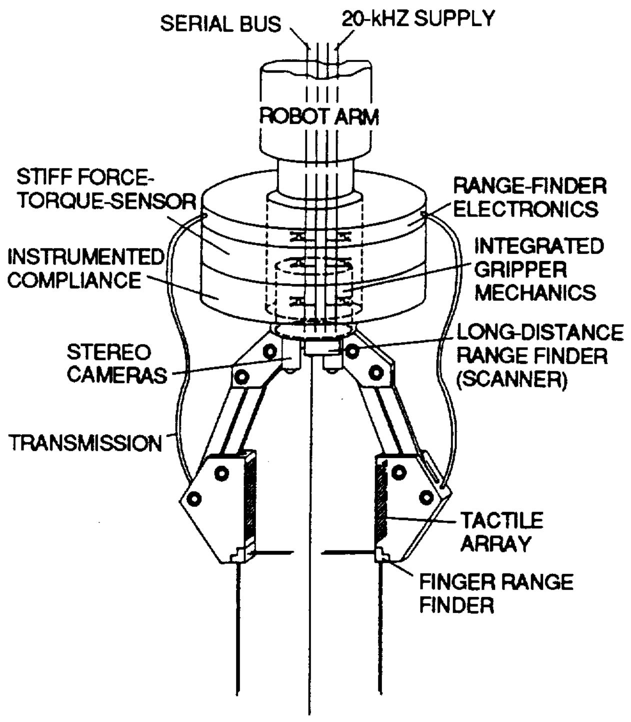
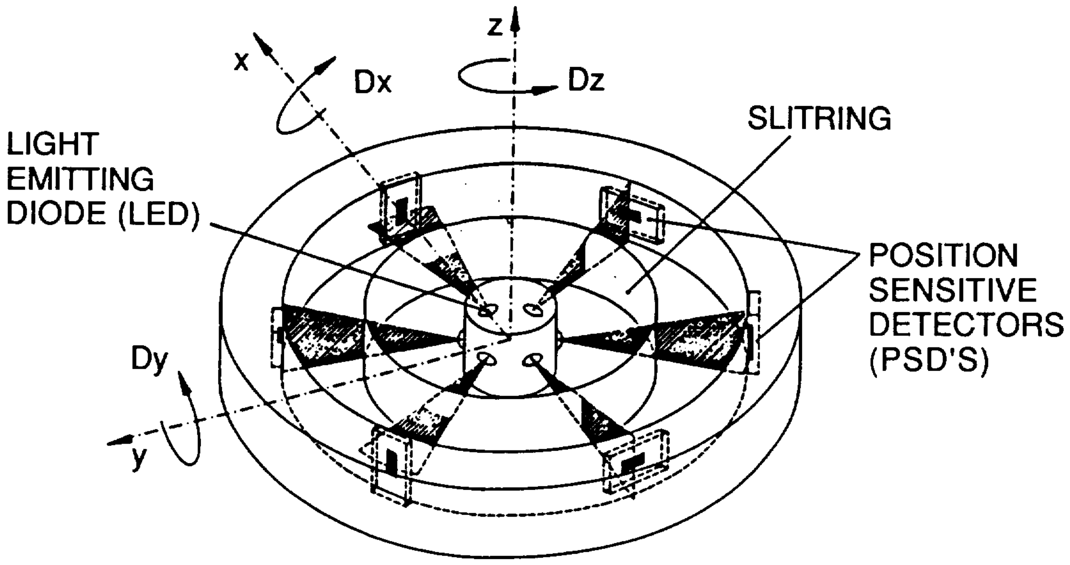


Fig. 4 Schematic arrangement of sensors in the gripper



z
x
Dx
Dz
SLITRING
LIGHT EMITTING DIODE (LED)
POSITION SENSITIVE DETECTORS (PSD'S)
Dy
y

Fig. 5 Compliant optical 6-axis force-torque sensor

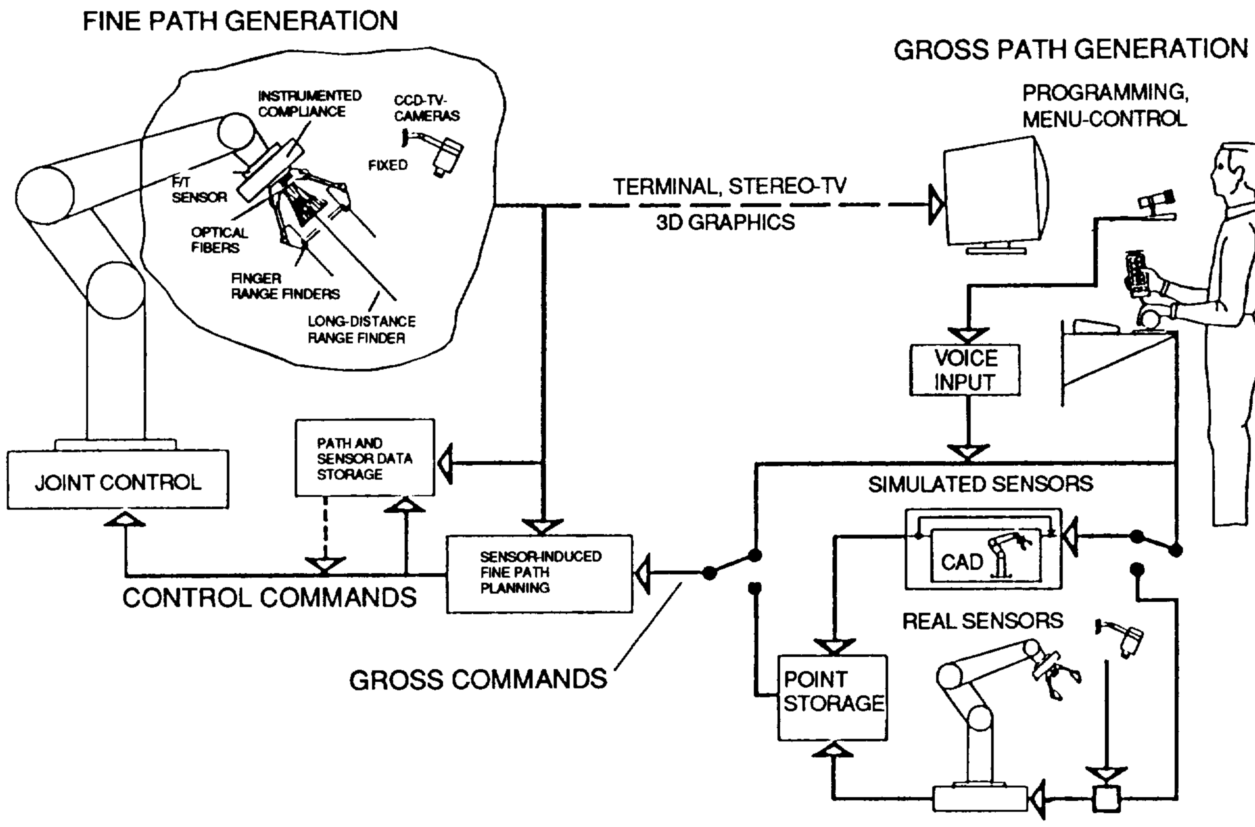


Fig. 6 DLR's telerobotic concept

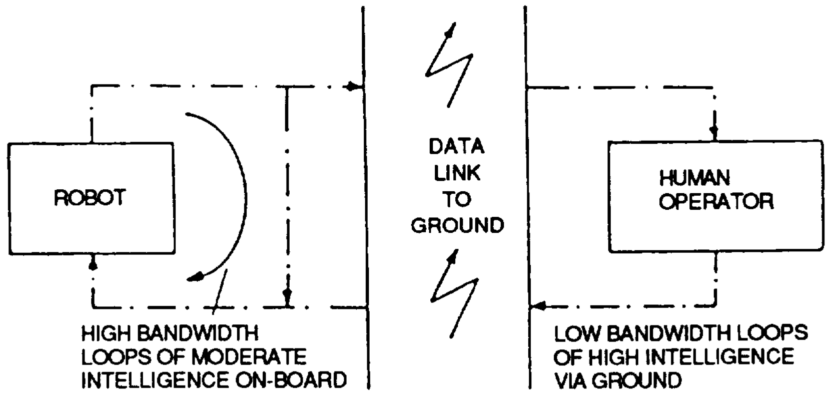

ROBOT
DATA LINK TO GROUND
HUMAN OPERATOR
HIGH BANDWIDTH LOOPS OF MODERATE INTELLIGENCE ON-BOARD
LOW BANDWIDTH LOOPS OF HIGH INTELLIGENCE VIA GROUND

Fig. 7 Supervisory control-concept

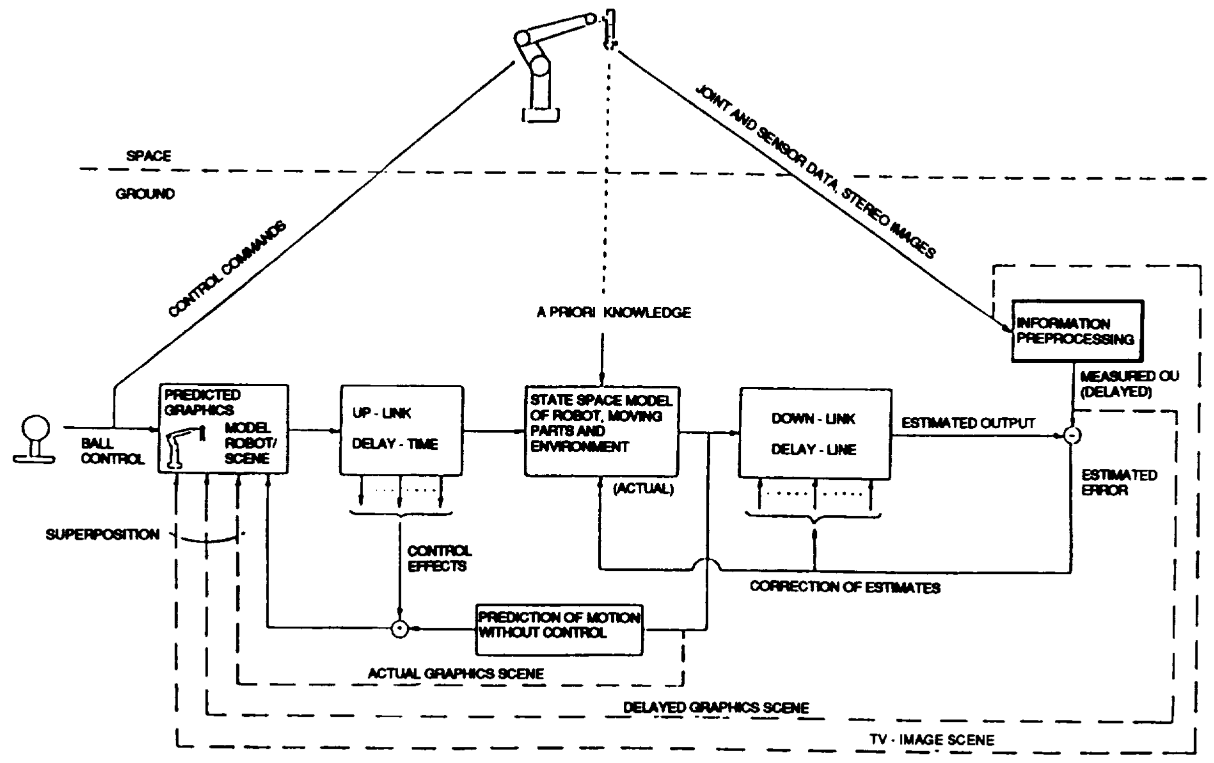


Fig. 8 Block structure of predictive estimation scheme

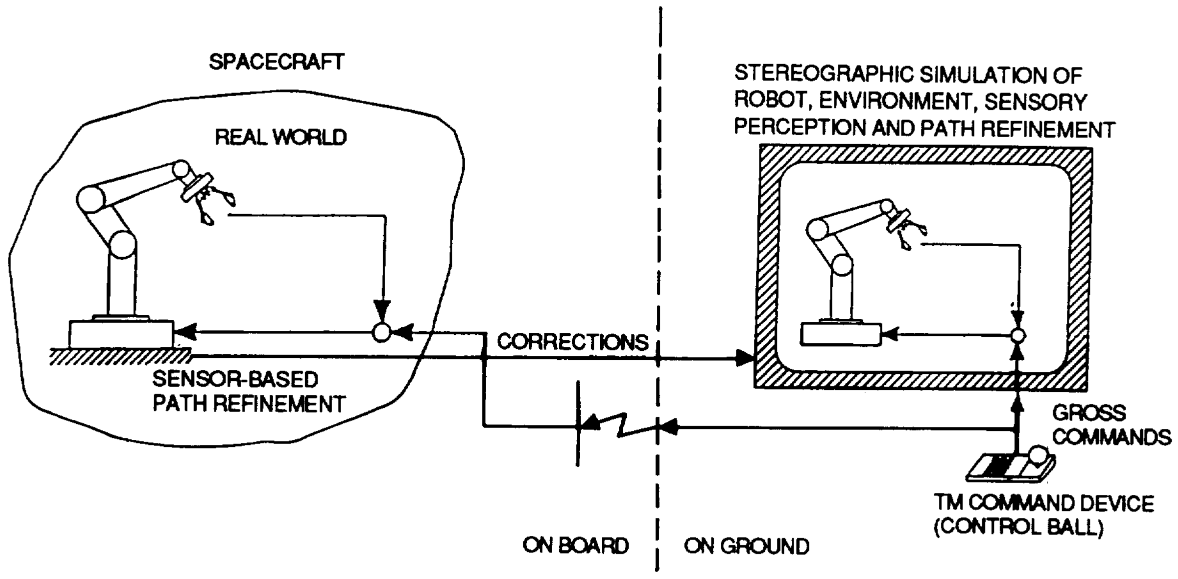

SPACECRAFT
REAL WORLD
SENSOR-BASED PATH REFINEMENT
CORRECTIONS
STEREOGRAPHIC SIMULATION OF ROBOT, ENVIRONMENT, SENSORY PERCEPTION AND PATH REFINEMENT
GROSS COMMANDS
TM COMMAND DEVICE (CONTROL BALL)
ON BOARD
ON GROUND

Fig. 9 Presimulation of sensory perception and path refinement in case of teleoperation from ground

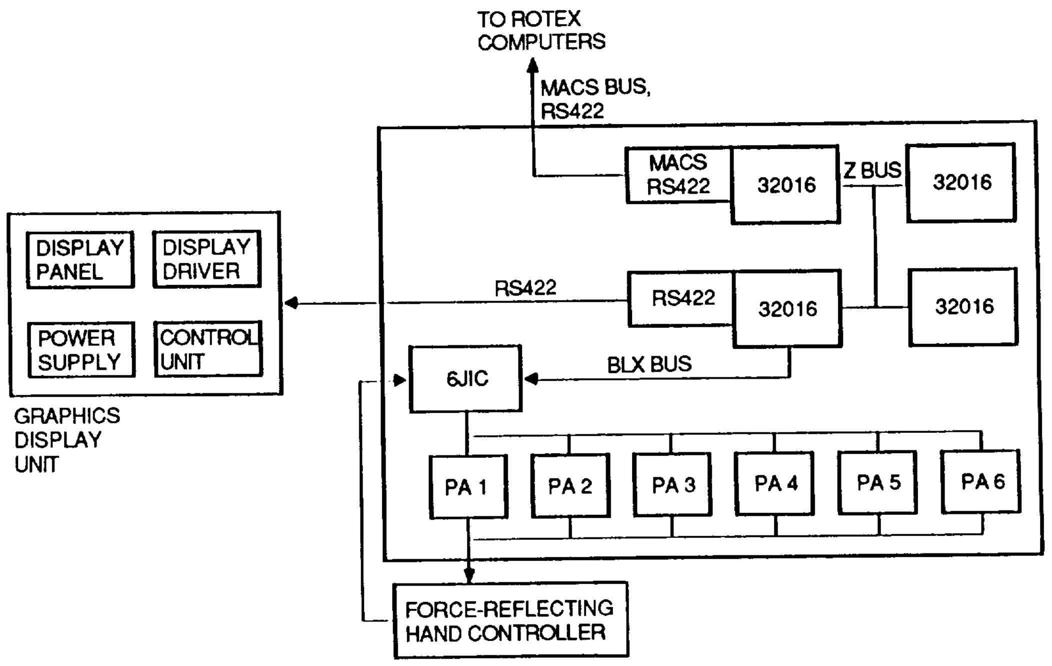


Fig. 10 TRIIFEX on-board system schematics

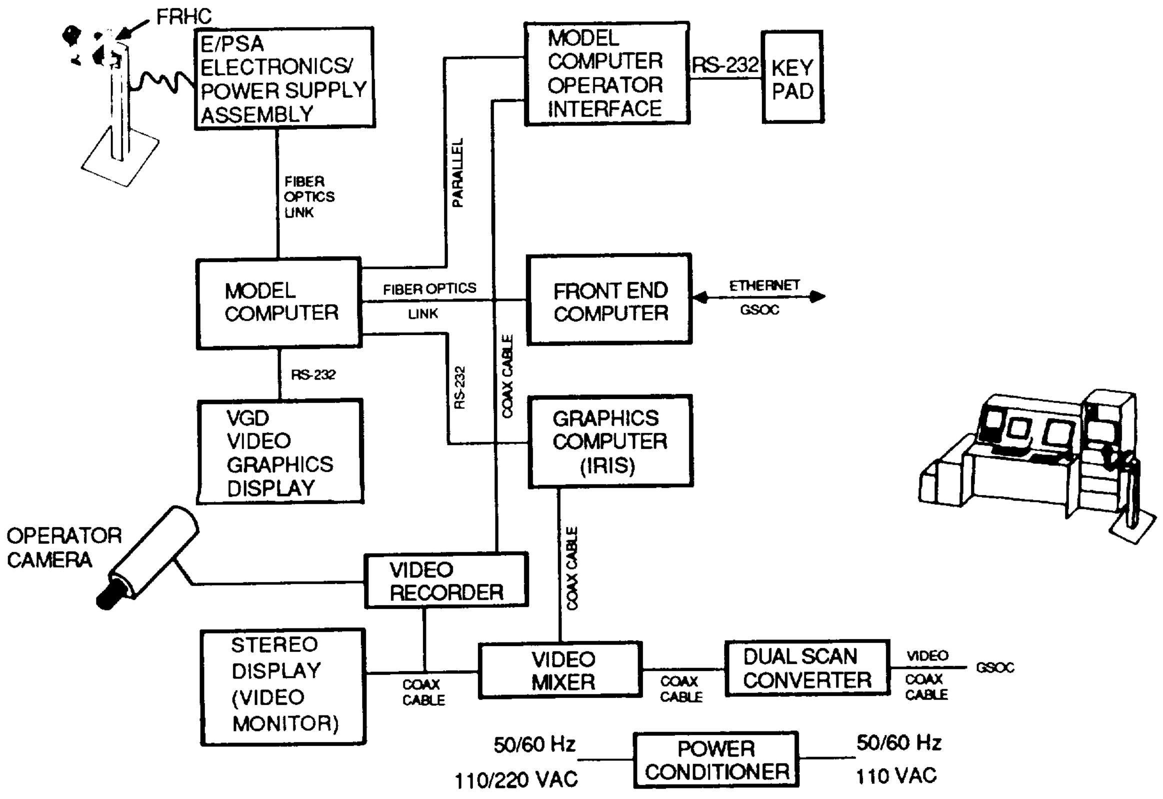


Fig. 11 TRIIFEX ground system schematics

N90-28343

TEST AND TRAINING SIMULATOR FOR GROUND-BASED TELEOPERATED IN-ORBIT SERVICING

Bernd E. Schäfer

German Aerospace Research Establishment (DFVLR)
D-8031 Oberpfaffenhofen, Federal Republic of Germany

Abstract

For the Post-IOC-Phase of COLUMBUS it is intended to use robotic devices for the routine operations of ground-based teleoperated In-Orbit Servicing. A hardware simulator for verification of the relevant in-orbit operations technologies, the Servicing Test Facility, is necessary which mainly will support the Flight Control Center for the Manned Space-Laboratories for operational specific tasks like system simulation, training of teleoperators, parallel operation simultaneously to actual in-orbit activities and for the verification of the ground operations segment for telerobotics. This paper describes the present status of definition for the facility functional and operational concept.

1. Introduction

In-Orbit Servicing has emerged as one of the paramount features of the COLUMBUS program, which is the European contribution to the International Space Station, to establish an open-end orbital infrastructure. It comprises maintenance, repair, supply, configuration change and experiment handling of the COLUMBUS elements and payloads. The Man-Tended Free Flyer (MTFF) is the basic element for the need of servicing inside and outside the module by telemanipulation and autonomously operated robotic devices. Simulation of servicing procedures is therefore a vital step in the direction of actual orbital operation to assure feasibility and reliability of procedures and to establish strategies involving a wide spectrum of eventualities.

In the current planning of ESA (*European Space Agency*) for the initial operational phase of the COLUMBUS elements APM (*Attached Pressurized Module*) and MTFF, the so-called IOC-Phase (*Initial Operational Capability*), it is not foreseen to perform In-Orbit Servicing by ground-based telemanipulations. In this phase, servicing tasks will be performed by astronauts in situ, either by extra-vehicular activities or with the aid of HERA, the Hermes manipulator arm, which will be operated by the crew on-board Hermes, the European orbiter. This servicing scenario will take place twice a year, each servicing mission requiring about 7 to 10 days of duration.

MTFF-internal manipulators will not be foreseen within the IOC-Phase. Correspondingly, within this phase the present MTFF operational concept does not consider ground-based teleoperated in-orbit servicing with the aid of robotic devices. On the other hand, since the last two or three years, ESA has established different studies and projects for technology development and demonstration for the use of Automation and Robotics (*A&R*) inside and outside the MTFF and the APM:

- EMATS (*Equipment Manipulation and Transportation System*) and EMS (*Experiment Manipulator System*) for internal robotic servicing [1,2],
- SMS (*Service Manipulator System*) technology like HERA for external servicing [3,4],
- BIAS (*Bi-Arm Servicer*) for internal and external servicing with two co-operative manipulators [5],

- ROSSA (*Robotics Spacecraft Servicing and Assembly*), analyzing a mission scenario of mostly automated payload operations by means of A&R within MTFF and APM, starting a few years after the begin of the COLUMBUS IOC-Phase [6].

Figure 1 illustrates different design approaches for MTFF-internal manipulators, being presently investigated in more detail by different ESA studies [1,6]: a) rack-external devices like *the single- or the multi-rack robot*, which are decentralized handling devices to serve one or several experiments, b) *the central transport robot*, which is a decentralized and general purpose handling device to serve all experiments, e.g. EMATS. In Figure 2 two different approaches for external manipulator design, partly still based on SMS technology [7], are sketched (both of having seven degrees of freedom): The first one shows one of the latest version of Fokker's relocatable HERA design [3], largely a symmetrical manipulator of about 11.5 m length with identical end effectors at both ends thus being able to walk over from Hermes to MTFF or even relocate on the MTFF and henceforth being permanently MTFF-based. The other one shows a manipulator concept studied by MBB/ERNO [4], the total length being 10.6 m. For comparison, ESA's former manipulator approach for external servicing, the SMS [7], had a total length of roughly 7.5 m. More recently, investigations about the operational working volume of an MTFF-based manipulator favourize travelling concepts guided either by a linear or a circular rail mounted on the exterior of the MTFF thus providing an additional degree of freedom for servicing operations [8].

Rather than in the IOC-Phase, in the so-called Post-IOC-Phase of COLUMBUS (formerly AOC, *Autonomous Operational Capability*) which will start about 2 to 4 years after IOC begins, it is intended to make use of the A&R technology to be developed. ESA's robotic technology programme, briefly described above, aims at this goal, also the German *Robotics Technology Experiment, ROTEX*, [9] which will be flown by the Space Shuttle on the next D2-mission, presently scheduled for late 1991. A basic objective of all these technological initiatives is to perform in-orbit robotic servicing during un-manned phases by telemanipulation from ground (cf. also [10,11]). The main reasons to do so are to reduce costs and risks for both the transportation and the operations and control as well as for the astronauts.

2. The Servicing Test Facility within the MSCC

Based on the ESA council meeting on ministerial level (Nov 10-11, 1987, The Hague), ESA entrusted DFVLR with the conductance for operating the COLUMBUS manned space laboratories: in case of APM, under NASA leadership, the *German Space Operations Center* (GSOC at the DFVLR site at Oberpfaffenhofen) will be responsible for the payload operations, in case of MTFF the responsibility for the operations of the complete system and the payloads (excluding Hermes visits) lies within GSOC management [12], execpt in those cases where the MTFF is within the operational command and control zone (CCZ) of the International Space Station *FREEDOM*. The planning and set-up of the *Manned Space Laboratories Control Center, MSCC*, has already started; its operational readiness is scheduled for 1991 since the German D2-mission should already be operated by the new complex.

For in-orbit operations technology the APM and the MTFF flight control center will be supported by three test facilities, the *In-Orbit Operations Simulation Facilities, IOSF*, which will be installed within the MSCC [13]:

- An European Proximity Operations Simulation (EPOS) Facility,
- A Servicing Test Facility (STF),
- A Test Facility for Large Flexible Spacecraft Control.

This ground infrastructure will be developed by the DFVLR at Oberpfaffenhofen with national and European fundings. The facilities will be used for technology development and space system operation of the future European space programs.

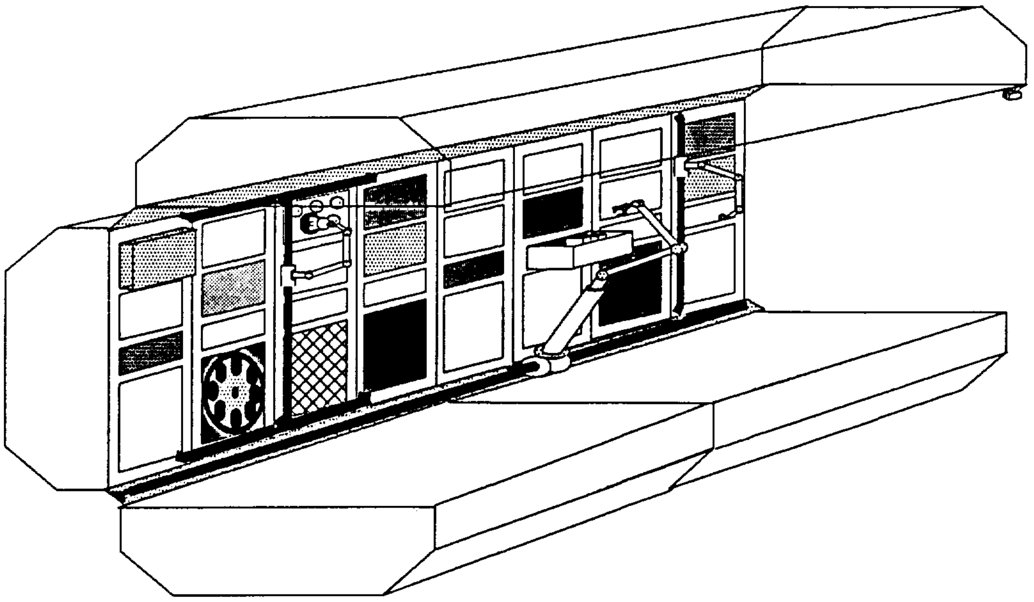
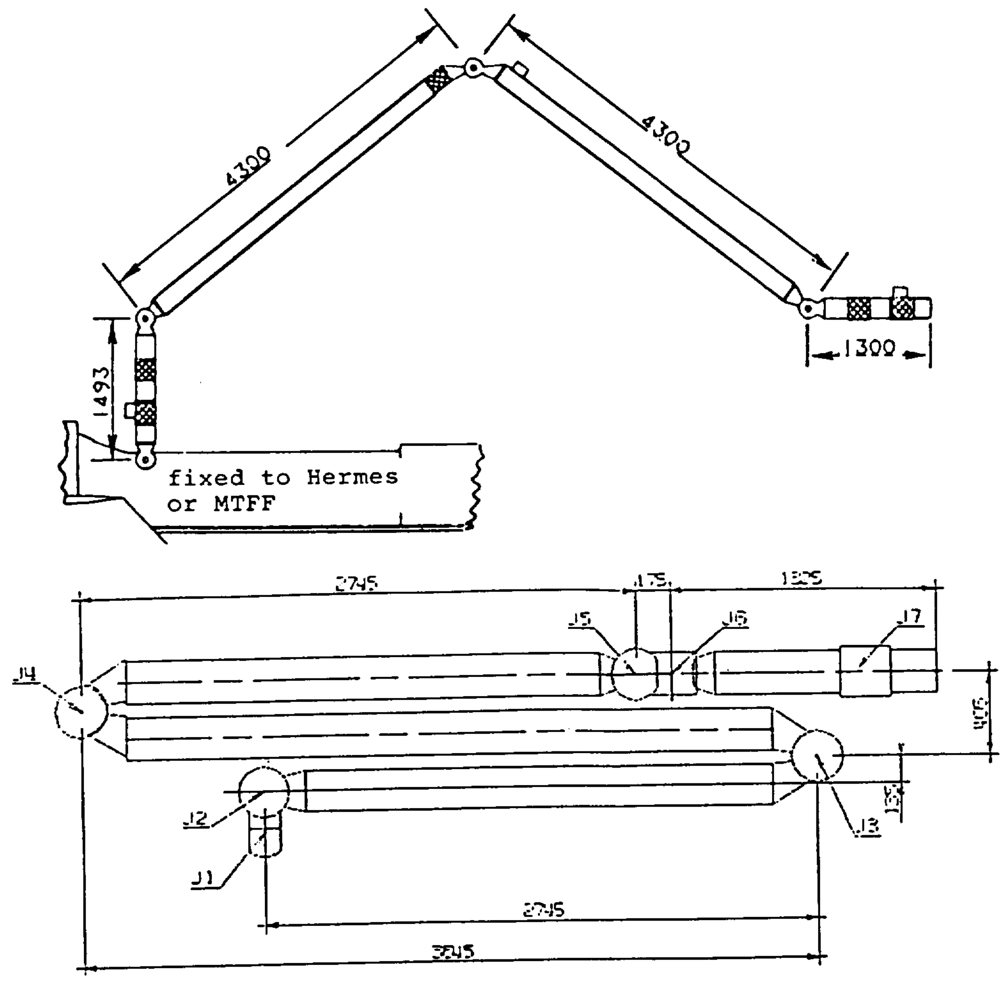


Figure 1. Functional drawing of different A&R concepts for MTFF internal servicing [6].



Joint Limits:	
J1:	+180/-180
J2:	+210/- 30
J3:	0/-270
J4:	+270/ 0
J5:	+120/-120
J6:	+ 90/- 90
J7:	+180/-180

(values in deg)

Figure 2. Large manipulator design for MTFF external servicing. Upper one: relocatable concept [3]; lower one: MBB/ERNO design [4].

The high costs of actual orbital operations imply a high degree of realism with regard to the simulation facilities on the ground to assure mission success even at a multitude of potential adverse events. Viewing this from a negative point of view, any programming or operator mistake would result in a tremendous amount of expenses in the case of an actual space mission as compared with costs incurring in an adequate simulation facility. Therefore all activities referring to teleoperated robotic routine operations must be planned and observed on ground. As a consequence, very early it proved to be extremely necessary to set up an adequate ground-based hardware simulator, the Servicing Test Facility, for in-orbit operations technology verification.

The STF is intended to be a *Test and Training Simulator for Ground-Based Teleoperated In-Orbit Robotic Servicing* during un-manned phases. With this intention, the STF is projected being a facility for the support of the MTFF *routine operations* during the Post-IOC-Phase. Because of the peculiar and complex pretensions, specifically to real-time flight operational support, the facility has to be in direct vicinity of the flight control center. The mean lifetime of the MTFF is projected for at least 30 years, and since the MSCC is in duty for all operational tasks, adequate provisions for a long-term operational concept have to be considered already in the current planning for later use in the Post-IOC-Phase.

3. High Fidelity Simulation Facility

Based on the future technological challenges described above it is essential to provide as far as possible a true-scale hardware simulator for the verification of the robotic servicing procedures under realistic conditions. The facility has to allow the implementation of important hardware components, possibly even flight specific hardware as e.g. end effectors or sensors, in a real-time and real-size simulation to increase the confidence in the ground operating system. Pure computer simulations will not be sufficient for the qualification of the ground operating system for robotic servicing tasks to guarantee mission success.

The STF therefore will provide the capability to perform the following spectrum of four main tasks [14-16]:

1. System simulation for the development and verification of mission procedures;
2. Teleoperator training with specific regards to signal delay times;
3. Verification of ground operations segment for in-orbit robotic tasks;
4. Simultaneous parallel simulation of on-going in-orbit routine operations during the Post-IOC-Phase.

While the first three tasks will support mainly all ground-based robotic activities in the ground operations preparatory phase, the parallel simulation will be a necessary task to be performed during the actual mission specifically for reasons of trouble shooting. Moreover, due to the signal delay times of several seconds (even up to about 10 sec during the Spacelab D1-mission) between ground and on-board system, actual status knowledge of the flight systems on ground is essential. In this case all telecommands may be transmitted to both systems in parallel, thus the ground-based operator will be able to observe the effects of manipulation in the ground facility in advance without time delay. Delayed on-board system information then will be played back to the STF via telemetry links and will be displayed additionally to the ground-based manipulator. Hence, it is possible to observe deviations between both systems and increase safety of the ground operating system.

Regarding specifically the inherent difficulties in the signal delay times the training philosophy and corresponding concepts for ground-based teleoperators must be based on high-fidelity equipment. Both the ground manipulator and animations of its computer simulated counterpart have to be provided to the operator by adequate monitoring devices having capabilities for

3D-stereoscopic imaging. According to the training philosophy three different steps shall be envisaged, each step gradually increasing in complexity:

1. *Training with ground manipulator (with time delays):*
   a. by direct view into the facility lab,
   b. by indirect view via video monitoring.
2. *Training by computer simulated animation using computer graphics system (with time delays):*
   a. for a dynamic model of the ground manipulator,
   b. for a dynamic model of the flight manipulator.
3. *Hybrid training:*
   a. by video monitoring of both the ground manipulator (without time delay) and the equivalent computer generated animation of the ground model (regarding time delay),
   b. by video monitoring of both the ground manipulator (without time delay) and the equivalent computer generated animation of the flight model (regarding time delay).

   This last training step is regarded to be of highest complexity. Monitoring of both images on two different screens would be a first approach, but the final aim should be an overlay of both images on one single screen.

For the purposes of teleoperator training and parallel operation to the onboard activities, it is essential to have the representative and detailed behaviour of the real manipulator and environment (internal and external) available on ground, as well. To assure conflict-free operations, the real manipulator geometry and kinematics must be available, including the geometry of the MTFF interieur and exterieur. Only for such a configuration of true-scale models it will be guaranteed that the teleoperator performs the manipulator activities within the bounds of the MTFF working area successfully.

The studies performed on the feasibility and the needs of the STF [14-16] have identified the following basic requirements:

- Representative behaviour of the manipulators and end effectors.
- Representative behaviour of the MTFF subsystem/payload mechanisms and functions as far as being relevant for automation and robotics.
- Representative video picture processing.
- Representative teleoperator station.

Regarding these functional and operational requirements the main STF components were identified giving:

- Replica of external and internal manipulator as far as possible in true scale with real onboard geometry and kinematics including control electronics. For the 1g-environment, the large length of about 11 m for the external manipulator requires a sufficiently stiff laboratory system in order to perform manipulations in all three dimensions. The real onboard dynamics will therefore be simulated by software.
- Replica or, if required, the real flight hardware of all flight end effectors to be connected to the manipulator.
- Software simulation of the onboard manipulator kinematics and dynamics, and of the onboard end effector kinematics and functions.
- Standardized software simulation system to allow for easy adaptation of different manipulator and end effector kinematic simulations.
- Teleoperator work station to allow for complete remote control of manipulator / end effector from ground, including 3D-display, status display, joystick / sensor ball control and signal delay simulation.
- Mockup of MTFF exterieur and interieur, as far as automation and robotics are concerned, in true scale as well as single standard ORU mockups and single racks.

- Lighting system, especially for sun simulation, shadowing effects.
- Telemetry / telecommand- and video-connection to the onboard manipulator system in real time to allow for remote control and simultaneous simulation (link via MSCC).
- Computing facilities for software simulation in real time, manipulator/equipment control, remote control station support, data recording and procedure development.
- Real-time 3D-stereoscopic graphic simulation system connected to the computer facilities for training purposes and rapid prototyping of In-Orbit Servicing procedures.
- If required in case of flight hardware implementation, clean room conditions are foreseen for operating the STF.

Moreover, for realistic teleoperator training of very detailed and sophisticated manipulations, where basically the end effector is used at the location of the object to be manipulated, the overall motion of the manipulator is not of main interest. In all these cases of servicing training tasks in the proximity of the object, it is very necessary to incorporate the manipulator dynamics as well as the forces and torques applied by the mechanisms of the MTFF specific objects. Here, the special simulation capabilities of EPOS will be favourably used. A close connection of both facilities, STF and EPOS, together with the MSCC is therefore required in order to guarantee for realistic simulations of In-Orbit Servicing teleoperations.

Figure 3 presents a functional overview of the Servicing Test Facility (a computer generated scene of the laboratory: the MTFF Mockup, the large 1g-lab manipulator, an internal manipulator, and the teleoperator workstation). Figure 3 also gives an overview of the complete ground-to-orbit scenario with interfaces to the other relevant ground-based facilities and the in-orbit COLUMBUS element MTFF.

4. Definition of the STF Basic Components

Presently, the Phase B Study for definition of the different basic hardware and software components has been finished. This refers to the electro-mechanical system of the laboratory manipulators for both the MTFF-internal and the -external robotics, and to the software and computer concept for operating the facility.

4.1 The Electro-Mechanical System

The manipulators and end effectors to be used inside and outside of the MTFF are still in the definition phase, and the final flight version may change according to the current specification. This important fact requires a flexible simulator design which can be easily adapted to different design modifications which especially applies for changes in manipulator geometry or in the kinematic behaviour. Therefore, the concept of a modular build-up of the STF is foreseen that allows gradual adaptation to the respective state of actual hardware equipment. This modular concept is used for both the hardware and the software simulation part of the STF. In case of the large external manipulator, the approach in Figure 2, lower one, was identified being the more complex one to be realized in the 1g-environment of the laboratory. Hence, once having qualified the more complex manipulator for operational readiness, less complex versions are regarded to apply as well.

Both the smaller internal robot and the large external manipulator must be operated in the lab in all three spatial dimensions and hence are strongly affected by gravity. Since no greater difficulties are expected to arise from the technical realization of a duplication of the internal robots, the most effort in the facility design therefore will originate from the hardware copy of the large manipulator. This replica of the flight version has been designed such as to physically simulate the flight-equivalent geometric and kinematic behaviour in all 3 dimensions. Obviously the dynamic behaviour will be much different since an almost very stiff construction is required. The major loads on the manipulator are given by the relatively high weights of the joint actuators rather than by the influence of the limb structure which will be made of light-weight material, CFRP. Moreover, the limb structure is designed in order to sustain elastic deflections within a dedicated margin (less than 10 cm end effector displacement for a worst case assumption of a horizontally cantilevered manipulator arm), and the necessary compensation of positioning inaccuracies due to the deformations will be performed by respective joint actuator commands. (Of course, closed loop control by telemanipulations via video sensor feedback will increase positioning accuracy without doubt.) Figure 3 gives an impression of the MTFF mockup and external lab manipulator arrangement: in case of linear or rotating manipulator travelling concepts, the MTFF mockup will be moved correspondingly with respect to the laboratory fixed manipulator base; appropriate provisions like rail guides are foreseen.

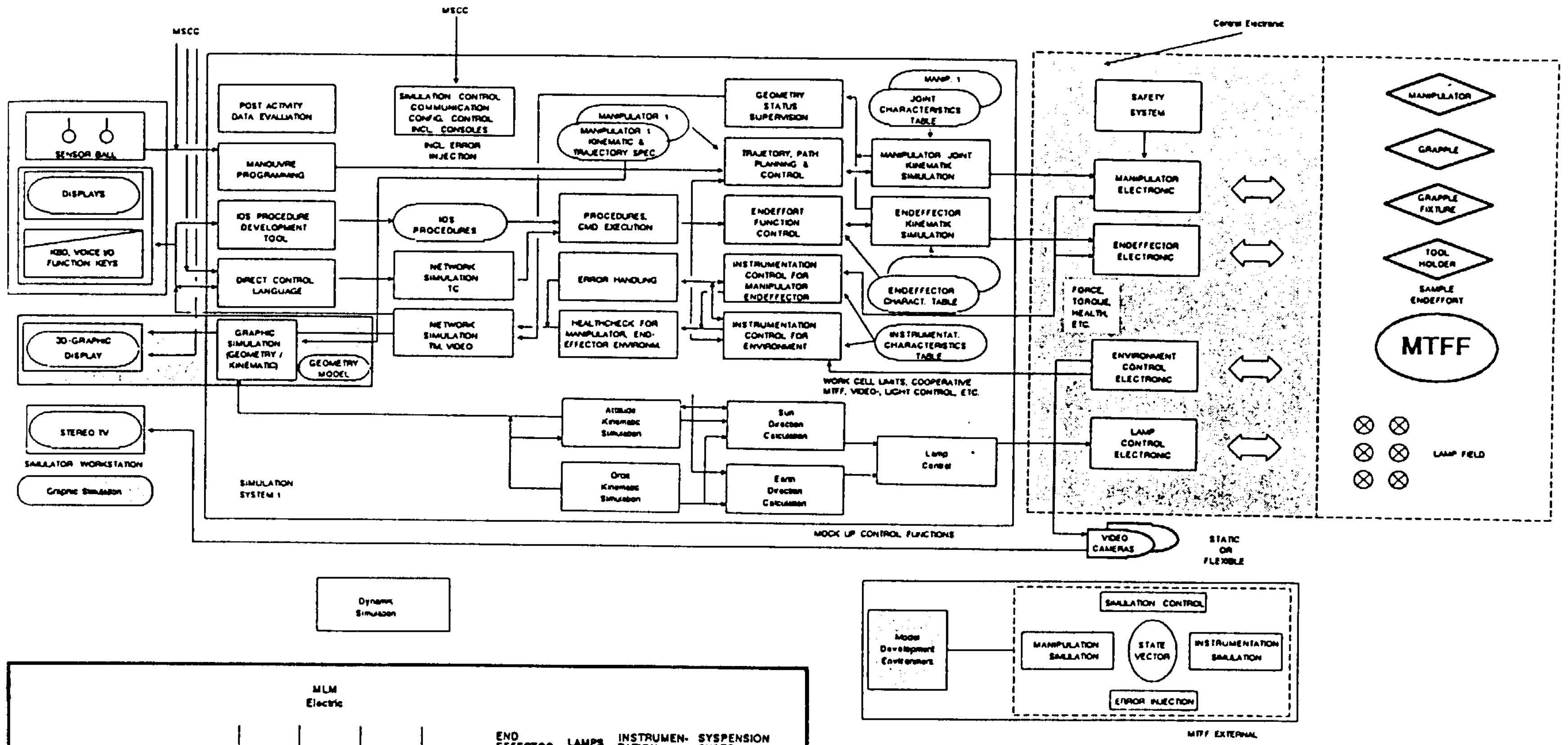
The torques exerted at the joints will be tremendously high as compared with those in the 0g-environment. Especially the bending moments at the shoulder joint actuator at the manipulator base are excessively high due to the exponential accumulation of the torques arising from the other actuator weights along the manipulator arm structure. For this reason proper actuator design was accomplished and moreover, the selection of specific actuator types influenced again the limb structural behaviour: A careful trade-off between both the joint actuator and the structural design was necessary. The final selection was to use HERA/SMS similar design with integrated electrical drive for the end effector joints and electrical drives with cyclo gearboxes or harmonic drives for the other joints. Optionally, the use of hydraulic torquers for the stronger actuators will be presently analyzed.

4.2 The Software and Computer Concept

The STF intelligent system is structured to fulfil the requirements of the different applications. A global design structure has been derived to allow the reuse of the same facility with only simple reconfigurations. The software simulation system is designed such that basically all configuration and construction dependent parameters of the manipulators, the end effectors and the MTFF can be stored in software tables which easily can be replaced. The control and table interpretation software ist therefore unchanged in case of manioulator changes. For MTFF external servicing the STF design concept is presented in Figure 4; the equivalent concept applies for internal servicing.

As far as possible, off-the-shelf computer systems are used with commercial software systems and networks. Moreover, use shall be made of the software capabilities of European systems supporting flight segment development like EUROSIM (*European Robotic Operations Simulator*) at ESTEC or CSF (*Columbus Simulation Facility*). All fast data processing in direct communication with the electronic or electric systems is performed around an IEEE 488 data bus and on dedicated micro-processors. The coordination of the different joint control processors (a decentralized joint control strategy for the large manipulator ist favoured), together with the associated transformation, will be performed on a dedicated minicomputer.

On the other hand, system supervision and activity coordination will be performed by one single authority, which is the STF simulation system, Figure 5. The corresponding simulation software acts as the central control system for the complete system set-up. All simulations, device coordinations, system supervision and programming tools operate here. For the graphic simulation a special hardware system is necessary that allows 3D real-time animation of robot manipulations, preferably for surface shaded volume models, by manual control via sensor ball or joysticks. The corresponding dynamic simulation requires a very powerful computing system together with a dedicated software package possibly tailored to the flight design (e.g. HERA simulation capabilities within EUROSIM). Man-machine interface is standardized by using off-the-shelf work station tools. State-of-the-art communication is by window mechanisms and pop-up menues/icons commanded by mouse.



MSCC
MSCC
Central Electronic
SENSOR BALL
DISPLAYS
KBD, VOICE I/O FUNCTION KEYS
3D-GRAPHIC DISPLAY
STEREO TV
SIMULATOR WORKSTATION
Graphic Simulation
POST ACTIVITY DATA EVALUATION
MANOUVRE PROGRAMMING
IOS PROCEDURE DEVELOPMENT TOOL
DIRECT CONTROL LANGUAGE
GRAPHIC SIMULATION (GEOMETRY / KINEMATIC)
GEOMETRY MODEL
SIMULATION CONTROL COMMUNICATION CONFIG. CONTROL INCL. CONSOLES
INCL. ERROR INJECTION
IOS PROCEDURES
NETWORK SIMULATION TC
NETWORK SIMULATION TM, VIDEO
MANIPULATOR 1
MANIPULATOR 1 KINEMATIC & TRAJECTORY SPEC.
PROCEDURES, CMD EXECUTION
ERROR HANDLING
HEALTHCHECK FOR MANIPULATOR, END-EFFECTOR ENVIRONM.
GEOMETRY STATUS SUPERVISION
TRAJECTORY, PATH PLANNING & CONTROL
ENDEFFORT FUNCTION CONTROL
INSTRUMENTATION CONTROL FOR MANIPULATOR ENDEFFECTOR
INSTRUMENTATION CONTROL FOR ENVIRONMENT
MANIP. 1
JOINT CHARACTERISTICS TABLE
MANIPULATOR JOINT KINEMATIC SIMULATION
ENDEFFECTOR KINEMATIK SIMULATION
ENDEFFECTOR CHARACT. TABLE
INSTRUMENTAT. CHARACTERISTICS TABLE
WORK CELL LIMITS, COOPERATIVE MTFF, VIDEO-, LIGHT CONTROL, ETC.
Attitude Kinematic Simulation
Orbit Kinematic Simulation
Sun Direction Calculation
Earth Direction Calculation
Lamp Control
SIMULATION SYSTEM 1
MOCK UP CONTROL FUNCTIONS
SAFETY SYSTEM
MANIPULATOR ELECTRONIC
ENDEFFECTOR ELECTRONIC
FORCE, TORQUE, HEALTH, ETC.
ENVIRONMENT CONTROL ELECTRONIC
LAMP CONTROL ELECTRONIC
MANIPULATOR
GRAPPLE
GRAPPLE FIXTURE
TOOL HOLDER
SAMPLE ENDEFFORT
MTFF
LAMP FIELD
VIDEO CAMERAS
STATIC OR FLEXIBLE
Dynamic Simulation
Model Development Environment
SIMULATION CONTROL
MANIPULATION SIMULATION
STATE VECTOR
INSTRUMENTATION SIMULATION
ERROR INJECTION
MTFF EXTERNAL

Figure 4. Software and computer concept.

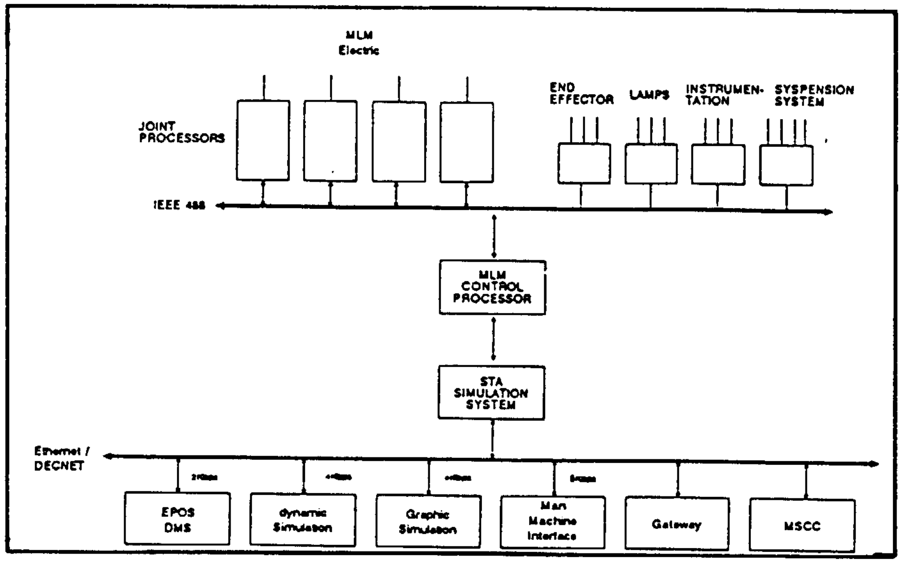


Figure 5. STF simulation system environment.

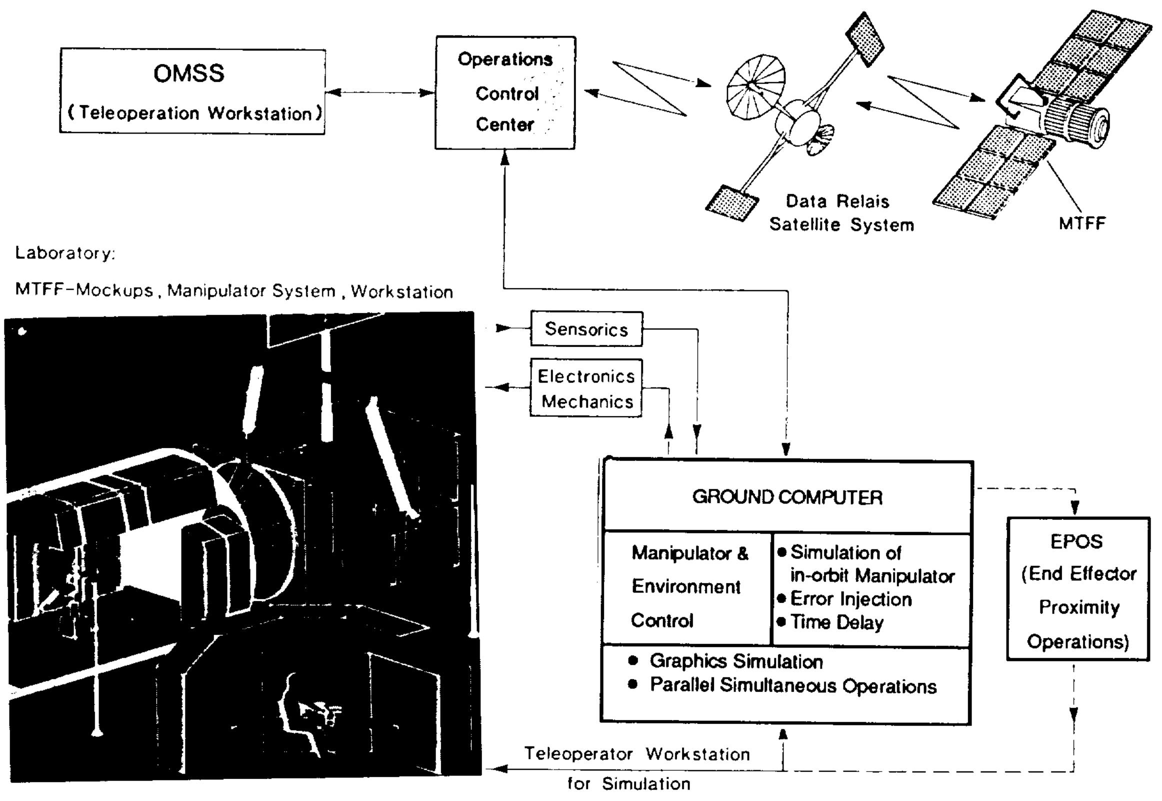

OMSS
(Teleoperation Workstation)
Operations Control Center
Data Relais Satellite System
MTFF
Laboratory:
MTFF-Mockups, Manipulator System, Workstation
Sensorics
Electronics Mechanics
GROUND COMPUTER
Manipulator & Environment Control
• Simulation of in-orbit Manipulator
• Error Injection
• Time Delay
• Graphics Simulation
• Parallel Simultaneous Operations
EPOS (End Effector Proximity Operations)
Teleoperator Workstation for Simulation

Figure 3. Functional overview of the Servicing Test Facility. Complete ground-to-orbit scenario.

5. Concluding Remarks

Rather than for technology development (cf. HERA 1g-Demonstrator facility, or MARS = MTFF A&R System Testbed) the Servicing Test Facility will be used dominantly for the support of the ground operating system within the MSCC for all ground-based teleoperated robotic routine operations. According to this objective, hardware and software components being typical for robotic servicing needs (e.g. teleoperator control station tailored to ground-based remote operations) shall largely be provided and incorporated within the facility by the specific developers. These can be ESA, industrial companies or non-profit institutions like universities or even DFVLR. The basic facility equipment is provided by DFVLR. The on-going activities for the facility set-up are presently faced with the detailed design of the basic components and studies on alternatives to real-sized hardware simulations such as scaled-down versions or cable-suspended manipulator designs.

Acknowledgements

All members of the *System Technologies Group* within the *Spaceflight Dynamics Section* of GSOC deserve recognition for their valuable contributions. Major part during the definition phase of the electro-mechanical system and the software and computer concept for the set-up of the Servicing Test Facility has been carried out by the German space companies MBB/ERNO and Dornier System under DFVLR contract No. 5-575-4359.

References

[1] Dornier System:
Equipment Manipulation and Transportation System (EMATS); International Automation and Robotics Workshop, ESTEC, 21-22 June 1988, Noordwijk, The Netherlands.

[2] Dornier System:
Automatic Sample Handling by Means of an Experiment Manipulation System (AHS/EMS); International Automation and Robotics Workshop, ESTEC, 21-22 June 1988, Noordwijk, The Netherlands.

[3] Fokker Space and Systems:
HERA Design Documentation; Report No. FSS-R-88-080, 14 July 1988, Amsterdam, The Netherlands.

[4] MBB/ERNO:
The HERA Manipulator Arm Limb Subsystem; Technical Proposal, Vol. 1, 12 Aug 1988, Bremen, Fed. Rep. of Germany.

[5] Matra Espace:
Bi-Arm Servicer (BIAS), Study Status and Potential Use; International Automation and Robotics Workshop, ESTEC, 21-22 June 1988, Noordwijk, The Netherlands.

[6] MBB/ERNO:
Study on Robotics, Spacecraft Servicing and Assembly (ROSSA); International Automation and Robotics Workshop, ESTEC, 21-22 June 1988, Noordwijk, The Netherlands.

[7] ESTEC:
Service Manipulator System, System Description Handbook; ESA-Document RTS/SMS/1/ESA/HBK, 7 March 1985, Noordwijk, The Netherlands.

[8] E.-L. Klingelhöfer, J. Puls:
Potentials of Robotic Operations on Board the Man-Tended Free-Flyer; 38th Congress of the International Astronautical Federation, Paper No. IAF-87-17, 10-17 Oct 1987, Brighton, United Kingdom.

[9] G. Hirzinger:
ROTEX, Germany's First Step into Space Robotics; AIAA/NASA First International Symposium on Space Automation and Robotics, Paper No. AIAA-88-5008, 29-30 Nov 1988, Arlington, VA, USA.

[10] R.H. Bentall, D. Kassing:
Man Tended Options for European Space Robotics; First European In-Orbit Operations Technology Symposium, 7-9 Sept 1987, Darmstadt, Fed. Rep. of Germany.

[11] ESA/ESTEC:
ESA Technology Research Programme for 1989 and Medium Term Plan for 1990-91; TD(89)1, 27 Nov 1988, Noordwijk, The Netherlands.

[12] J. Kehr, K. Reinel:
The Manned Space Laboratories Control Center (MSCC) at DFVLR, Oberpfaffenhofen; 39th Congress of the International Astronautical Federation, Paper No. IAF-88-087, 8-15 Oct 1988, Bangalore, India.

[13] K. Reinel, G. Heimbold, T. Lange, B. Schäfer:
Aufgaben der Simulationsanlagen für den Raumflugbetrieb im Betriebszentrum für bemannte Weltraumlabors; DFVLR Nachrichten, Vol. 54, June 1988.

[14] B. Brand, J. Puls, B. Schäfer:
Servicing Test Facility - Phase A Study; DFVLR IB 575-86/1, 19 Dec 1986, Oberpfaffenhofen, Fed. Rep. of Germany.

[15] MBB/ERNO:
Aufbau von Simulationsanlagen für den Raumflugbetrieb; Final Report, DFVLR Contract No. 5-575-4331, 15 Dec 1987, Oberpfaffenhofen, Fed. Rep. of Germany.

[16] MBB/ERNO, Dornier System:
Planung, Auslegung und Aufbau der Simulationsanlagen für das Betriebszentrum - Servicing Test Facility, Phase B Study; DFVLR Contract No. 5-575-4359, 8 Dec 1988, Oberpfaffenhofen, Fed. Rep. of Germany.

N90-29344

CONCEPT SYNTHESIS OF AN EQUIPMENT MANIPULATION AND TRANSPORTATION SYSTEM EMATS

W. De Peuter, ESTEC, The Netherlands
E. Waffenschmidt, Dornier Federal Rep. Germany

Abstract

The European Columbus Scenario is established. One of the Columbus Elements, the Man Tended Free Flyer will be designed for fully autonomous operation in order to provide the environment for micro gravity facilities. We discuss the Concept of an autonomous automation system which perform servicing of facilities and deals with related logistic tasks.

1. Introduction

The importance of Automation and Robotics (A&R) has grown rapidly in recent years due to challenging demands for autonomous serivicing in space.

Many of the techniques and experience gained from industrial development will be used in space application, as indicated by various robotics activities at the US., Europe and Japan.

The extensive use of robots in future space production, research and exploration and their importance for servicing and maintenance of autonomously operating facilities is obvious.

Running such space facilities with minimal human involvement is a unique challenge and opportunity to apply intelligent robotic techniques in experiment and processing systems.

At present, the use of robotics in the European space scenanrio concentrates on the Columbus Man-Tended Free Flyer (MTFF). The MTFF is a free flying "quiet laboratory" in orbit which provides the environment for microgravity experiments with only very low disturbances (10^{-6} g). The MTFF is planned to be unmanned for a time period of 6 months and man-tended during the servicing events (when it is attached to the ISS or docked to HERMES).

During the absence of men, the MTFF must be operated autonomously by an automation system installed inside the Module, which performs all required manipulation and transportation tasks. This paper deals with a first concept synthesis for this Equipment Manipulation and Transportation System (EMATS) for the internal servicing of the MTFF Laboratory.

2. MTFF Servicing Scenario and Model Mission

The first stages in European manned space flight where extensive A&R systems are needed will be (see Figure 2-1)

- MTFF in nominal unmanned period
- MTFF/HERMES during manned Servicing

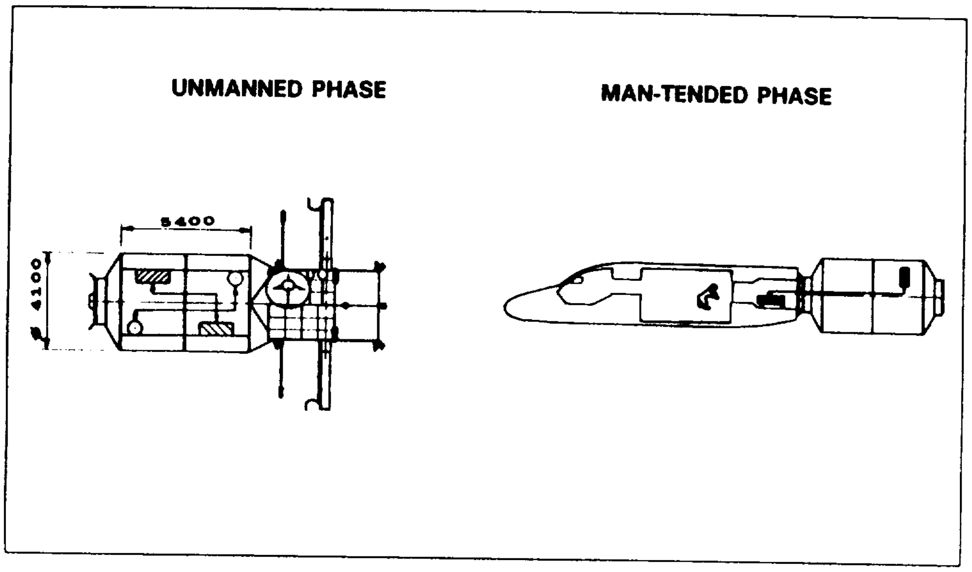
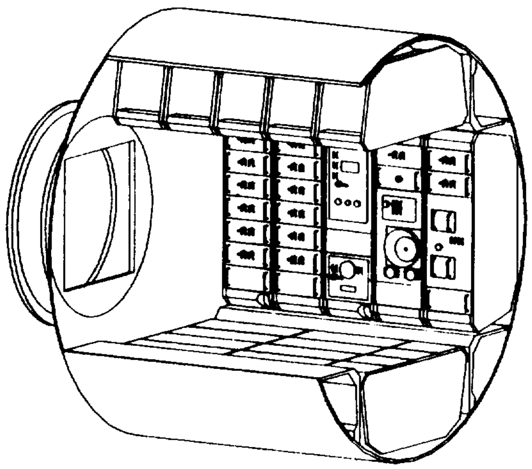


Figure 2-1: EMATS Application Scenarii

They represent the basic MTFF scenarii and hence they are the most relevant scenarii for the applications of EMATS.

It is assumed that the reference payload for the first mission of the MTFF will be a mixture of Materials Science facilities and Life Science facilities called M/C 400. The principle accommodation of these experiment facilities inside the Pressurized Module of the MTFF is shown in Figure 2-2.



• MATERIAL SCIENCE	
• Gradient Furnace	(GFQ)
• Containerless Processing	(CLF)
• Thermophysical Properties	(TPP)
• Vapour Growth	(VGF)
• Solution Growth	(SGF)
• Liquid Phase Epitaxy	(LPE)
• Flux Growth	(FGF)
• Traveling Solvent	(TSF)
• Critical Point	(CPF)
• Transport Properties	(TPF)
• LIFE SCIENCE	
• Aquarack	(AQR)
• Biochamber	(GBL 1)
• Plant Facility	(GBL 2)
• CELSS	
• Cell Fusion	(BPF 1)
• Electrophoresis	(CFF)
• Phase Partitioning	(BPF 2)
• Downstream Process	(BPF 3)
• Cell Cultivation	(BPF 5)

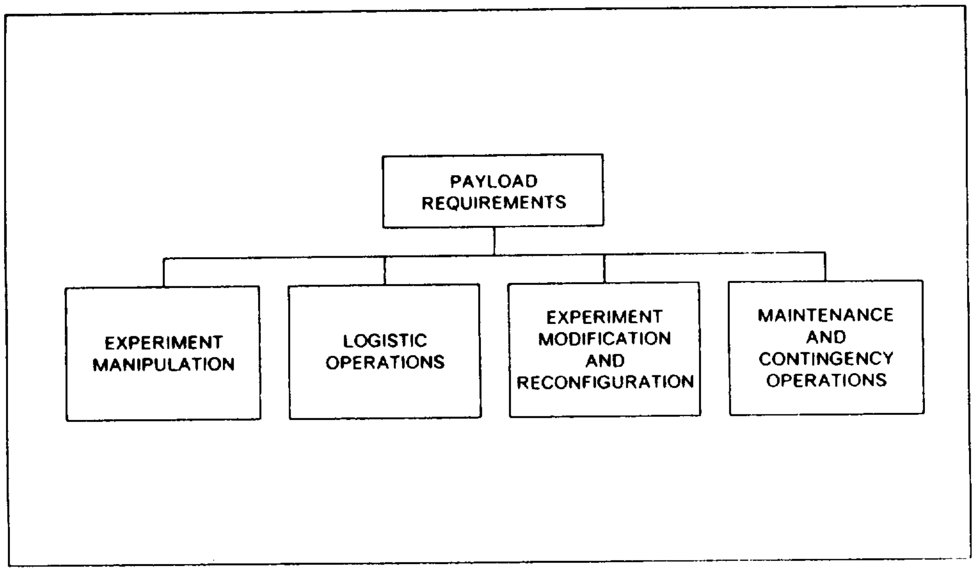
Figure 2-2: Accommodation of MIC 400 Payload

3. EMATS Tasks and Functional Requirements

Based on the analysis of the application of A&R for the MTFF Model payload and the MTFF servicing scenarii the tasks for robotics can be identified by answering the both questions:

- What shall be done?
- How and where shall it be done?

Analysing "what" the manipulators shall do, leads to the classification of the tasks in the following four groups:

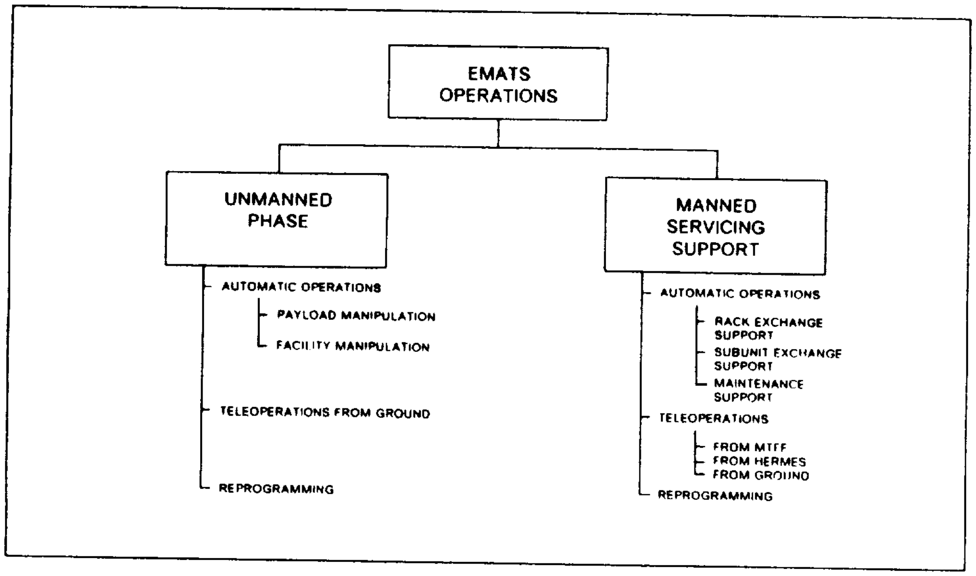

PAYLOAD REQUIREMENTS
EXPERIMENT MANIPULATION
LOGISTIC OPERATIONS
EXPERIMENT MODIFICATION AND RECONFIGURATION
MAINTENANCE AND CONTINGENCY OPERATIONS

Based on the major Payload Requirements the Generic Functions of the Equipment Manipulation and Transportation System like

- MOVE MANIPULATOR TO PAYLOAD POSITION
- REMOVE PAYLOAD (e.g. Sample)
- INSTALL PAYLOAD
- TRANSPORT PAYLOAD
- PAYLOAD INSPECTION
- OPEN DOOR
- CLOSE DOOR
- FACILITY INSPECTION WITH EE CAMERA
- FACILITY CLEANING WITH SPECIAL TOOL
- TELEMANIPULATION
  - SINGLE JOINT CONTROL
  - CARTESIAN CONTROL
  - END EFFECTOR CONTROL
  - CAMERA CONTROL
- CONTINGENCY HOLD

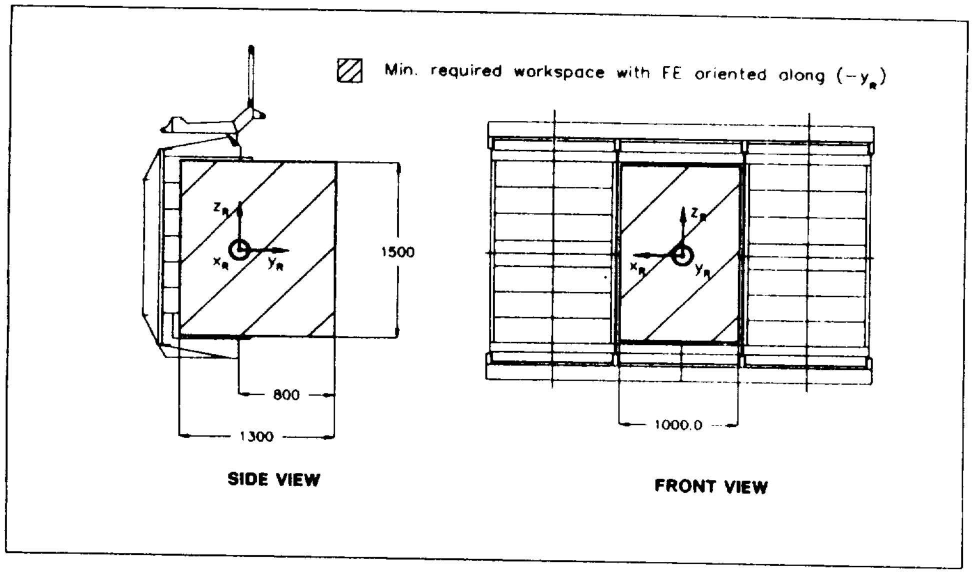
were generated.

These "Generic Functions" leads to the EMATS Operations namely



Analysing "how" and "where" the tasks shall be done leads to the identification of robotic requirements

- workspace needed
- orientation performance



4. EMATS Concepts and Trades

The Results of the Analysis of EMATS Tasks and functional Requirements form the basis of the Concept development.

In order to illustrate the systematic and evolutionary synthesis of an EMATS concept, the following classification of A&R Systems was applied.

D:	Dedicated Mechanism
F:	(Permanently) Fixed Manipulators
R:	Rail-based Manipulators
T:	Manipulators with Transplantable Base
C:	Climbing Manipulators
E:	Exotic Concepts (e.g. free flying robots...)

The evolution starts from class "D" which can be seen as the ultimate of a "convential" non-robotic approach. The next classes add more and more sophistication, intelligence and flexibility while in general reduces the "volume" of apparatus or devices needed.

The upper end is represented by fictitious "exotic" concepts with ultimate flexibility, but for the time being also imense development risk. They are supposed to indicate a "ceiling" for technology and show that the class "R" and "C" concepts are indeed the current peak of the evolution.

Figure 4-1 gives an overview of the different concepts.

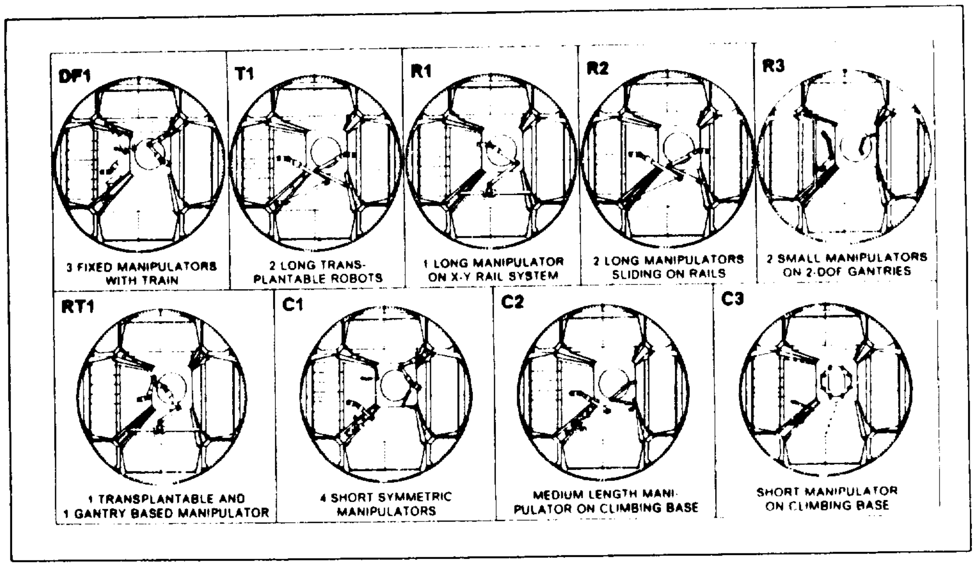

DF1
3 FIXED MANIPULATORS WITH TRAIN
T1
2 LONG TRANS-PLANTABLE ROBOTS
R1
1 LONG MANIPULATOR ON X-Y RAIL SYSTEM
R2
2 LONG MANIPULATORS SLIDING ON RAILS
R3
2 SMALL MANIPULATORS ON 2-DOF GANTRIES
RT1
1 TRANSPLANTABLE AND 1 GANTRY BASED MANIPULATOR
C1
4 SHORT SYMMETRIC MANIPULATORS
C2
MEDIUM LENGTH MANIPULATOR ON CLIMBING BASE
C3
SHORT MANIPULATOR ON CLIMBING BASE

Figure 4-1: EMATS Manipulator Concepts

A trade off, based on some typical MTFF relevant criteria like:

	CONCEPTS										
CRITERIA	D	DF1	T1	R1	R2	R3	RT1	C1	C2	C3	E
MINIMUM PM IMPACT	- -	0	0	0	--	+	0	+	+	+ +	+ +
FULFILLMENT OF USER REQUIREMENTS	+	-	0	+	+	+ +	+	+	+ +	+	+ +
RELIABILITY	+	0	0	-	0	0	+	-	-	-	-
FLEXIBILITY	- -	- -	0	0	0	+	+	+	+	+	+ +
FEW IN-ORBIT OPERATIONS	-	-	-	+	0	+	-	-	0	+	0
DEVELOPMENT COST/RISK	+	+	0	+ +	+	+	0	- -	- -	- -	- -
OPERATIONAL COST	- -	-	0	0	0	+	0	0	+	+ +	0
APPLICATION/DESIGN GROWTH	- -	- -	+	-	0	-	+	+ +	+ +	+ +	+ +

results in the selection of Concept R3 and C3 for final comparison. Figures 4-2 and 4-3 show the preselected concepts

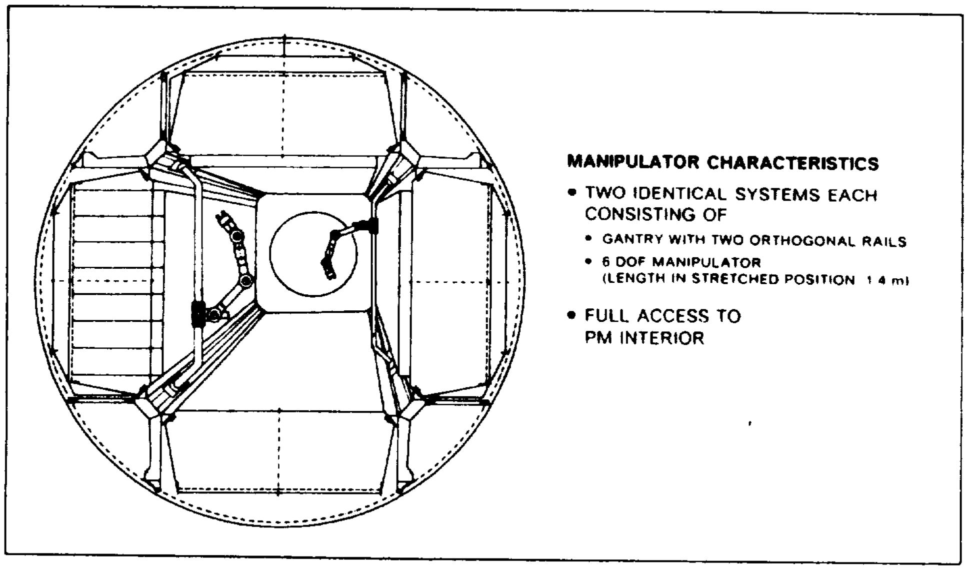
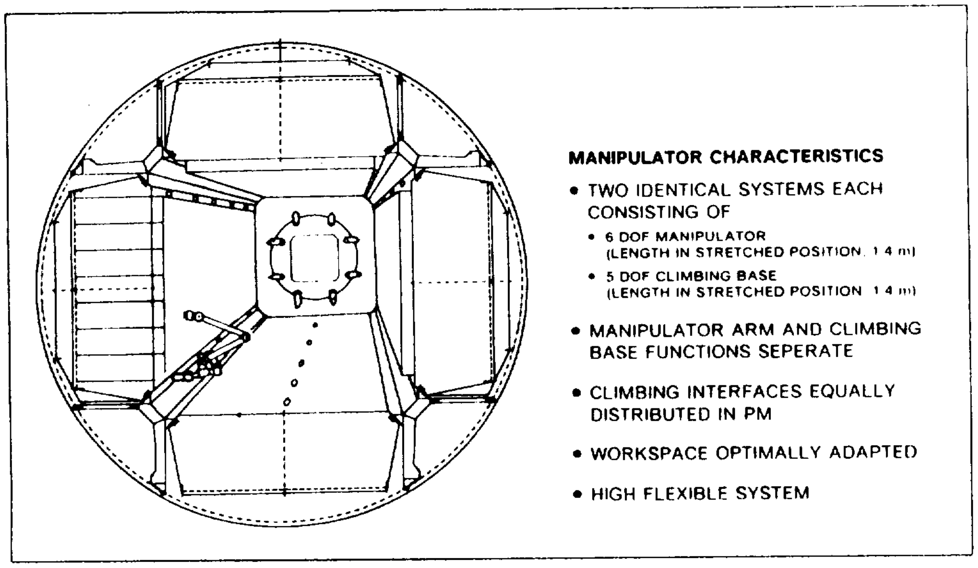

MANIPULATOR CHARACTERISTICS
• TWO IDENTICAL SYSTEMS EACH CONSISTING OF
• GANTRY WITH TWO ORTHOGONAL RAILS
• 6 DOF MANIPULATOR (LENGTH IN STRETCHED POSITION 1.4 m)
• FULL ACCESS TO PM INTERIOR

Figure 4-2: Gantry Based Concept R3

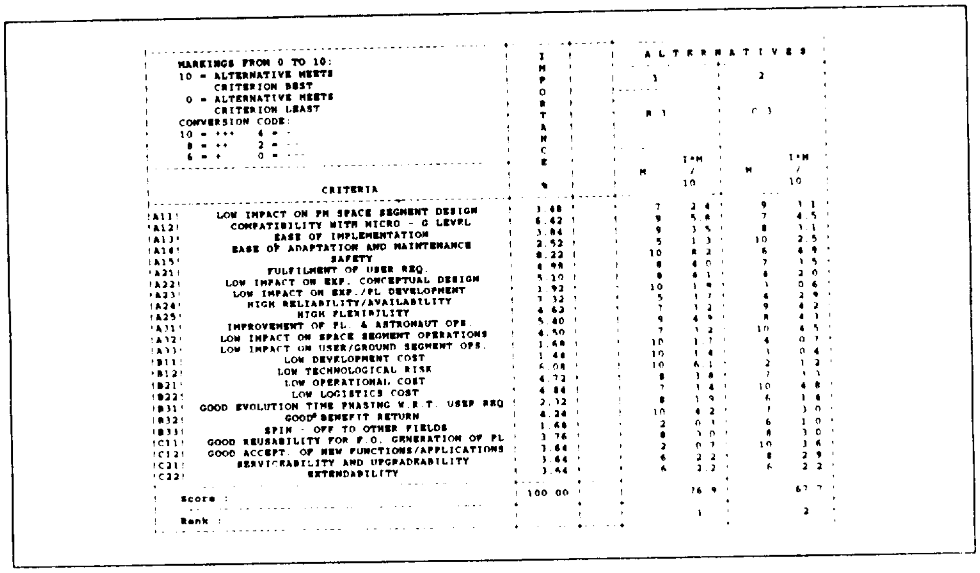


MANIPULATOR CHARACTERISTICS

- TWO IDENTICAL SYSTEMS EACH CONSISTING OF
  - 6 DOF MANIPULATOR (LENGTH IN STRETCHED POSITION 1.4 m)
  - 5 DOF CLIMBING BASE (LENGTH IN STRETCHED POSITION 1.4 m)
- MANIPULATOR ARM AND CLIMBING BASE FUNCTIONS SEPERATE
- CLIMBING INTERFACES EQUALLY DISTRIBUTED IN PM
- WORKSPACE OPTIMALLY ADAPTED
- HIGH FLEXIBLE SYSTEM

Figure 4-3: Climbing Concept C3

The criteria and weighing factor for the final trade are given together with the evaluation in Figure 4-4.



MARKINGS FROM 0 TO 10:
10 = ALTERNATIVE MEETS CRITERION BEST
0 = ALTERNATIVE MEETS CRITERION LEAST
CONVERSION CODE:
10 = +++ 4 = -
8 = ++ 2 = --
6 = + 0 = ---

	CRITERIA	IMPORTANCE %	ALTERNATIVES 1 R1 M	ALTERNATIVES 1 R1 I*M/10	ALTERNATIVES 2 C3 M	ALTERNATIVES 2 C3 I*M/10
A11	LOW IMPACT ON PM SPACE SEGMENT DESIGN	3.48	7	2.4	9	3.1
A12	COMPATIBILITY WITH MICRO - G LEVEL	6.42	9	5.8	7	4.5
A13	EASE OF IMPLEMENTATION	3.84	9	3.5	8	3.1
A14	EASE OF ADAPTATION AND MAINTENANCE	2.52	5	1.3	10	2.5
A15	SAFETY	8.22	10	8.2	6	4.9
A21	FULFILMENT OF USER REQ.	4.98	8	4.0	7	3.5
A22	LOW IMPACT ON EXP. CONCEPTUAL DESIGN	5.10	8	4.1	4	2.0
A23	LOW IMPACT ON EXP./PL DEVELOPMENT	1.92	10	1.9	3	0.6
A24	HIGH RELIABILITY/AVAILABILITY	7.32	5	3.7	4	2.9
A25	HIGH FLEXIBILITY	4.63	7	3.2	9	4.2
A31	IMPROVEMENT OF PL. & ASTRONAUT OPS.	5.40	9	4.9	8	4.3
A32	LOW IMPACT ON SPACE SEGMENT OPERATIONS	4.50	7	3.2	10	4.5
A33	LOW IMPACT ON USER/GROUND SEGMENT OPS.	1.68	10	1.7	4	0.7
B11	LOW DEVELOPMENT COST	1.44	10	1.4	3	0.4
B12	LOW TECHNOLOGICAL RISK	6.08	10	6.1	2	1.2
B21	LOW OPERATIONAL COST	4.72	8	3.8	7	3.3
B22	LOW LOGISTICS COST	4.84	7	3.4	10	4.8
B31	GOOD EVOLUTION TIME PHASING W.R.T. USER REQ	2.32	8	1.9	6	1.4
B32	GOOD BENEFIT RETURN	4.24	10	4.2	7	3.0
B33	SPIN - OFF TO OTHER FIELDS	1.68	2	0.3	6	1.0
C11	GOOD REUSABILITY FOR F.O. GENERATION OF PL	3.76	8	3.0	8	3.0
C12	GOOD ACCEPT. OF NEW FUNCTIONS/APPLICATIONS	3.64	2	0.7	10	3.6
C21	SERVICEABILITY AND UPGRADABILITY	3.64	6	2.2	8	2.9
C22	EXTENDABILITY	3.64	6	2.2	6	2.2
	Score :	100.00		76.9		67.7
	Rank :			1		2

Figure 4-4: Concept Trade off

5. Conclusions and Outlook on Future Work

Concept R3 comes out as prefered system. Its major advantages are:

- No safety concerns
- Low technological risk and development cost
- Very low impact on experiment/payload design and development (including good 1 g compatibility)
- Very good μg compatibility
- No serious impact on user/ground segment operations
- Very high improvement or payload and astronaut operations
- Uncritical stowage and implementation
- Completely satisfactory flexibility and manipulation/transportation capability at low complexity and low operational cost

Points of relative weakness are:

- Reliability/availability strongly determined by reliability of the rail and gantry subsystems
- Possible maintenance problems in case of rail failure
- The need for PM interfaces at the bottom standoffs for rail attachment (at the moment, no MTFF document seem to prohibit this, though)

The major disadvantage of R3 is

- Transport capability into servicing vehicles can only be performed with the help of dedicated devices inside those vehicles. This, however, seems an acceptable penalty.

On the other hand, concept C3 offers as advantages:

- Very high flexibility
- No problem with implementation or maintainability
- Good improvement of payload and astronaut operations
- Excellent acceptance of extended vehicles tasks
- No logistic problems
- Very good serviceability, upgradeability, reuseability.

These, however, are overshadowed by serious drawbacks:

- Very high technological risk and development cost, mainly due to the complex control of the redundant d.o.f. for climbing coordination
- For the same reason, doubts on reliability/availability and possibly high ground control operations impact
- Need for rack center I/Fs that may restrict experiment design (or, restricting center I/Fs, significantly reduced flexibility)
- Not completely negligible safety hazard.

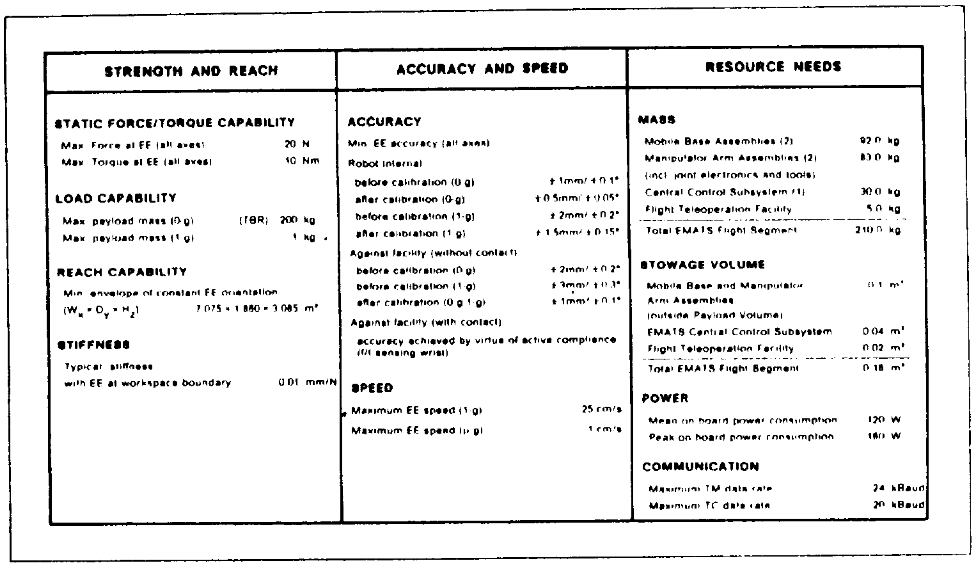
This results in a final score for R that is 13 % higher than C. This lead is very robust against perturbations in the criteria weighing. R3 dominates C3 by 17 % in the "technological" criteria and by 8 % in the "programmatic" criteria. Finally, there does not seem to be any serious and unrepairable deficit in R3, this being a very straightforward and conservative approach for which good confidence is derived.

Therefore, we recommend as the preferable EMATS concept:

R3 (Double Manipulator on longitudinal rails)

with its main characteristics:

FIRST TECHNICAL DESIGN DATA



STRENGTH AND REACH		ACCURACY AND SPEED		RESOURCE NEEDS	
STATIC FORCE/TORQUE CAPABILITY		**ACCURACY**		**MASS**	
Max Force at EE (all axes)	20 N	Min EE accuracy (all axes)		Mobile Base Assemblies (2)	92.0 kg
Max Torque at EE (all axes)	10 Nm	Robot internal		Manipulator Arm Assemblies (2)	83.0 kg
LOAD CAPABILITY		before calibration (0 g)	± 1mm/ ± 0.1°	(incl. joint electronics and tools)	
Max payload mass (0 g)	(TBR) 200 kg	after calibration (0-g)	± 0.5mm/ ± 0.05°	Central Control Subsystem (1)	30.0 kg
Max payload mass (1 g)	1 kg	before calibration (1-g)	± 2mm/ ± 0.2°	Flight Teleoperation Facility	5.0 kg
REACH CAPABILITY		after calibration (1 g)	± 1.5mm/ ± 0.15°	Total EMATS Flight Segment	210.0 kg
Min envelope of constant EE orientation		Against facility (without contact)		**STOWAGE VOLUME**	
(W_x * D_y * H_z)	7.075 × 1.880 × 3.085 m³	before calibration (0 g)	± 2mm/ ± 0.2°	Mobile Base and Manipulator Arm Assemblies (outside Payload Volume)	0.1 m³
STIFFNESS		before calibration (1-g)	± 3mm/ ± 0.3°	EMATS Central Control Subsystem	0.04 m³
Typical stiffness		after calibration (0 g 1-g)	± 1mm/ ± 0.1°	Flight Teleoperation Facility	0.02 m³
with EE at workspace boundary	0.01 mm/N	Against facility (with contact)		Total EMATS Flight Segment	0.16 m³
		accuracy achieved by virtue of active compliance (f/t sensing wrist)		**POWER**	
		SPEED		Mean on board power consumption	120 W
		Maximum EE speed (1-g)	25 cm/s	Peak on board power consumption	160 W
		Maximum EE speed (µ g)	1 cm/s	**COMMUNICATION**	
				Maximum TM data rate	24 kBaud
				Maximum TC data rate	20 kBaud

The future planned activities are:

- definition of the EMATS hierarchical control structure
- definition of the Central Control Subsystem configuration
- definition of Arm Controller and Mobile Base Controller
- preliminary mechanical design
- preliminary specifications

FORCE-REFLECTIVE TELEOPERATED SYSTEM WITH SHARED AND COMPLIANT CONTROL CAPABILITIES

Z. Szakaly, W. S. Kim, A. K. Bejczy

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

ABSTRACT

The force-reflecting teleoperator breadboard described in this paper is the first system among available R&D systems with the following combined capabilities: (a) The master input device is not a replica of the slave arm. It is a general purpose device which can be applied to the control of different robot arms through proper mathematical transformations. (b) Force reflection generated in the master hand controller is referenced to forces and moments measured by a six-d.o.f. force-moment sensor at the base of the robot hand. (c) The system permits a smooth spectrum of operations between full manual, shared manual and automatic, and full automatic (called traded) control. (d) The system can be operated with variable compliance or stiffness in force-reflecting control. Some of the key points of the system are the data handling and computing architecture, the communication method and the handling of mathematical transformations. The architecture is a fully synchronized pipeline. The communication method achieves optimal use of a parallel communication channel between the "local" and "remote" computing nodes. A time delay box is also implemented in this communication channel permitting experiments with up to 8 sec. time delay. The mathematical transformations are computed faster than 1 msec so that control at each node can be operated at 1 kHz servo rate without interpolation. This results in an overall force-reflecting loop rate of 200 Hz.

I. INTRODUCTION

Force-reflecting master-slave manipulator systems are widely used in the industry in high radiation or in other dangerous environments. The major advantage of these systems is threefold. (i) The comparatively direct type of control of a six-degree-of-freedom device since control coordination of six joints in position control mode is inherent to these systems. The master arm movements provided by the operator through the hand grip and the movements of the slave arm fully agree in position, direction and velocity, and are synchronized. (ii) The genuine impression of forces and torques transmitted to the operator's hand with respect to the forces exercised or received. That is, force reflection which kinesthetically connects the operator to the working slave unit. (iii) The relatively high working speed resulting from the direct type of motion control.

The industrial master-slave force-reflecting (or bilateral) manipulator systems have two important characteristics. (i) The master arm is a duplicate (possibly a scaled-down duplicate) of the slave arm. (ii) Force-reflection in the servo-type master-slave systems is implemented through a bilateral-type position control, possibly with some current and differential velocity loops added to it. This means that the basic source of force feedback at the master arm is the position error between master and slave joints and not a genuinely sensed force at the slave.

Evolving capabilities in the technology of advanced robot control and intelligent interaction with remote robots are based on sensing and computing intelligence leading to flexible automation and flexible man-robot interaction. Following the principles of this modern technical approach, a laboratory research system has been developed at the Jet Propulsion Laboratory (JPL) for advanced force-reflecting teleoperation. Force reflection in this system is referenced to forces and torques sensed by a six-d.o.f. force-moment sensor at the base of the robot hand. The master device is a general purpose Force-Reflecting Hand Controller (FRHC), not a replica of any slave arm. It can be applied to the control of different robot arms through the proper kinematic transformations. The mechanism of FRHC is described in [1].

The JPL advanced force-reflective teleoperation system permits a spectrum of operations between full manual, shared manual and automatic, and full automatic (called traded) control, and can be operated with variable active compliance referenced to force-torque sensor in force-reflecting manual control. Shared manual and automatic control is implemented by freezing the data output of the master controller in some task space coordinates which are selectable by the operator from a menu. Motion in the frozen task space coordinates can then be controlled by a computer algorithm which can be referenced to force-moment or proximity sensor information. Variable compliance control is implemented through a low pass software filter in the hybrid position-force control loop. This permits the operator to control a "springy" or less stiff robot. Active compliance with damping can be varied by changing the filter parameters in the software menu. Setting the spring parameter to zero in the low pass filter will reduce it to a pure damper which results in a high stiffness in the hybrid position-force control loop.

First we briefly describe the overall system, its electronics architecture with related software development, and present capabilities. In the second part of the paper we discuss active compliance, communication time delay, experimental results and future development plans.

2. OVERALL SYSTEM

The advanced force-reflecting teleoperation system currently consists of (i) a six degree-of-freedom (d.o.f.) PUMA 560 robot arm, (ii) a smart robot hand on the robot arm equipped with a six-d.o.f. force-moment sensor, grasp force sensors and local processing and control electronics, (iii) a six-d.o.f. generalized Force-Reflecting Hand Controller (FRHC), (iv) two computing nodes for control and information display, one at the robot side and one at the FRHC (control station) site, and (v) computer graphics terminal at the control station site. Each computing and control node is built on a MULTIBUS using NS32016 microprocessors. The communication between the two nodes is on a parallel line. Integrated with each computing node is a compact, computerized Universal Motion Control (UMC) system developed at JPL providing rich motor and state sensing, control, safety and self-test capabilities. The computer graphics terminal utilizes (i) a PARALLAX graphics board to generate a real-time graphics display of force-moment and grasp force information and (ii) an IRIS graphics workstation to generate a real-time perspective graphics image of robot arm motion. Figure 1 shows the schematics of the overall system.

The UMC architecture and capabilities, developed at JPL, have been described in several earlier publications ([2] and [3]), where they can be found in more detail. In short, the UMC electronics consists of PWM power amplifiers for up to 1 kW motors and provides sensing of motion parameters at servo rates 1000 Hz. The communication from the motor control elements to the joint processor is a private bus called the BLX bus that makes the joint motion parameters memory mapped. It is notable that with the UMC up to 16 joints can be controlled by a single joint servo processor. The processor currently used is the NS 32016. There is a large number of processors from which we could choose. The NS 32000 family has proven to be a very good candidate for our task. The family has a number of processors with a wide performance range and object level compatibility between the members. Its assembly language has proven to be powerful as well as easy to use. The UMC electronics, thanks to the NASA Technology Utilization program, is now available commercially for up to 10 kW motors either brushed or brushless [4].

2.1 Electronics Architecture

To save development time we used the DB32000 development board which comes with a MULTIBUS interface. This forced us to use MULTIBUS for interprocessor communication. This is a lower bandwidth bus than more recent 32-bit busses, but the available bandwidth is more than enough for our application so the use of MULTIBUS did not hamper the performance of our system. With the upcoming development of new processor boards (still using the 32000 family) a new proprietary bus (the ZBUS) will be introduced that is optimized for high bandwidth shared memory applications.

The internode communication is done via a parallel port that carries one byte periodically at every 125 microseconds in each direction. The narrow bandwidth and periodic use of the communication channel are important parameters. If the usage of the channel is not periodic that means that the bandwidth has to be higher than the number of bytes transmitted per second. This is a waste of the channel bandwidth. This 125 μsec byte transfer rate is also used to synchronize the remote node to the local one. Eight bytes are transmitted in every servo loop from the local to the remote node. The first one is the header byte that is used to determine which byte belongs to which degree of freedom. This is followed by the position change of the X, Y, Z, pitch, yaw, roll degrees of freedom. The communication is done in relative Cartesian coordinates. In every servo loop a change in the range of -7 to +7 is transmitted. These changes are added by the receiver to the robot Cartesian position setpoint number. This method has a number of merits: (i) Small communication bandwidth used. (ii) Error tolerance in communication. (iii) Velocity limiting. (iv) Easy method of indexing the robot. It should be noted that this communication method does not cause any granularity in robot speed whatsoever. It simply limits the granularity of the robot position to 1/10th of a mm. The robot could not be positioned more accurately than that anyway.

The parallel internode communication cable in the future will be replaced by a fiber optic link with a much higher bandwidth, but the principle of communication between the two sides will remain the same.

Artificial time delay between the "local" and "remote" computing nodes has also been implemented to allow the experimental man-in-the-loop study of the effect of communication time delay on control performance. The time-delay MULTIBUS cardcage between the "local" and "remote" computing nodes contains two processors performing the time delay function between 2 ms and 8 sec.

In summary, the "local" node cardcage contains:

- Two joint interface cards (part of local UMC)
- PWM amplifiers for 8 motors (part of local UMC)
- Joint processor (part of local UMC)
- Kinematic transformation processor
- Communication processor with user interface
- Graphics processor
- Parallax graphics card

The "remote" node cardcage contains:

- Remote node UMC (3 cards and power amplifiers)
- Communication processor
- Inverse kinematics processor
- Forward kinematics processor

The communication from the smart end effector to the "remote" node and from the "remote" node to the IRIS graphics robot simulator is via fiber optic RS232 lines at 9600 baud rate.

Figure 2 shows the block diagram of the system and interconnections. Figure 3 indicates the timing of events and the sequence of computations. All computations are carried out at a 1000 Hz servo rate. The force feedback signal is currently received at a 125 Hz rate due to the limitation of the RS232 communication channel used between the Smart End Effector and the communication processor. The total round trip time delay is 5 ms for the position error-based force feedback and it is around 10 msecs for the sensor-based force feedback.

2.2 Software System and Development

The programming language used was the assembly of the NS 32016 itself since this promised the most performance and the fastest results. It has to be noted that the most convenient development environment such as a C cross compiler and UNIX operating system does not necessarily produce the fastest result and the best program performance. Compilers have the tendency to mask the real world of a processor from the programmer making it harder to generate complex interrupt hierarchies and hardware interfaces. We used a development system that one of us (Szakaly) wrote for the IBM-PC. This system makes it possible to edit and store the assembly source programs in the PC as well as up and download object files. In the current version an integrated assembler, developed at JPL, is used which runs on the IBM-PC. Portions of the system such as the force torque graphic display were developed in C using the SYS 32/20 development system marketed by National Semiconductor.

The motor control algorithm is a simple PD control loop. The servo rate is 1000 Hz overall allowing high gains to be used with the associated high tracking fidelity. The position gains are about 0.1 V/encoder unit. The UMC code generator program is used in the joint level controller. This program assures safe robot control by automatically generating the servo code that controls the joints. There is a set of parameters that have to be specified once for every robot. These parameters are stored in an electrically erasable EEPROM chip. When the program is activated it generates servo code and executes it. There is no possibility of breaking the robot due to human error in the coding.

The code generator is very flexible, it can control any number of motors up to 16, with any combination of hardware elements such as encoders, pots, temperature sensors, motors, brakes. All polarities are menu items so, for example, instead of having to switch the two encoder wires the user changes the encoder polarity from 'POS' to 'NEG' in the menu. The code generator will use a SUB instruction in place of an ADD in the servo code to accommodate the negative encoder hookup. The motor, the pot, the index and brake polarities can similarly be changed from the menu. The motor control processor interfaces to the rest of the system via the shared memory. More on the UMC software can be found in [5].

Since the remote node receives Cartesian position set points the inverse kinematic transformation is needed to calculate the robot joint position setpoints. This is carried out by one of the processors on the robot side. This transformation was implemented in integer arithmetic and takes around 700 μsecs to execute. Force feedback to the HC is based on robot position error as well as on force-torque sensor data so the robot end effector Cartesian position has to be computed as well. This is done by computing the robot forward kinematics.

The user has a large number of options available through the user interface. Every parameter can be changed on a degree of freedom basis. It is possible to activate a software spring for rate control on any degree of freedom that pulls the user's hand back to a center position. Any DOF may be in position or rate mode or it may be turned off. Any degree of freedom can have arbitrary force compliance with a zero or non-zero force setpoint. For example, orientation compliance with zero torque setpoint amounts to automatic peg alignment when performing peg insertion into a hole. An X compliance with non-zero force setpoint will press the end effector against the task board and will maintain contact force. Rate mode is useful when motion over large displacements is desired or when slow, constant velocity motion is the requirement.

Extensive experiments have been conducted to evaluate the usefulness of these operating modes and force feedback. The data shows that force feedback brings an improvement in terms of execution time as well as total force required. The shared control routines also bring about additional improvements. The experiments and results are described in detail in [6-8]. The ongoing experiments are concentrated on time-delayed operations using active compliance.

3. ACTIVE COMPLIANCE CONTROL

Variable active compliance has been implemented as a new feature to the current force-reflecting telerobot system. In a conventional telerobot system, each joint is controlled by a very stiff position servo, and thus the human operator has to control a stiff telerobot hand. A stiff telerobot hand tends to hit or bump into objects or walls hard. The implementation of active compliance, which emulates a programmable mechanical passive spring by computer software, allows the human operator to control a compliant or springy telerobot hand, not a stiff one. The compliant hand tends to touch objects or walls softly without exerting much force. It is also compliant to the environmental constraint, facilitating telemanipulation task performance. For example, in the peg-in-hole task, the compliant hand adjusts itself in accordance with the hole structure.

In order to implement active compliance on the telerobot hand, the force/torque signal sensed by the force/torque sensor (FTS) is first low pass filtered by computer software, and then fed back to the position/orientation output command signal (Figure 4). The force/torque sensor consisting of 8 pairs of strain gauges furnishes 3 force components (x, y, z) and 3 torque components (roll, pitch, yaw). Each of these 6 components, after low pass filtered, is individually fed back to the corresponding position (x, y, z) or orientation (roll, pitch, yaw) command input which comes from the hand controller controlled by the human operator. The mechanical equivalent of the above implementation consists of a spring connected in parallel with a damper (Figure 5). There are two parameters to control: compliance (or its inverse, stiffness) and damping (friction). The compliance of the active spring is proportional to the force feedback gain K. The damping (friction) of the active damper is proportional to T/K, where T is the time constant of the first-order low pass filter. In general, higher force feedback gain results in more compliance (less stiffness), but requires more damping (more sluggishness) to stabilize the system. If a pure gain is used instead of the low pass filter, a spring with no damper is realized. But, this turns out to be unstable. If an integrator is used instead of the low pass filter, a damper with no spring is implemented. A damper-alone system has a saturation problem due to the lack of a spring which allows "return-to-center" or enables the system to come back to the normal operating region when there is no force sensed. After a few runs of telemanipulation tasks, the damper tends to saturate and the damping effect disappears in the saturated direction.

Since a compliant telerobot hand is now implemented and available, the following two human-telerobot shared control schemes are suggested for efficient telemanipulation, depending upon the time delay. When the time delay is less than 1 second, approximately, both force reflection (long loop between the human operator and the telemanipulator) and active compliance (telerobot autonomous loop) can be used (Figure 6A). It is observed that a compliant telerobot tends to stabilize the force reflection long loop, and thus force reflection can be still useful even when the time delay is longer than 0.5 second. This also implies that the fidelity of the force reflection from the telerobot hand to the force-reflecting hand controller can be improved.

When the time delay is greater than 1 second approximately, active compliance alone without force reflection can be used (Figure 6B). This active compliance scheme turns out to be extremely useful when there is a long time delay. For example, a peg-in-hole task was successfully accomplished by a human operator even with 8 seconds time delay. It was not possible without active compliance. It took about 0.5 to 1 minute to complete the task with no time delay, 3 minutes with 4 seconds time delay, and 7 minutes with 8 seconds time delay. This proves the significance of the use of active compliance, since a conventional force-reflecting telemanipulator without active compliance cannot be used beyond about 0.5 seconds time delay due to the stability problem. The use of a compliant telerobot hand is also important for safety reasons, because the compliant hand tends to touch a wall softly without exerting much force or bumping into it hard.

4. FUTURE PLANS

To support a long list of man-machine interaction research topics (dual and redundant arm control, dexterous end effector control, coordinated manipulator and visual system control, etc.), the future developments in control electronics and control computing include: (a) An advanced bus architecture to eliminate the bottlenecks of commercial bus systems. (b) New processor cards using two NS 32016 processors or the NS 32332 processor within the advanced bus. (c) 5 Mbit and 15 Mbyte fiber optic links. (d) New smart hand electronics featuring very high (10 kHz) data rates with 12 bit A/D and with fiber optic link. (d) A new assembler to provide an environment similar to Turbo Pascal and other integrated development systems. After some experience with the new assembler improvements will be made to the syntax such that the usage will have the appearance of a high level language. This will provide many of the benefits of high level languages without the associated performance and control loss. This will facilitate to upgrade and expand the control software with new performance capabilities in supervisory control.

ACKNOWLEDGEMENT

This work was carried out at the Jet Propulsion Laboratory of the California Institute of Technology under contract to the National Aeronautics and Space Administration. The computing architecture, electronic hardware and software design and coding was done by Z. Szakaly. Active compliance control was implemented by W. S. Kim.

REFERENCES

[1] Bejczy, A. K., Salisbury, J. K. Jr., Controlling Remote Manipulators Through Kinesthetic Coupling. Computers in Mechanical Engineering (CIME), Vol. 2, No. 1., July 1983, pp. 48-60.

[2] Bejczy, A. K., Szakaly, Z. F., Universal Computer Control System (UCCS) for Space Telerobots. Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, NC, March 30 - April 3, 1987, pp. 318-324.

[3] Bejczy, A. K., Szakaly, Z. F., A Synchronized Computational Architecture for Generalized Bilateral Control of Robot Arms. Proceedings of the Conference on Advances in Intelligent Robotic Systems, SPIE and the International Society for Optical Engineering, Cambridge, MA, November 1-6, 1987, SPIE Vol. 851, pp. 123-134.

[4] Motion Tek, Box 9, Lord Ave., Brunswick, NY 12180.

[5] Szakaly, Z., and Bokor, E., Teleoperator Subsystem/Teleoperator Demonstrator: Hand Controller/Universal Controller Software Document. _JPL D-5173_, (Internal Document) January 1988.

[6] Bejczy, A. K., Hannaford, B., and Szakaly, Z., Multi-Mode Manual Control in Telerobotics. _Proceedings of ROMANSY'88_, Udine, Italy, September 1988.

[7] Bejczy, A. K., and Hannaford, B., Man-Machine Interaction in Space Telerobotics, _Proceedings of the International Symposium on Teleoperation and Control_, Bristol, England, July 1988, IFS Publ., Bedford, U.K., pp. 135-149.

[8] Hannaford, B., and Wood, L., Performance Evaluation of a 6-Axis High Fidelity Generalized Force-Reflecting Teleoperator. _Proceedings of NASA Conference on Space Telerobotics_, Pasadena, CA, January 1989. (See elsewhere in this proceedings.)

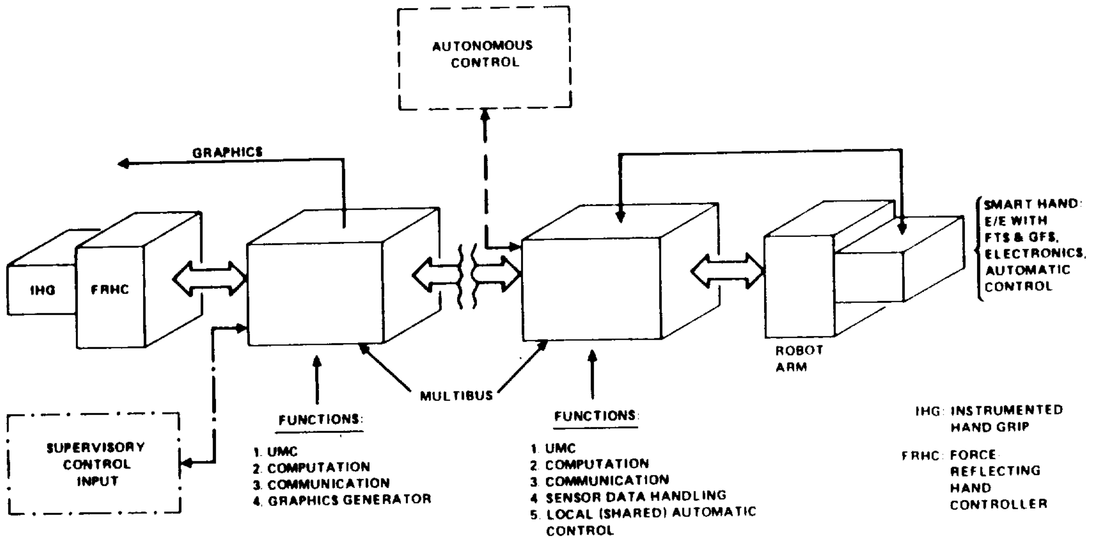

AUTONOMOUS CONTROL
GRAPHICS
IHG
FRHC
SMART HAND: E/E WITH FTS & GFS, ELECTRONICS, AUTOMATIC CONTROL
ROBOT ARM
MULTIBUS
SUPERVISORY CONTROL INPUT
FUNCTIONS:
1. UMC
2. COMPUTATION
3. COMMUNICATION
4. GRAPHICS GENERATOR
FUNCTIONS:
1. UMC
2. COMPUTATION
3. COMMUNICATION
4. SENSOR DATA HANDLING
5. LOCAL (SHARED) AUTOMATIC CONTROL
IHG: INSTRUMENTED HAND GRIP
FRHC: FORCE REFLECTING HAND CONTROLLER

Figure 1. Overall Advanced Teleoperation System with Distributed Computing.

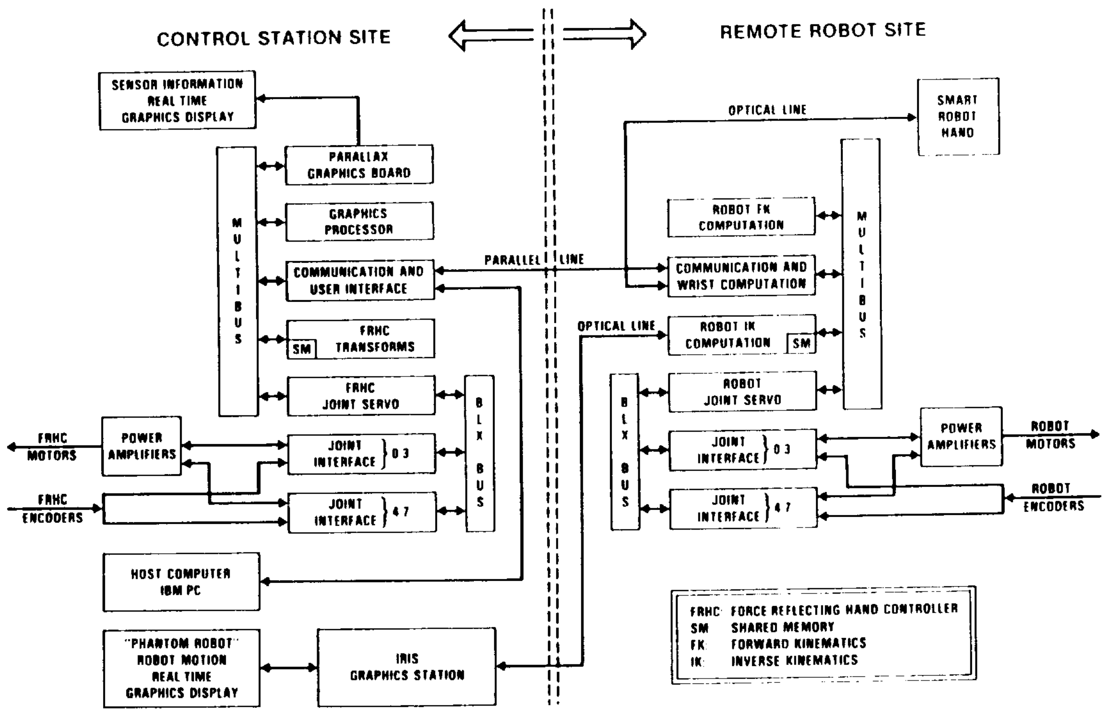


Figure 2. Electronic and Computation System Block Diagram.

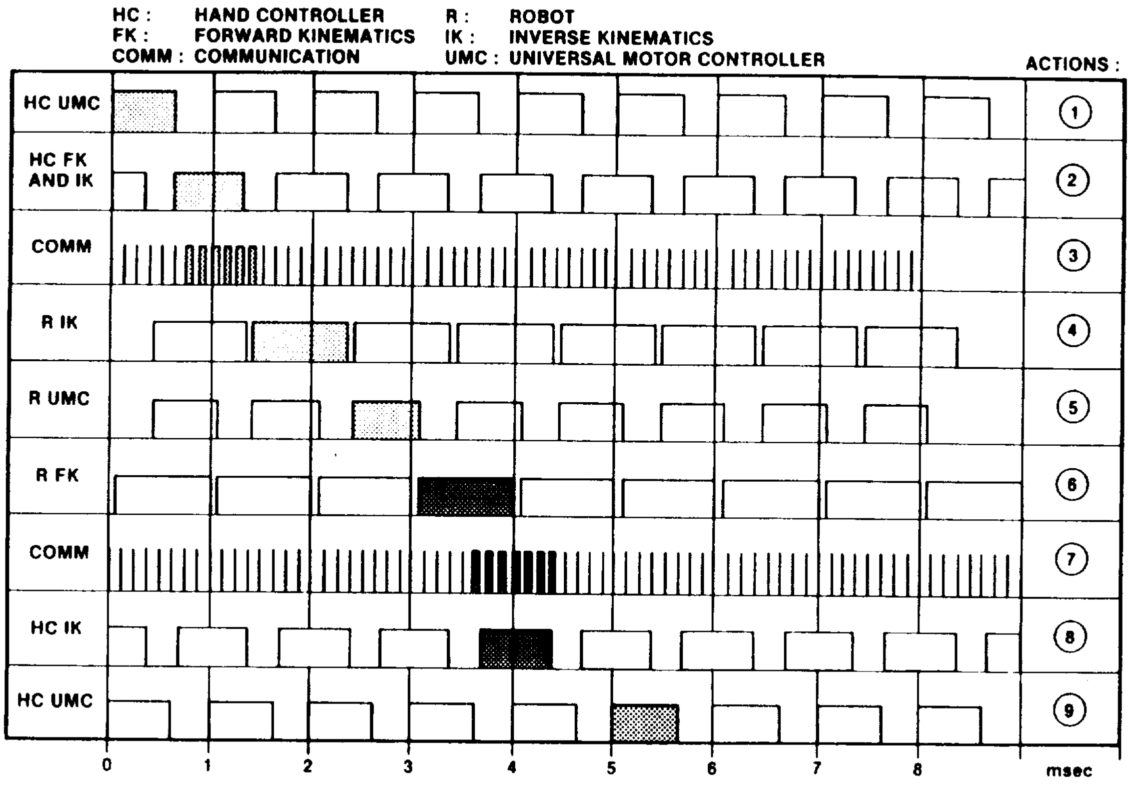


Figure 3. Synchronized Computational Event Sequence Block Diagram.

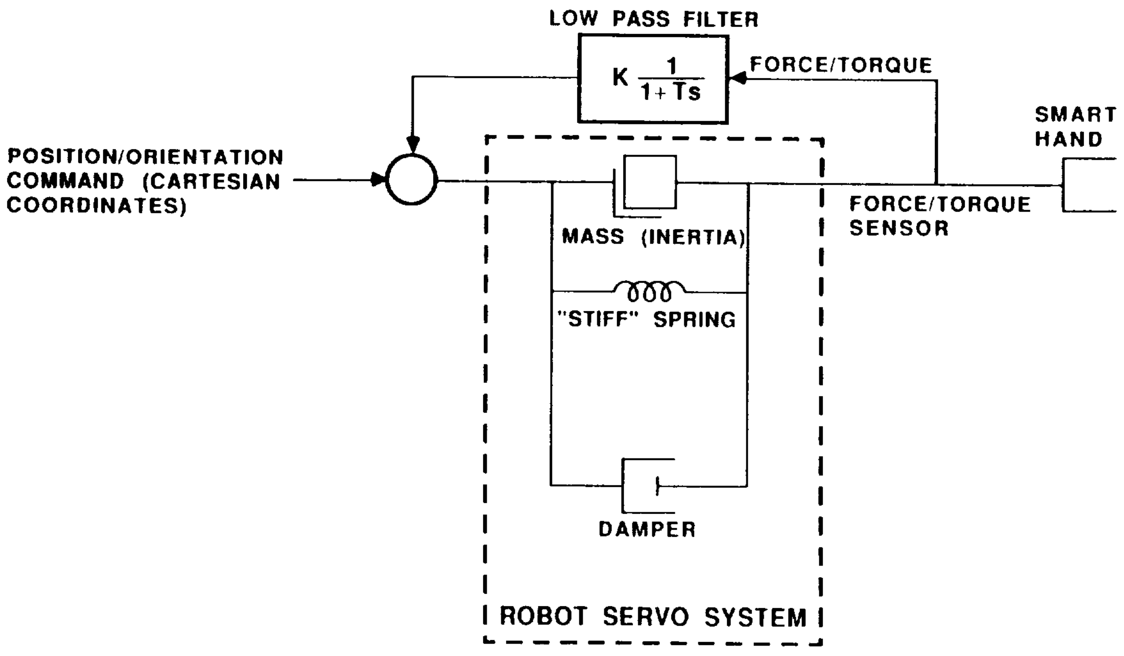


Figure 4. Implementation of Active Compliance for Shared Control.

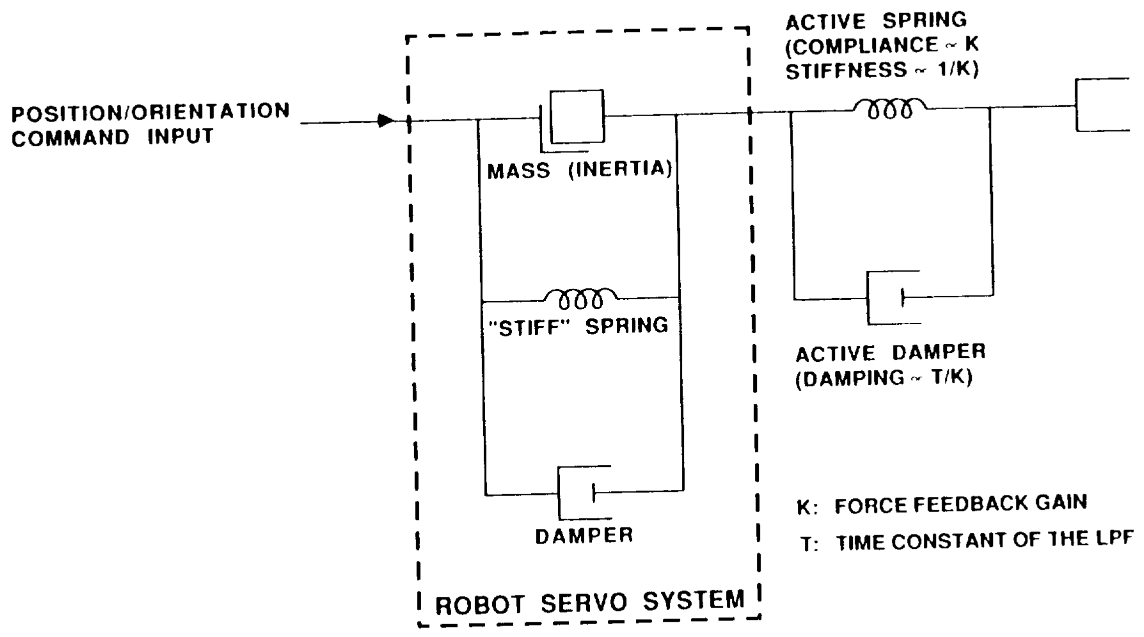

POSITION/ORIENTATION COMMAND INPUT
MASS (INERTIA)
"STIFF" SPRING
DAMPER
ROBOT SERVO SYSTEM
ACTIVE SPRING (COMPLIANCE ~ K STIFFNESS ~ 1/K)
ACTIVE DAMPER (DAMPING ~ T/K)
K: FORCE FEEDBACK GAIN
T: TIME CONSTANT OF THE LPF

Figure 5. Mechanical Equivalent of Active Compliance Implementation.

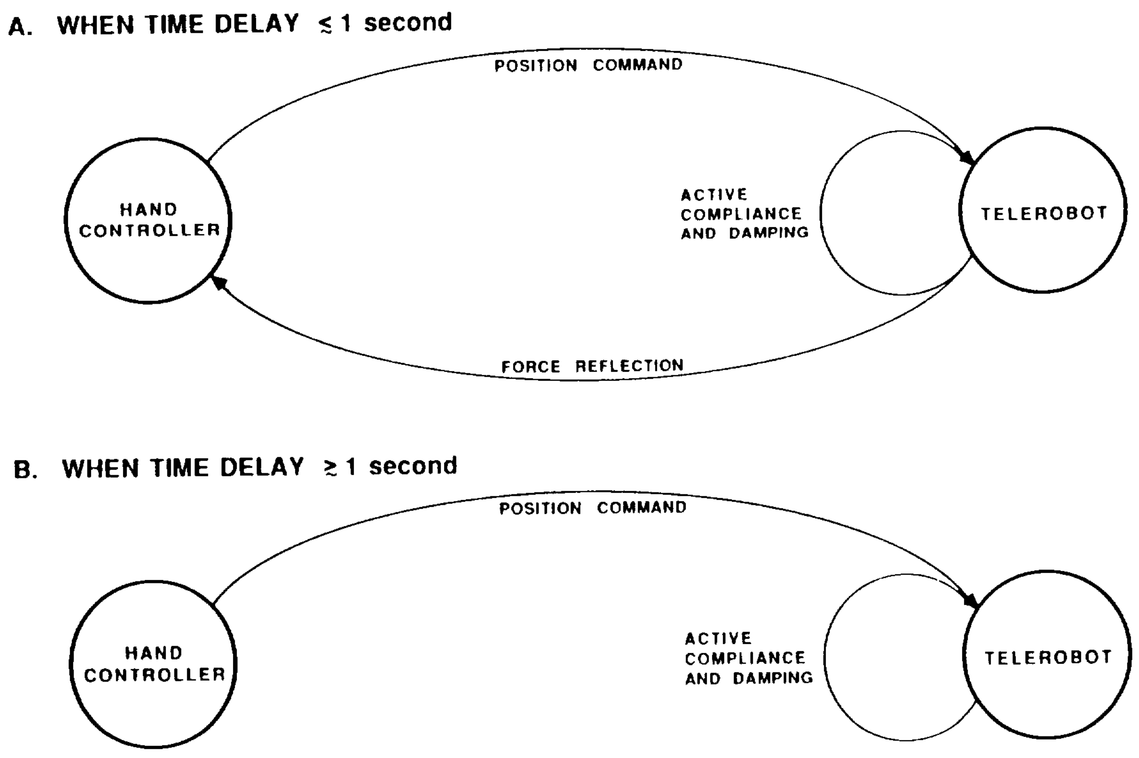

A. WHEN TIME DELAY ≤ 1 second
POSITION COMMAND
HAND CONTROLLER
ACTIVE COMPLIANCE AND DAMPING
TELEROBOT
FORCE REFLECTION
B. WHEN TIME DELAY ≥ 1 second
POSITION COMMAND
HAND CONTROLLER
ACTIVE COMPLIANCE AND DAMPING
TELEROBOT

Figure 6. Control Schemes for Using Active Compliance Control.

N90-29346

Information management in an Integrated Space Telerobot

(+)S. Di Pippo (*) G.Pasquariello (+)G.Sylos Labini

(+)ASI–Italian Space Agency
V.le Regina Margerita 202
Rome Italy

(*)IESI–CNR
Via Amendola 52
Bari Italy

ABSTRACT

The in–orbit operations, like space structures inspection, servicing and repairing, is expected to be one of the most significant technological area for application and development of Robotics and Automation in Space Station environment.The Italian National Space Plan (PSN) has started up its strategic programme SPIDER (SPace Inspection Device for Extravehicular Repairs) in the early 1987, this program is now continued by the Italian Space Agency that in may 88 have take over the role of national agency for space activities.SPIDER programme is scheduled in three phases, with the final goal of performing docking and precision repairing in the Space Station environment.SPIDER system is an autonomous integrated space robot, using mature Artificial Intelligence tools and technics for its operational control.This paper describe the preliminary results of a joint study between ASI and IESI on the information architecture of the spacecraft.

I. INTRODUCTION

The main goals of SPIDER system are visual inspection in a fly–around mission and precision repairing activities on the space structures.
The main characteristics of SPIDER are the following:
- small dimension (-1 mc) and low weight (- 400 kg.)
- "biological" evolution capability
- retrievable by a platform or spacecraft based robotic arm.

The SPIDER programme is scheduled in three main phases:.

-in the first phase, SPIDER will be a space vehicle for visual inspection around large and/or small space structures, by means of a flying–around approach. In this phase, SPIDER will be strictly teleoperated

-in the second phase, SPIDER capabilities of autonomous navigation will be extended limiting commands to very high level instructions. The human operator will act in this phase only as a supervisor

-in the third phase, SPIDER will be able to do docking and repairing, increasing its autonomy. The presence in this phase of two small cooperative arms (linear dimension - 1m) and a docking robotic arm will allow to operate precision repairing and micro-manipulation capability.

In the following, we describe mainly the SPIDER-I system.
SPIDER-I mission, around the space structures, will allow to:

- test SPIDER fly-around capability

- support visual inspection of external devices

- find damaged areas of space structures, increasing crew safety and reducing dangerous extravehicular human interventions.

In the SPIDER evolution through the different described phases will be followed by a similar modularity in the Robotics Intelligence Subsystems. In section II we give the requirements for the SPIDER -I design reference mission. In section III we give a general platform description, in section IV the Architecture of the control system is depicted. In section V the relevant aspects concerning the interaction of sensor and controls SPIDER subsystems are described .

II. SPIDER-I Design Reference Mission

In order to define the SPIDER-I system specifications, the following LEO external visual inspection scenario has been hypothesized:

- SPIDER should be deployed in LEO by a space transportation system (STS, HERMES,OTV...).

- Fly by the target structure (eg. MTFF, ISS).

- Demonstrate proximity operation capability flying at a fixed distance from the coorbiting structure at different relative velocities

- Report accurate image and other information about the external environment and the target

- Exploit passive docking capability

- Admit a mission duration of at least 1 h and a station keeping period of 24 h before a retrieval operation performed by other space transportation systems or orbiting permanent facilities.

The SPIDER system specifications has been defined on the basis of these mission requirements.

III. PLATFORM DESCRIPTION

The SPIDER is a small dimension free-flyng spacecraft. It's aspect is that of a cylinder with polygonal bases and with a design reference mass of 400 kg.
The SPIDER overall dimension are:

-Length 150 cm

-Diameter 90 cm

The Robotic Intelligence System (RIS) is located the forward of the spacecraft with a mass of about 170 kg. The spacecraft reaction control system (RCS) will permit medium-range and proximity operations. The cold gas will be used as propellant to prevent pollution of optical or other exposed surfaces during proximity operation. The use of low-impulse trhusters (e.g. electrical propulsion) will be considered for the SPIDER-III mission. The first prototype will be equipped with 2 x 12 high impulse (30 Nw) cold gas trhuster for rotation and/or coarse movements and 2 x 4 low-impulse cold gas trhusters for precision movements. Four tanks will assure a total impulse of 4000 Nw/s.
Two third of the spacecraft will be covered with solar cells arrays in order to provide 150 Wh for each orbital period. A set of rechargeable batteries will provide power during eclipse.

IV. ROBOTICS INTELLIGENCE SYSTEM

The RIS will support in the final SPIDER prototype all the main high level function of the spacecraft:

-On board data handling

-Guidance and Navigation

-Perception and Reaction

-Man Machine Interfacing

-Remote Manipulation

Also if in the first phase some of this function will be mainly controlled by the remote human operator, the SPIDER system will exploit an intelligent behavior in the area of guidance and navigation,man machine interface and mission planning. In the following the RIS architecture and information flow are described.

1. FUNCTIONAL ARCHITECTURE

In order to exploit this "intelligent" behavior the RIS provide a set of specialized modules for different tasks (Fig. 1). This solution permits:

- a concurrent processing of different tasks.

- a specialized Knowledge structures

- a substantially fail-safe design

The interaction between the processes is performed through a "blackboard" that shares common interest information. The coexistence of different priorities among different processes forces a substantially asynchronous access at the blackboard. So, we can refer to our system as a white-board architecture /1/.
In order to extend the module design flexibility and the system design modularity, the information shared in the blackboard must be, as far as possible, process independent, in the sense that any process can access it in the more easy way.

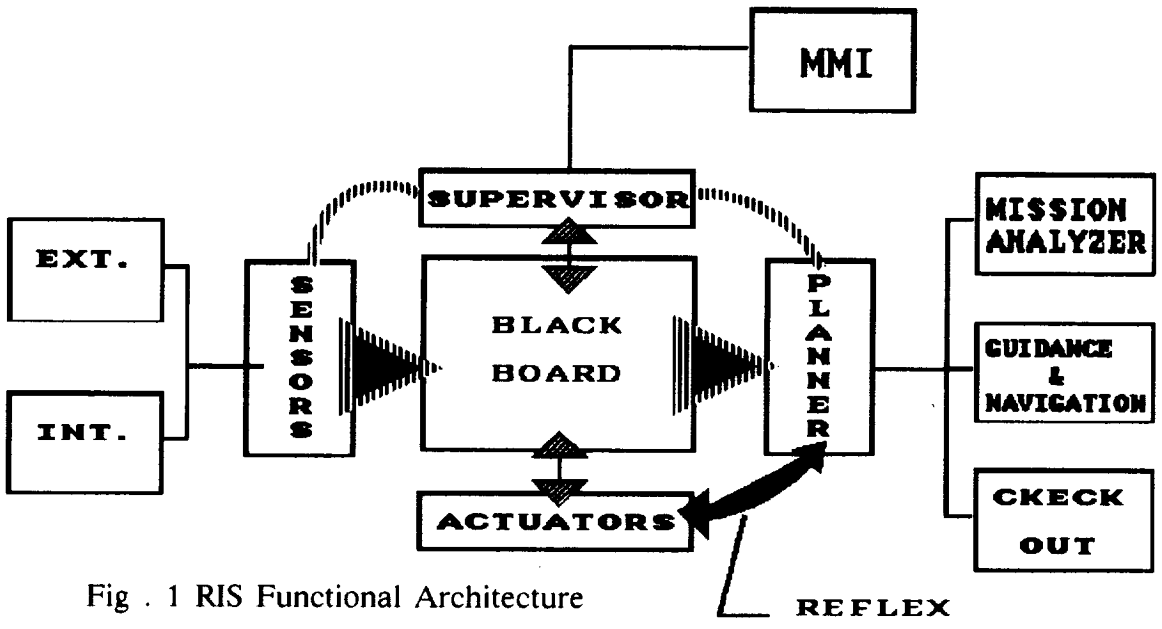


Fig . 1 RIS Functional Architecture

Supervisor

The white-board architecture claims for a Supervisor module that handles the specialized module priority, use of parameters, synchronization and so on. In addition, the role of such a module is of reporting all operator commands and triggering blackboard maintenance process.
This module is structured in Internal and External sensors. The Internal Sensor module provides to the Knowledge Base all information about spacecraft internal state, classical subsystems included. The External Sensor module handles all the information about the

outdoor space, such position and velocity of the spacecraft, target characteristics, etc. A deep difference exists between these two modules, because while the first interacts mainly with maintenance and internal monitoring process, the latter is involved both with main spacecraft task (inspection) and the guidance and navigation
subsystem. In order to perform these activities an extensive use of sensor information fusion is made. This goal is achieved using as interface among external sensor module and blackboard system a set of "virtual sensors" /1/,/2/.
These sensors are obtained using different specialized sensor information in order to obtain high level information (e.g. depth and color). The virtual sensor characterization is driven by the supervisor module using the requests made by the operator or by other modules. In tab.1 a set of the SPIDER
proposed external sensors is shown.

Absolute position sensors

- **Star Sensors**
- **GPS Receiver**

Relative position sensors

- **Continous wave laser**
- **Accelerometers**
- **CCD cameras**
- **Mw sounder**

Tab. 1

We have already mentioned the sensor fusion task; it must be noted that different purposes can be met by means of these definitions /3/:

Sensor cooperation	- Use of mixed perceptual information
Sensor competition	- Use different sensors for the same task with different accuracy
Sensor independence	- No interaction between sensors

Anyway, the complete definition of the external sensor subsystem will be made after completion of a certain number of technological assessment studies expected for the end of the 1988.

Planning

The Planning system controls the system reasoning on the data acquired by the sensor information module, on the directive made by the module and drives self check procedure of the spacecraft. It also converts High Level directive in execution sequences, usable by SPIDER subsystems.

In order to perform these different tasks the Planning module is decomposed in three modules:

Mission Analysis

Guidance & Navigation

Check-Out

The first module is an aid to the human operator that permits a fast evaluation of the mission requirements, or of a subset of operation connected with a SPIDER mission. The Guidance and Navigation module implements the missions proposed by the Mission Analysis module and/or using information from the blackboard, implementing also collision avoidance maneuver.The Check-Out module is a diagnostic expert system, specialized to the SPIDER subsystem trouble shooting. It also maintains records of malfunctioning of spacecraft sections.It's important to point out that the planning system cannot directly implement its decisions, except that in the case of severe danger for the spacecraft or for others coorbiting objects, in that case the system will bypass human control implementing the reflexive behavior.

Man Machine Interfacing

This is one of the key subsystem of the SPIDER system. Infect, in a teleoperated system, the man machine interaction must permit an easy and high level dialogue between the operator and the spacecraft. In this area, all state-of-the-art tools will be used in order to argument the: scene rendering, situation simulation, alarm transmission. Also in this area specific studies are ongoing in order to select appropriate devices and software tools.

Actuators

This subsystem is the interface between the Robotic Intelligent System and the other spacecraft subsystems that implements the directive processed by the RIS or directly transmitted by the human control.Naturally it comprends the RCS, the thermal control, and the power subsystems. In the following, spacecraft evolution will comprend also the active docking mechanism and the manipulation arms and end-effectors.

Hardware Architecture

The hardware implementation of the described functional architecture will face with several problems mainly connected to the space qualification of terrestrial processors and software. Moreover, a lot of problems that do not yet even have a solution in ground based situation should be resolved. An other point susceptible of carefully evaluation is that of the bandwidth of the radio-link among the user station and spacecraft and the choice of the format of image supported information transfer. This problem will be obviously related with the space qualified hardware available and with the trade off in the distribution of computational charge among the spacecraft and the user station.
Also the opportunity of using a ground based workstation will be evaluated having in mind:

-redundancy

-signal propagation delay

-computational power

2. INFORMATION FLOW

The first SPIDER mission will be a demonstration flight that will show the capabilities of the system in the free fly and user supervised inspection. A generic SPIDER-I mission is brunched in the following Tasks:

A- Deployment by a space transportation system

B- Free-fly to the target

C- Target Inspection

D- Free-Fly Back to a Station Keeping Position

E- Retrieval

Phase A and E are for the SPIDER-I spacecraft quite passive task, i.e. the spacecraft will have only a passive docking interface that will be docked to the deployer. Phase B and D are substantially similar task with main difference in the start and end point. Task C is the core of the SPIDER-I mission in witch the external inspection capabilities of the spacecraft will be tested.At the purpose of demonstrate the compatibility of the described RIS functional architecture with these tasks, we have analyzed the structure of each task and pointed out their mutual relation and interaction with the spacecraft subsystem.The resulting representation is shown in Fig. 2.

1. Task A and E

As already shown the SPIDER during these phase will be totally passive. The only task that must be performed is the complete system Check-Out. A rule of the type:

IF Check-Out-Succes THEN Deploy

IF Check-Out-Succes THEN Retrieve

Will be the only condition-action pair that will control both the operations.

2. Task B and D

After the deployment the SPIDER will be in a Station-Keeping position. The problem is to implement the best trajectory to the target. In the problem definition, and without a loss of generality we have supposed that:

-There is no relative motion between SPIDER and the Target at the moment of deployment.

-And that the Spider trajectory to the Target is described by a plane curve.

The subtask of the free-flyng among two position are:

-Path Planning

-Check Out

-Guidance and Navigation

The Path Planning must compute a transfer orbit for SPIDER from the Start Point to the target proximity. Their input data must be :

-Spider State Vector at the Deploy

-Target State Vector

-Internal Subsystem State

-Known Obstacles on the Path (IF any)

-A Path Optimization Criteria

The Output of this process will be a Transfer orbit and a firing sequence for the RCS.The Check-Out will provide information of Spacecraft Faults, if any, a fault recovery analysis if possible. These process, active in all the Task of SPIDER, will change is focus of attention depending the actual system state and on the basis of the actual task goal.The guidance and navigation must pilot the spacecraft from the start to the end point implementing the strategy selected by the Planner. In practice his task is to compare external sensor readings with the data suggested by the planner, and if evidence of a divergence from the path is found send a request to the planner for a new plan. The G&N will also implement a collision avoidance strategy, if unknown obstacle are detected by the external sensors.

3. TASK C Target inspection

The target inspection, as described, will be one of the main goal of the SPIDER demonstration flight.

C-3

During this part of the mission the spacecraft should demonstrate it's capability in proximity flight, self-planning, and high-level command interpretation capability. Main sub-tasks of Target Inspection are:

-Proximity Flight
-External Sensor Data Handling
-Target State Vector

Proximity Flight

During proximity flight SPIDER will free-flight at a fixed distance around the target. At proper angle must zero is velocity respect to the target, and activate proper External Sensor. A tight interaction exist in this phase between:

-Planner
-Guidance and Navigation-
-External Sensor Data Handling.

In fact, in the actual system configuration, the external sensor are connected in a rigid way to the spacecraft

External Sensor Data Handling

This sub-task control all the data flow between external sensors and RIS. Implementing also the virtual sensor requested by different process (mainly by planner).
The ESDH send also request to the planner in order to force a stop of the spacecraft if requested by some sensors.
High level subtask of ESDH are:

-Stop and Zoom on operator Request
-Special Target part Recognition (thermal shields,solar arrays etc..)
-Supervised Inspection

Target State Vector Determination

This High Level Task determine the target center of mass motion parameters, in the SPIDER frame of reference, and target motion around its center of mass, to do that send request to the ESDH in order to activate proper Virtual Sensors connected to range finders and microwave sensors.
Data produced by this task are then used by the planner in order to correct the SPIDER orbit or implement user request.

V Conclusions

Actually the described architecture has been implemented using a commercial tool that offers knowledge representation facilities both in form of rules and frames. The explicit goal of this activity is to better understand the information flow and the knowledge

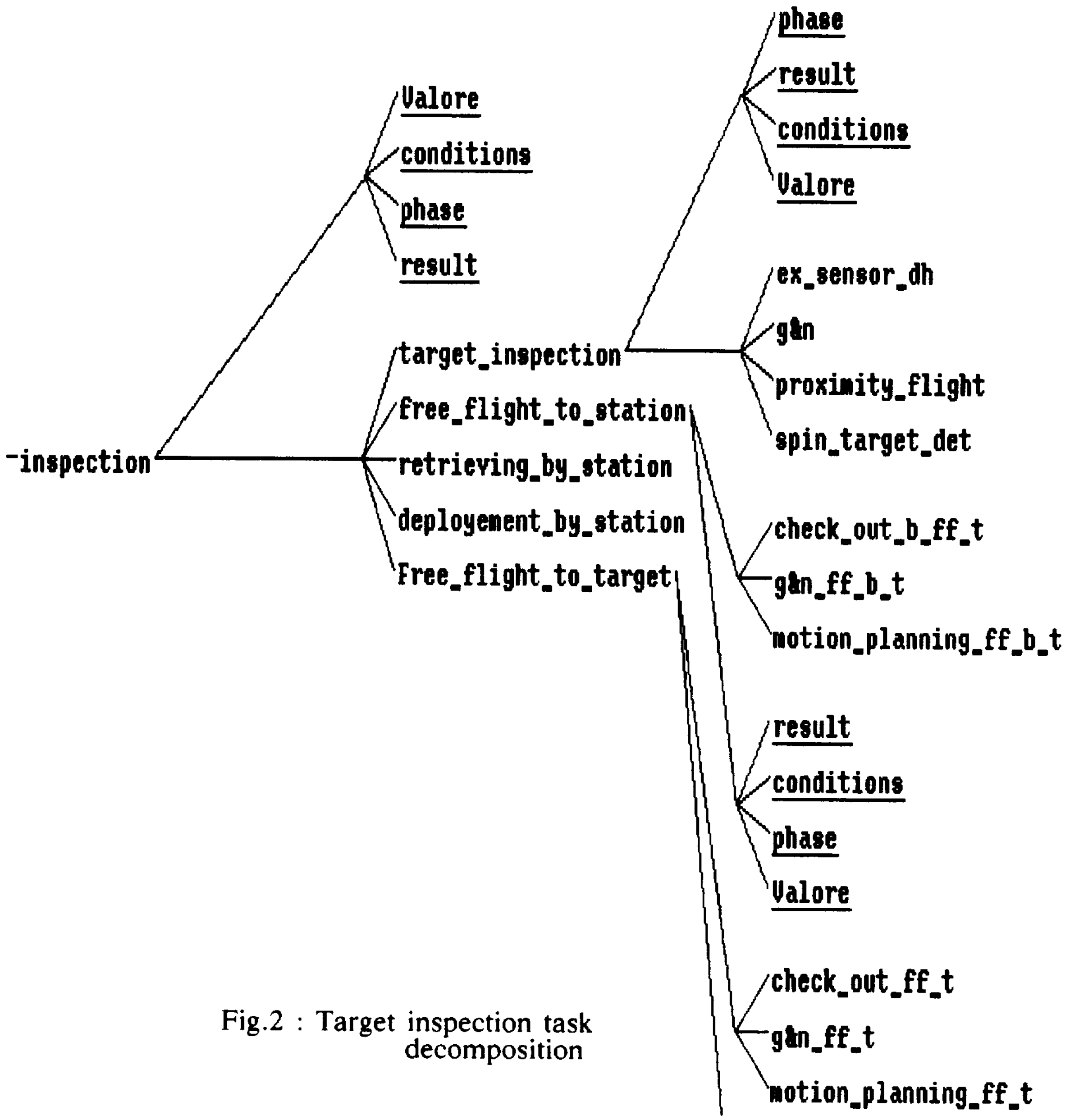


Fig.2 : Target inspection task decomposition

structure in order to define the final architecture design for the high level components of SPIDER robotic intelligence subsystem.

Bibliography

/1/ M.Daily and Others- Autonomous Cross Country Navigation with the ALV - Proc. IEEE conference on Robotics and Automation -April 88-

/2/ An Architecture For Reflexive Autonomous Vehicle Control - Ieee Conference On Robotics And Automation - April 86.

/3/ A. Stenz & Y. Goto - The CMU Navigational Architecture - DARPA Image Understanding Workshop -feb. 1987

N90-29847

Redundancy in Sensors, Control and Planning of a Robotic System for Space Telerobotics

Prof. A.Rovetta, Eng. S.Vodret, Eng. M.Bianchini

Department of Mechanics, Politecnico di Milano.

Abstract

This paper discusses the analysis and development of a manipulator redundant in structure and sensor devices controlled by a distributed multiprocessor architecture.
The goal has been the realization of a modular structure of the manipulator with evident aspects of flexibility and transportability.
The distributed control structure, thanks to his modularity and flexibility could be integrated in the future into an operative structure aimed to space telerobotics.
The architecture is applied to the 6 DOF manipulator Gilberto, developed at Department of Mechanics, Politecnico di Milano.

1. Introduction

The experimental activity of research and development has been originated by the precise need of improvement and integration of different indipendent projects already advanced in Department of Mechanics, Politecnico di Milano described as follows:

1) Development of 6 DOF robot with voice control system
2) Development of a dexterous hand provided with sensors and advanced control capabilities
3) Development of vision systems, with single and multiple cameras, for pattern recognition and objects analysis
4) Study of an expert system oriented to obstacle avoidance and path optimization
5) Application of a simulator for assembly problems solving

In this first phase the activity consisted in the optimization of a traditional manipulator with 6 DOF and his upgrading into a flexible structure provided with a hierarchical hardware control structure and relative software in order to make feasible real time control with a sufficient level of precision and throughput.
This work was mainly concerned with analisys and first development of a modular architecture both from hardware and software point of view.

2. Structure requirements

Thinking to the tipical needs of telemanipulator applications, it has been decided to organize the global control structure on a hierachical multilayered basis for software, and on a distributed structure for the hardware.
The goal is to obtain the following caracteristics:

- modularity
- expandibilty
- chance of increase parallelism degree without global changes of the existing structure.

3. The system architecture

The system is provided with an operator site for the handling and supervision of the system, that is on line with the control architecture of the manipulator.
At this level the operator is provided with interactive devices like microphone for voice control, several monitors connected to lower level units and the keyboard.
The control unit is a personal computer provided with a 80386 microprocessor.
From this level the operator can operate the whole system and receive a continuos feedback of the system status.
The processing unit has been also thought as gateway to external operating unit providing other activities that need task execution from the robot cell.
The main unit is connected by a standard serial bus to the manipulator supervisor, a computer unit provided with a 65816 microprocessor.
This lower level unit is oriented to control and handling of third level units, on the basis of tasks requested from the operator site unit.
The third and lowest level is the one that provide the operative units, called MPx, mainly provided with eight bit microprocessor.
The MP1 unit is oriented to real time control of manipulator, data monitoring, handling of manipulator initialization and shutdown.
The MP2 unit is dedicated to the voice control.
It performs voice analysis and commands handling for task execution.
Recently a third unit based on a 16 bit microprocessor and provided with a mathematic coprocessor has been connected to the system, which will be used for on-line computation of kinematics and dinamics. At the present time, these two tasks are demanded to the mainpulator supervisor.
The following step is the integration of a unit for handling of a vision system already experimented on a stand alone unit. The

vision system has been already developed, is operating on a 80386 microprocessor based unit, and is the one described in introduction.

4. The software system

The software utilized on the system is the one written for the lower simple task on single MPx units, and the real time operative system especially developed for the multilayered architecture.
The real time system is modular in its structure, and is provided with all of the services essential to a multitasking system. Any lower level unit is provided with the communication protocols and rules for exchanging data and functions, as the higher control unit is provided with dispatching and priority functions for the handling of subtask executed by the lower units.
At the present time the system is programmed in traditional high level languages like Basic or C. The applications written in such languages can use real time operating system services by way of function call mechanism; the next step will be the developement of a language oriented to handle the system and a new user interface.

5. Conclusions

This work is just the first attempt to subdivide whole control system into subsytems provided with local autonomy, communicating through well-defined protocols, for the optimized and flexible handling of the various subtasks that can be individuated in complex teleoperator operations.
At this moment it is on development the connection to the system of a second manipulator, precisely an IBM SCARA robot.
At the present the system is provided with high level control software for friendly interaction with the operator, for autonomous task planning and operation.
The future activity will consist in development of dedicated language for the system programming and of an interface between a transputer network already installed into a 80386 based computer and the existing architecture.
The aim, besidrs the obvious aspects of modularity and easy-expandibility tipical of an open system, is to map a highly complex system, such as a teleoperator control unit, into a network of specialized subsystems which can be developed and optimized independently and in a transparent way to the whole system.

References

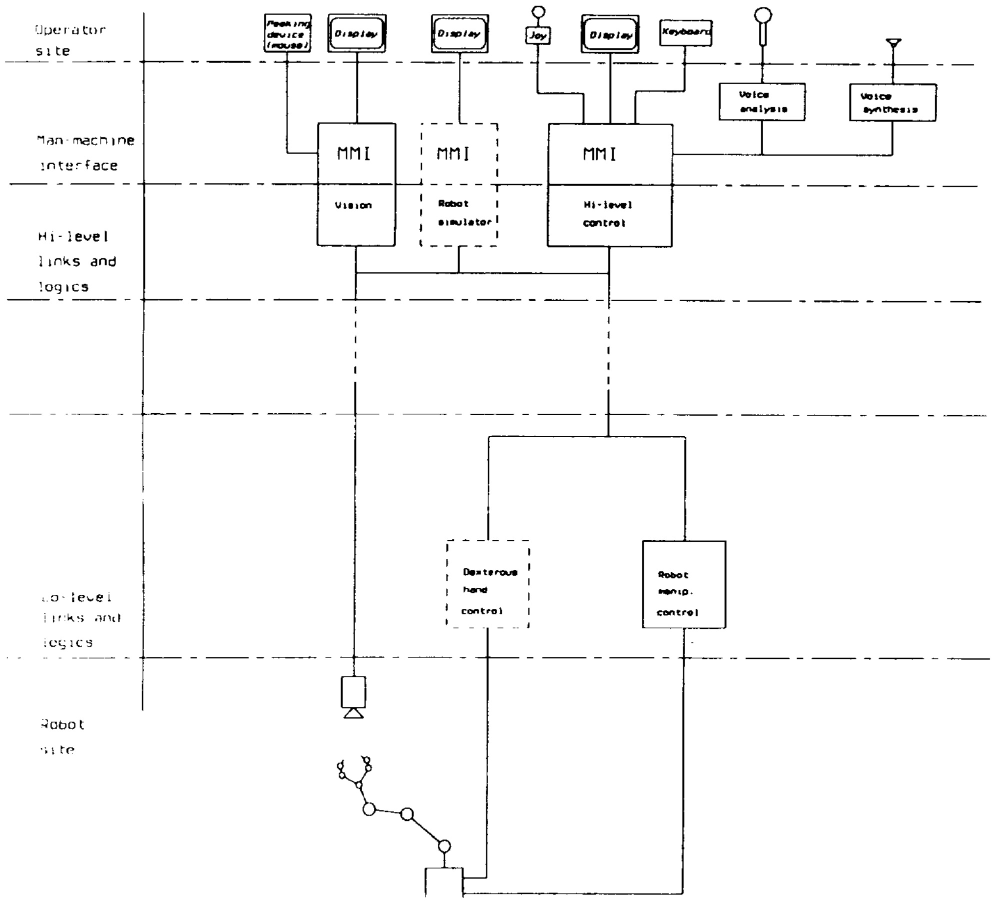
[1] Nato Advanced Research Workshop (ARW), "Robots with Redundancy: Design, sensing and control", Salo´, Italy, 1988

[2] 1988 IEEE International Workshop on Intelligent Robots and Systems (IROS88), Tokyo, 1988

[3] K.G.Shin, S.B.Malin, "A Hierarchical system structure for coordinated control of industrial manipulators", 1985 IEEE International Conference on Robotics, Atlanta, 1985

[4] T.Hagen, A.Coudry, R.W.Yeomans, "Design Rules for a CIM System", Istel, 1987

[5] Dubowsky, Norris, Shiller, "Time optimal robotic manipulator task planning", 6th RoManSy, Kracow, Poland, 1987



SENSOR-BASED PLANNING

N90-29848

How to Push a Block Along a Wall

Matthew T. Mason

Computer Science Department
and Robotics Institute
Carnegie Mellon University

ABSTRACT

Some robot tasks require manipulation of objects that may be touching other fixed objects. The effects of friction and kinematic constraint must be anticipated, and may even be exploited to accomplish the task. This paper analyzes an example task, presents a dynamic analysis, and derives appropriate effector motions. The goal is to move a rectangular block along a wall, so that one side of the block maintains contact with the wall. We construct two solutions that push the block along the wall.

1. Introduction

Consider the problem of pushing a rectangular block along a wall, so that one edge remains in contact with the wall. A few experiments (try pushing a paper-clip box with a paper-clip) will yield two solution strategies, and will also yield a variety of failure modes (Figure 1). This paper derives the two solutions from a dynamic analysis, and shows that there are no other solutions. The approach is to derive the entire mapping from applied force to block motion, and to compare this mapping with the set of all forces that can be applied through a pushing operation. In the process, we demonstrate the analysis of multiple-contact friction dynamics including distributed support friction, using acceleration centers to represent force, as described by Brost and Mason (1989).

1.1. Background

This paper falls in an area that has attracted considerable attention: rigid body mechanics applied to manipulation. The seminal work in this area is Simunovic's (1975) analysis of peg insertion, which was further elaborated by Whitney (1982). Ohwovoriole, Hill, and Roth (1980) provided a more general treatment, which was later extended to three dimensions (Ohwovoriole and Roth 1981). This line of work is mainly quasi-static: inertial forces are assumed negligible, motions are inferred from the direction of any imbalance of static forces. Later work, primarily Erdmann (1984) and Rajan, Burridge, and Schwartz (1987), included dynamic forces and uncovered some interesting subtleties, such as the existence of ambiguities, where the motion of the object may be under-determined. The present paper applies the methods of Erdmann, and Rajan et al., but uses the graphical representation of force described by Brost and Mason (1989).

Related work in the mechanics of grasping is also relevant to the present paper. In particular, we are constructing the locus of contact forces that can be applied on the perimeter of an object, as described by Mishra, Schwartz and Sharir (1986).

An important element in the present paper is the presence of frictional forces that are distributed over a positive area, rather than at known discrete points. Our estimates of the resulting forces draw primarily on (Mason 1986). Better estimates can sometimes be obtained: see (Peshkin and Sanderson 1988; Goyal 1989).

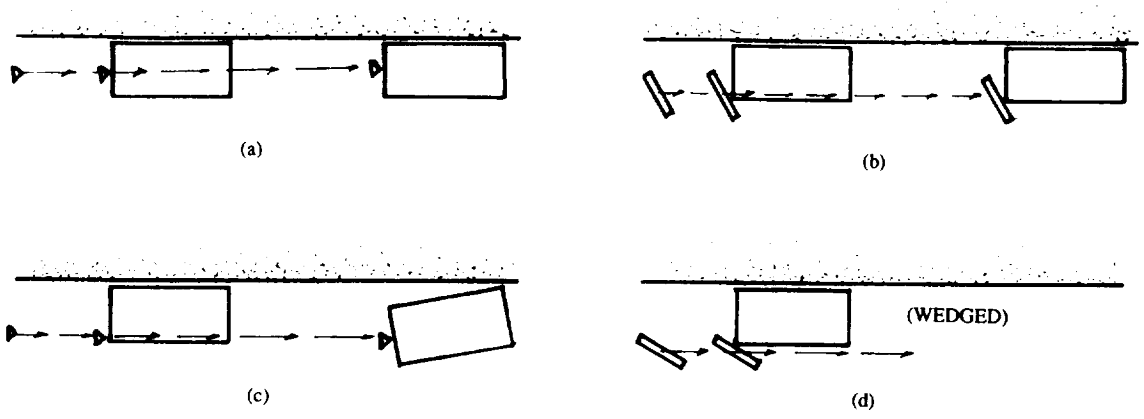


Figure 1: Four different pushing motions. (a) and (b) work just fine. (c) causes the block to lose edge-to-wall contact. (d) is wedged—the block will not move no matter how large the forces.

2. Representation of force by acceleration centers

This section reviews the representation of force by acceleration centers. This approach, as described by Brost and Mason (1989), yields a graphical method to analyze planar contact problems. The construction is similar to the use of velocity centers to analyze kinematic constraints, described by Reuleaux (1876). The key observation is that the velocity of any plane body can be described as a rotation about some motionless point called *velocity center*, or *instantaneous rotation center*. Obviously this would not work for a purely translational motion, but these can be handled by allowing the velocity center to range over the *projective* plane—a translation gives a velocity center at infinity. The velocity center is easily constructed: construct two lines, each orthogonal to the velocity of some point on the body. The intersection of the two lines is the velocity center.

The definition of an acceleration center is similar. For any plane acceleration, there is an unaccelerated point, called the acceleration center, which ranges over the projective plane. We can use the acceleration center to represent forces. Given some plane body with a mass m and angular inertia I, we can map any plane force into the resultant acceleration center. This mapping has a very useful property—the magnitude of the acceleration, and hence the magnitude of the force, is not represented. For problems involving frictional contact, this property is very useful, because the magnitudes of the contact forces are not constrained, only the lines along which the forces act.

In practice the use of acceleration centers to represent applied forces is quite simple. Some examples are shown in Figure 2. We place the center of mass at the origin, and choose a unit distance equal to the radius of gyration. Then the acceleration center lies on a perpendicular to the force through the origin. The acceleration center's distance from the origin is the inverse of the force's distance from the origin. Because it is necessary to represent the sign of the moment of force, we will use two projective planes with a common line at infinity. One plane corresponds to positive moments, one plane corresponds to negative moments, and we have the line at infinity for purely translational accelerations, i.e. zero moments. Topologically, the space is equivalent to a sphere. The upper hemisphere corresponds to positive moments, the lower hemisphere to negative moments, and the equator to zero moments.

The properties and applications of this mapping are more fully described by Brost and Mason (1989). We note two key properties:

- The mapping is nearly dual: a directed line of force maps into a point, and a point, corresponding to the set of all forces passing through that point, maps into a line, the locus of acceleration centers.
- Positive linear combinations of forces map into convex combinations of acceleration centers.

These properties are ideal for representing frictional contacts:

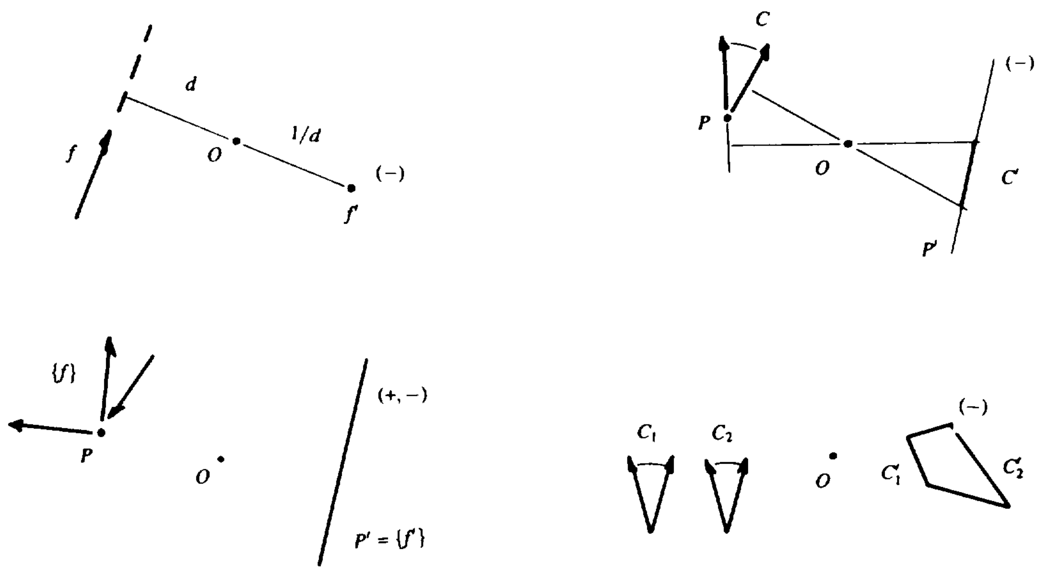


Figure 2: Some example acceleration centers, illustrating properties of the mapping of force to acceleration center. Here we have super-imposed the plane of positive moments and the plane of negative moments, using (+) and (−) to distinguish the points.

- The feasible motions at a point contact are represented by a linear constraint on the feasible acceleration centers.
- The feasible forces at a point contact are represented by a line segment, which is the locus of acceleration centers corresponding to a friction cone.
- The resultant of several friction cones, arising from several simultaneous contacts, lies in the positive linear combination of the forces, which defines a convex polygon in the space of acceleration centers.

Geometrically, the only real inconvenience is that we have two planes and a line at infinity. Sometimes we draw the planes separately; sometimes we superimpose them. When two points must be joined by a line segment, the construction is sometimes counter-intuitive: with the two planes superimposed, draw a line through the two points. Now, if the two points are on the same plane, the line segment is the part of the line between the two points, as usual. But if the two points are on different planes, take the part of the line *outside* the two points, and also include a point at infinity. The method will be illustrated by example.

3. The block-along-wall problem

The block-along-wall problem is formulated as follows.

- A rigid rectangular body is free to move in the plane, with one edge initially against a straight wall.
- We assume Newton's laws with Coulomb friction. Gravity acts normal to the support plane. Friction occurs with the wall and with the support plane. The distribution of support forces is unknown, and may vary with time.
- The goal is to move the object forward while keeping one edge against the wall.

To analyze the operation, we enumerate the contact modes, and determine necessary applied forces for each mode. A peculiarity of rigid body mechanics is that some forces are consistent with more than one mode. Nonetheless, a particular contact mode, i.e. sliding along the wall, can be assured by applying a force consistent with the desired

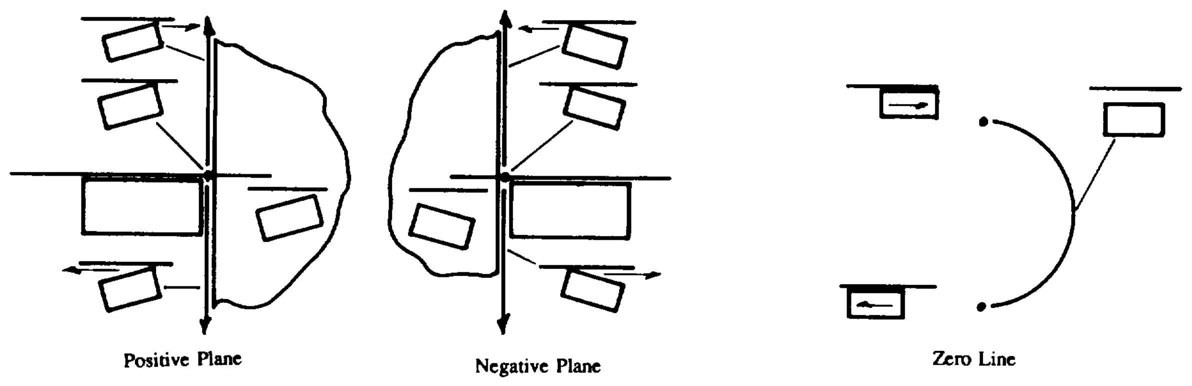


Figure 3: Acceleration centers A for each contact mode. We draw the block tipped to indicate which contacts are broken, and arrows show relative motion at any remaining contacts. The zero line is drawn as if the positive plane and negative plane were projected onto the northern and southern hemisphere, respectively, of a sphere. Then the zero line is the equator, as it would appear from the north pole. There is one contact mode not shown—rest—which corresponds to zero acceleration.

mode and inconsistent with any other modes. This observation allows us to specify constraints on applied force to produce the desired motion.

Our analysis will proceed as follows:

1. Enumerate the contact modes $\{i\}$.
2. For each contact mode,
   (a) Construct acceleration forces A_i,
   (b) Construct wall contact forces W_i,
   (c) Construct support friction forces S_i,
   (d) Construct pushing forces $F_i = A_i \ominus W_i \ominus S_i$.

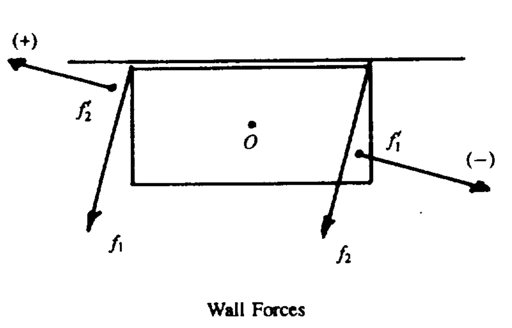
where $X \ominus Y$ is the set of all forces $x - y$, for $x \in X$ and $y \in Y$.

The result is a mapping from each contact mode i to a set of applied forces F_i. Some of these applied force sets overlap, so that the contact mode is not always uniquely determined by the applied force.

The first step is to enumerate the contact modes. Figure 3 applies Reuleaux' (1876) partitioning of the space of motion centers to determine the set of feasible contact modes. At each kinematic constraint, we construct a contact normal. To the right (left) of the normal only positive (negative) rotations are feasible. On the normal itself, either direction is feasible. We also construct the contact tangent—above the tangent positive rotations cause rightward motions and negative rotations cause leftward motions. Below the tangent, the opposite is true. The two contacts give rise to two normals, and a single tangent, which cut the space of acceleration centers into different sectors. Each sector, and each boundary segment, potentially corresponds to a different contact mode.

The next step is to iterate through every contact mode, constructing the corresponding set of applied forces. We illustrate the procedure for only one contact mode, namely the desired mode which pushes the block to the right.

(a) Construct acceleration centers A.
Figure 3 shows the acceleration centers for each contact mode. There is only one acceleration center for the desired mode, rightward sliding, which is on the line at infinity.



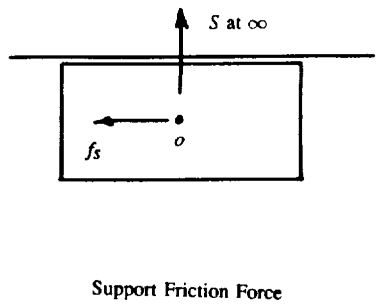


Figure 4: Wall contact forces W, and support frictional forces S for the desired motion.

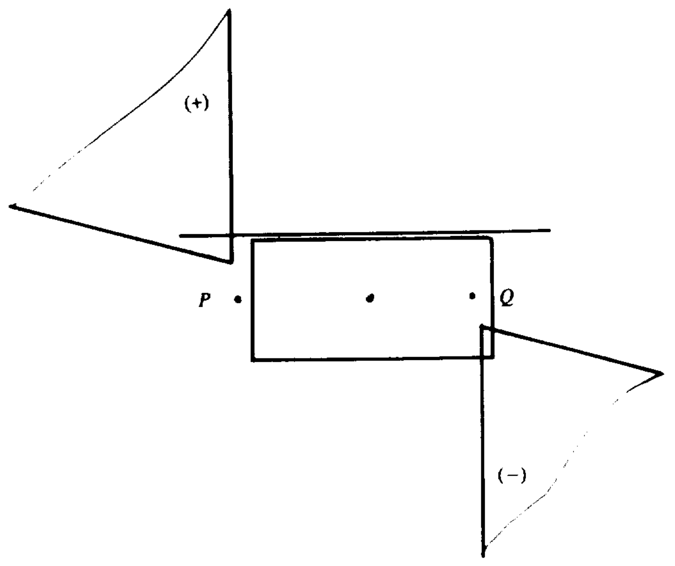


Figure 5: Applied forces F for the desired motion. The applied force must give an acceleration center in the indicated regions. An equivalent constraint is that the applied force must pass between P and Q, and make an angle greater than $\tan^{-1}\mu$ with the wall normal.

(b) Construct wall contact forces W.
The possible wall forces can be represented by two point contacts, one at each corner of the block. For the desired contact mode, rightward sliding, Coulomb's law constrains the direction of each force as shown in Figure 4. We construct acceleration centers for each force, and form the convex combination, to obtain the line segment shown.

(c) Construct support frictional forces S.
For the desired contact mode, rightward sliding, the support frictional force reduces to a single force acting through the center of mass, as shown in Figure 4. This maps to an acceleration center at infinity.

(d) Construct applied forces $F = A \ominus W \ominus S$.
The set of forces $X \ominus Y$ is the positive linear combination of X with $\ominus Y$, so we simply take the convex combination of the positive plane of X with the negative plane Y, and the convex combination of the negative plane of X with the positive plane of Y. When we apply this procedure to compute $A \ominus W \ominus S$, we obtain Figure 5.

3.1. A rotating contact mode

The main complication is in constructing the set of support frictional forces. For the desired contact mode it was easy, because a pure translation was involved. For rotations, the set of support frictional forces is partially indeterminate.

We can, however, construct a set S that represents a bound on the support frictional force. Figure 6 illustrates the method for a contact mode that involves rotation of the block. A bound on S is obtained by applying two constraints:

- A positive (negative) rotation requires a negative (positive) moment with respect to the center of mass. A translation requires zero moment (Mason 1986).
- If the support region lies in some sector with respect to the rotation center, then the force lies in a similar sector, rotated ninety degrees.

These constraints are sufficient to support synthesis of a block-pushing strategy. For other applications, more detailed approximations are required (see Peshkin and Sanderson (1988) for example).

By repeating the procedure for all contact modes, we obtain the complete atlas of Figure 7. This atlas is a multiple-valued mapping from applied force to contact mode. This mapping is an approximation; for some applied forces, some of the predicted motions cannot really occur. But any motion that can occur will be included in the predictions. Hence, where a unique contact mode is predicted, that prediction is a correct one. Note that the desired contact mode, rightward sliding, does not overlap other modes. We can guarantee the desired motion by generating any force in the the region. The problem of generating the required force is the subject of the next section.

4. Synthesizing a pushing motion

When we push the block, an additional frictional contact is applied somewhere on the boundary of the block. The problem is to determine where to push, and in what direction. First we construct the set of all forces that could be generated by pushing the block (Figure 8). Then, by intersecting with Figure 5, we identify two classes of valid strategies (Figure 9). To produce the desired motion, either push with a left-sliding contact along the trailing edge of the block, or use the face of the finger to push at the trailing vertex of the block.

5. Discussion and Summary

The motion center approach is well-suited to problems involving planar frictional contacts. For the support distribution, however, this approach has some limitations. The main limitation is that the motion center approach does not represent magnitudes, and the support friction is bounded in magnitude, unlike constraints in the plane. For example, if we naively construct the set of applied forces that can lead to rest, we obtain the set of all forces. In truth, a *small enough* force in any direction leaves the object at rest, but a large force in the same direction will accelerate the object. For the sliding block problem, we can manage this deficiency, but it represents a general difficulty for which there is no obvious remedy.

A second problem, which is not particular to the motion center approach, is the indeterminacy of the support distribution. The main difficulty in problems of this kind is to find some characterization of the support distribution that leads to a useful characterization of the motion. This paper assumes a known centroid, with the support confined to a known rectangle, which happens not to include feasible rotation centers. I am working to extend the method to more general support distributions.

Acknowledgements

I would like to thank Randy Brost who contributed to the paper in many ways, and Tom Mitchell and Alan Christiansen, who encouraged me to work on the block-along-wall problem.

This work was supported by the National Science Foundation under grant DMC-8520475.

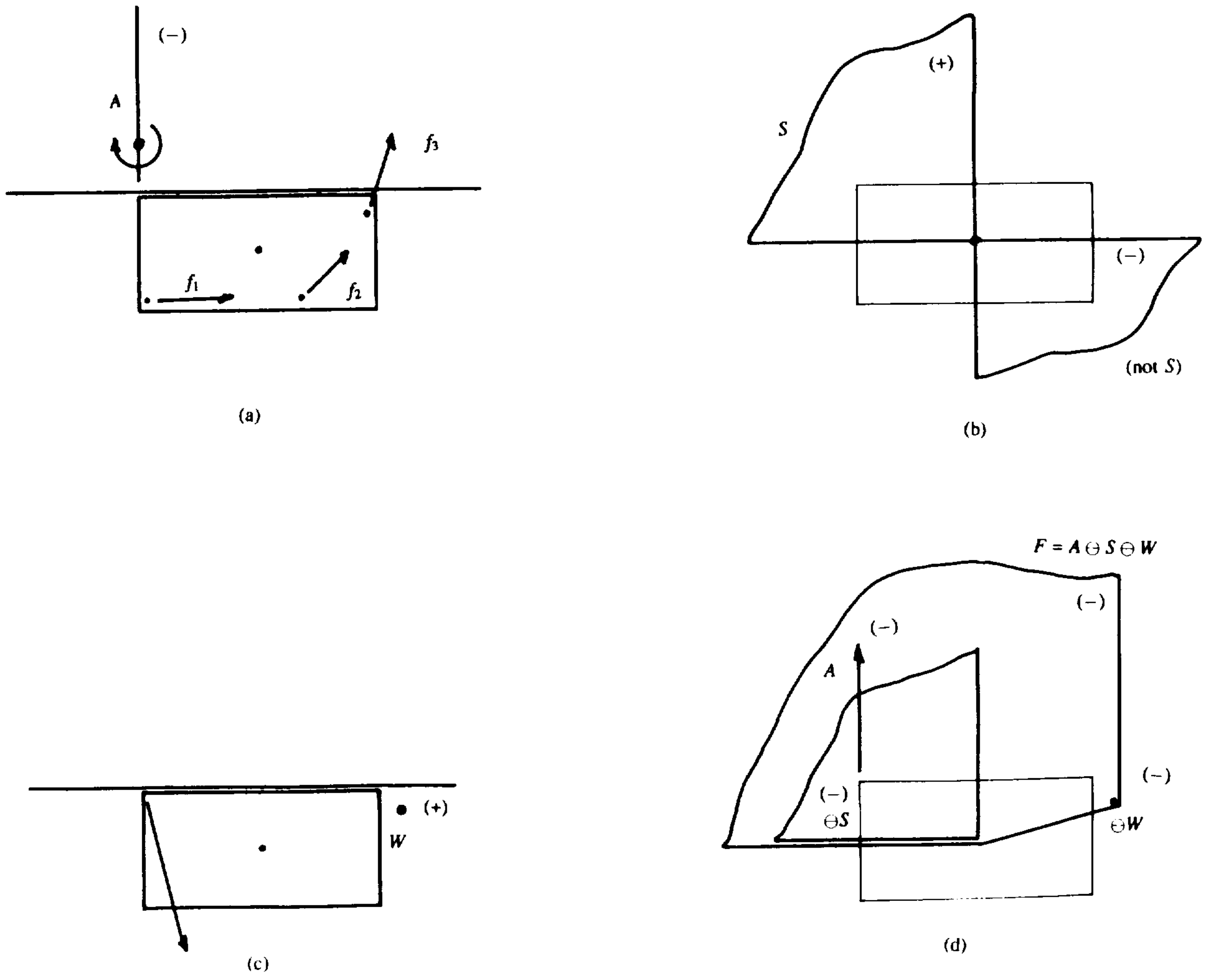


Figure 6: Constructions for a rotating contact mode. The block accelerates to the left, with the right corner losing contact. (a) shows the set of acceleration centers A. Note that because the support distribution is confined to the lower-right quadrant with respect to the acceleration centers, the force direction is confined to the upper-right quadrant. This constraint can be transformed into acceleration center space to obtain figure (b). S corresponds to the (+) half, because of a second constraint: the total force must give a positive moment with respect to the center of mass. (c) shows the wall force W. (d) shows the final set of applied forces F.

References

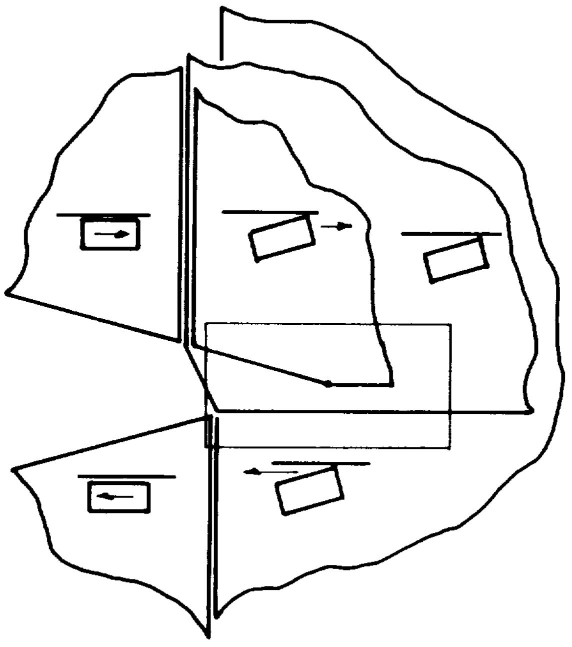
Brost, R. C., and Mason, M. T., "Graphical Analysis of Planar Rigid-Body Dynamics with Multiple Frictional Contacts," submitted to Fifth International Symposium on Robotics Research.

Erdmann, M. A., "On Motion Planning with Uncertainty," MIT Artificial Intelligence Laboratory Technical Report 810, August 1984.

Goyal, S., "Planar Sliding of a Rigid Body with Dry Friction: Limit Surfaces and Dynamics of Motion," Ph.D. thesis, Cornell University, January 1989.

Guibas, L., Ramshaw, L., and Stolfi, J., "A Kinetic Framework for Computational Geometry," *24th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society, November 7–9, 1983.

Mason, M. T., "Mechanics and Planning of Manipulator Pushing Operations," *Int J Robotics Research*, v5 n3, 1986.



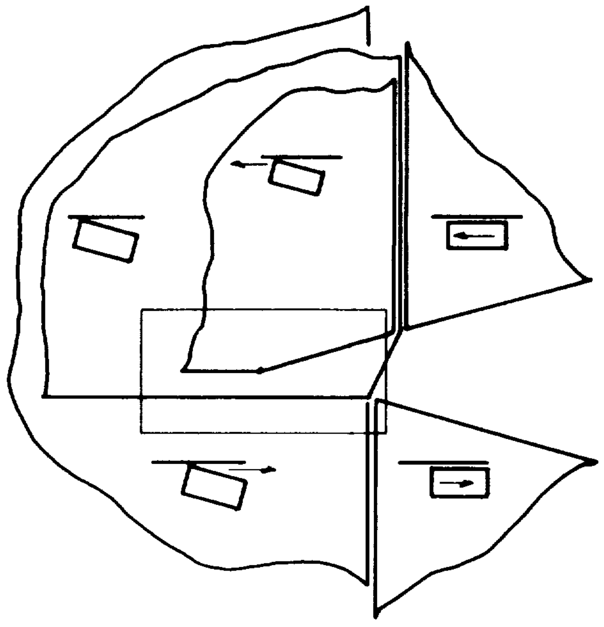


Figure 7: The complete atlas. Not shown is the contact mode that leaves the block motionless, which maps to the entire space of acceleration centers. The gaps in the figure correspond exactly to jamming forces, which fail to move the block no matter how large the force magnitude. Also not shown is the zero line, which in this case is a straightforward continuation of the positive plane.

Mishra, B., Schwartz, J. T., and Sharir, M., "On the Existence and Synthesis of Multifinger Positive Grips," Technical Report No. 259, New York University Courant Institute of Mathematical Sciences, November 1986.

Ohwovoriole, M. S., "An Extension of Screw Theory and Its Application to the Automation of Industrial Assemblies," Stanford Artificial Intelligence Laboratory Memo AIM-338, April 1980.

Ohwovoriole, M. S., Hill, J. W., and Roth, B., "On the Theory of Single and Multiple Insertions in Industrial Assemblies," *Proceedings, 10th International Symposium on Industrial Robots*, pp. 545-558, Milan, Italy, March 1980.

Ohwovoriole, M. S., and Roth, B., "An Extension of Screw Theory," *Trans ASME J of Mech Design, 103*, pp. 725-35, 1981.

Peshkin, M. A., and Sanderson, A. C., "The Motion of a Pushed, Sliding Workpiece," *IEEE J Robotics and Automation*, April 1988.

Rajan, V. T., Burridge, R., and Schwartz, J. T., "Dynamics of a Rigid Body in Frictional Contact with Rigid Walls," *Proceedings, IEEE Conference on Robotics and Automation*, pp. 671-677, Raleigh, North Carolina, March 1987.

Reuleaux, F., *The Kinematics of Machinery*, Macmillan, 1876. Republished by Dover, 1963.

Roth, B., "Screws, Motors, and Wrenches that Cannot Be Bought in a Hardware Store." In Brady, M. and Paul, R. (eds.), *Robotics Research: The First International Symposium*, pp. 679-693, MIT Press, 1984.

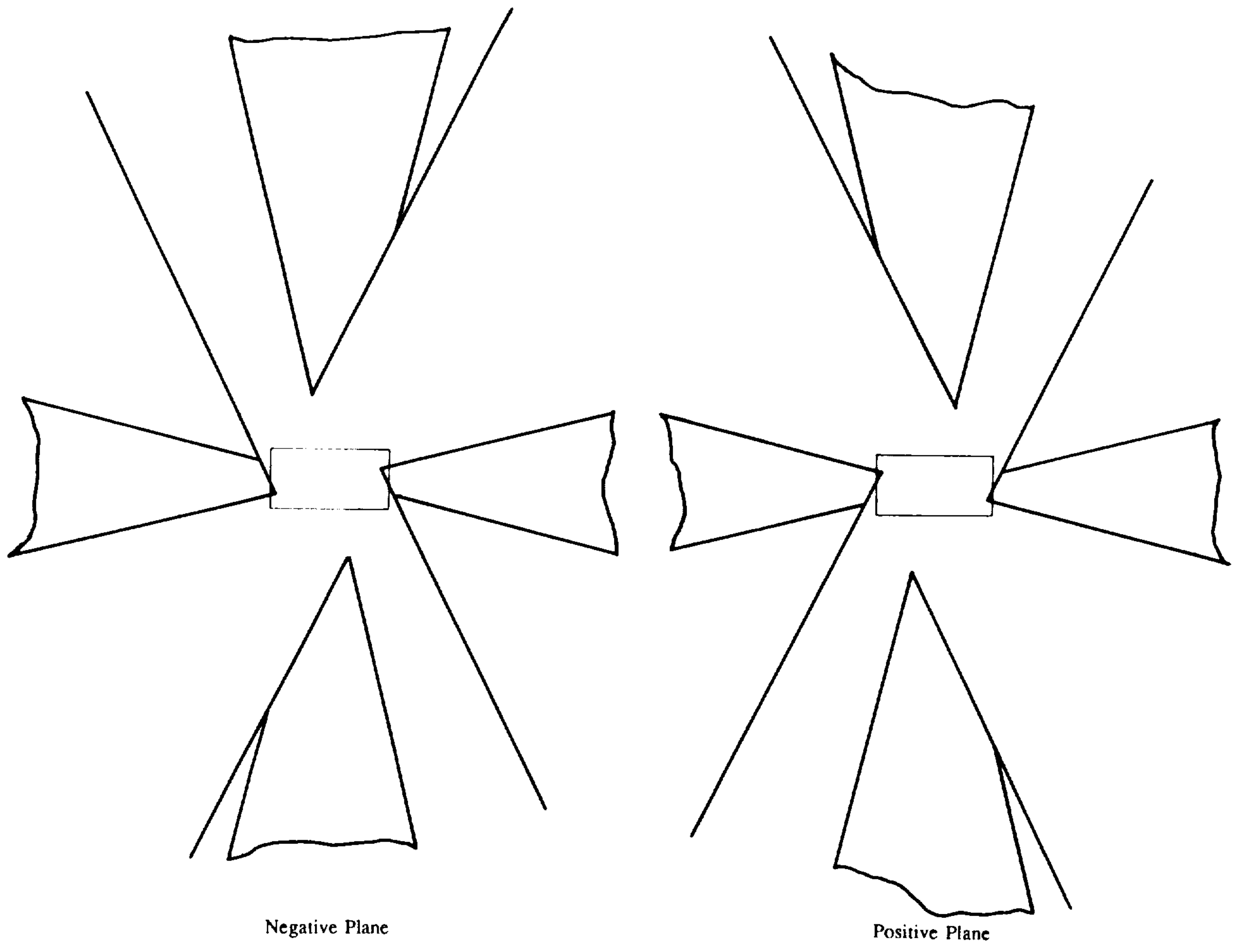


Figure 8: Contact forces arising from a pushing motion.

Simunovic, S., "Force Information in Assembly Processes," *Proceedings, 5th International Symposium on Industrial Robotics*, pp. 415-431, September 1975.

Whitney, D. E., "Quasi-Static Assembly of Compliantly Supported Rigid Parts," *Journal of Dynamic Systems, Measurement, and Control* **104**, pp. 65-77, March 1982.

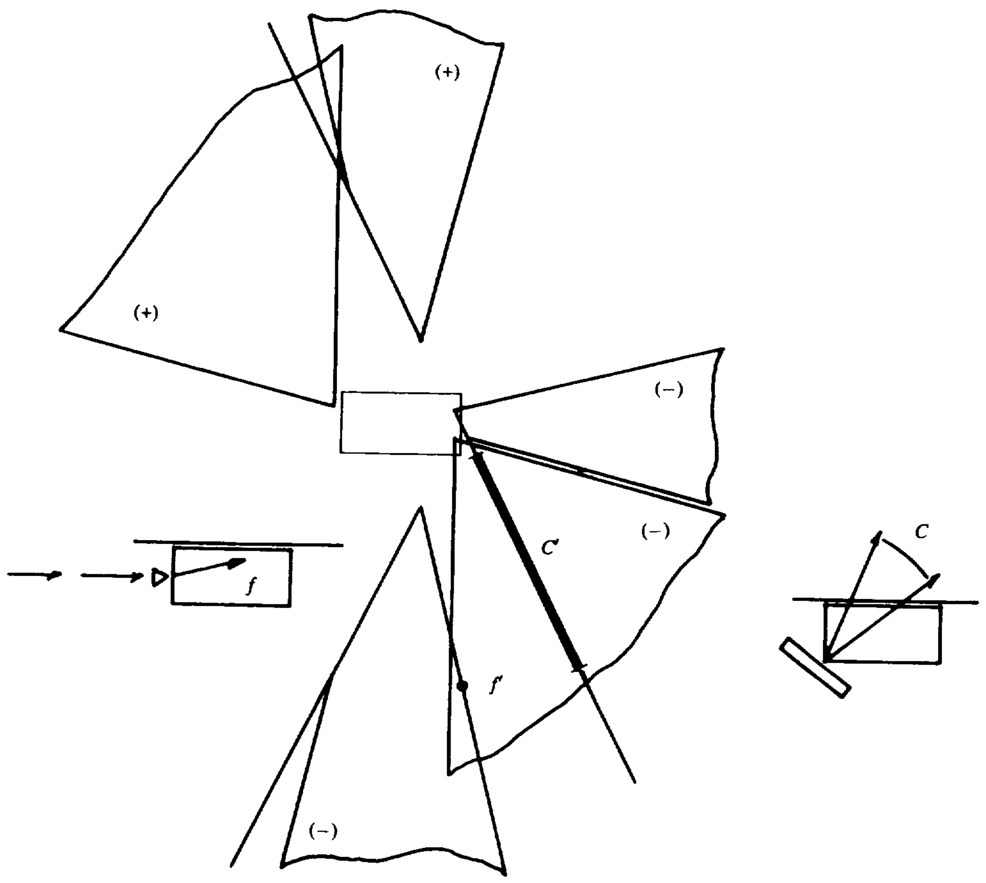


Figure 9: Of the set of possible pushing forces shown in Figure 8, there are just two lobes that intersect the desired forces F of Figure 5. When we intersect the two figures, we obtain two different solutions, illustrated here. f is obtained by pushing on the trailing edge of the block, with a velocity inclined slightly into the wall. C is obtained by pushing on the trailing corner of the block.

N90-29049

GLOBAL MODELS: ROBOT SENSING, CONTROL, AND SENSORY-MOTOR SKILLS

Paul S. Schenker
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive \ MS 23-103C
Pasadena, California 91109

ABSTRACT *

Robotics research has begun to address the modeling and implementation of a wide variety of "unstructured" tasks. Examples include automated navigation, platform servicing, custom fabrication and repair, deployment and recovery, and science exploration. Such tasks are poorly described at onset; the workspace layout is partially unfamiliar, and the task control sequence is only qualitatively characterized. The robot must model the workspace, plan detailed physical actions from qualitative goals, and adapt its instantaneous control regimes to unpredicted events. Developing robust representations and computational approaches for these sensing, planning, and control functions is a major challenge. The underlying domain constraints are very general, and seem to offer little guidance for well-bounded approximation of object shape and motion, manipulation postures and trajectories, and the like. In this paper we discuss this generalized modeling problem, with an emphasis on the role of sensing. We argue that "unstructured" tasks often have, in fact, a high degree of underlying physical symmetry, and such implicit knowledge should be drawn on to model task performance strategies in a methodological fashion. We propose a group-theoretic decomposition of the workspace organization, task goals, and their admissible interactions. This **group-mechanical** approach to task representation helps to clarify the functional interplay of perception and control, in essence, describing what perception is specifically for, versus how it is generically modeled. One also gains insight how perception might logically evolve in response to needs of more complex motor skills. We discuss why, of the many solutions that are often mathematically admissible to a given sensory motor-coordination problem, one may be preferred over others.

* Due to the length of this manuscript, only its abstract and a brief introduction are included within the proceedings. Those wishing a copy of the full paper should request it directly from Dr. Schenker at the above address.

I. INTRODUCTION

Traditionally, robotics applications have been in structured settings. Factory floor robotic assembly is an example -- it is known in advance where objects are located, how they are shaped, a desired sequence for their mating, and desired physical trajectories and forces for their grasp and manipulation. Recent robotics research has taken automation into semi-structured settings, where the robot itself can derive portions of this information during task execution. More flexible and diverse applications can be achieved, with reduced time for task set-up and programming. Supporting developments include CAD/graphical modeling, machine object location and recognition, geometric reasoning, proximity sensing applied to kinematic trajectory correction, contact sensing applied to force-position control adaptation, redundant kinematic design, and grasp dexterity.

Beyond such structured and semi-structured settings, there is a vast range of unstructured robotic tasks. Applications currently under investigation include reconnaissance, navigation, inspection, servicing, repair, recovery, and science exploration, for both terrestrial and space applications [1-2]. Aid-to-the-medically-impaired is another area of great opportunity. Tasks performed in these scenarios are characterized by the uncertain and the unknown. Objects, object motion, and workspace layout may be a priori unspecified; task goals are usually qualitative in nature; kinematic and dynamical control will encounter unmodeled environmental constraints. Thus, successful task performance depends heavily on the robot's ability to organize a physical understanding of its environment, dynamically plan an appropriate sequence of actions, and adapt its sensing and control regimes to the current environmental state [3-6]. Engineered constraints of structured task design expand to natural constraints of the unstructured task environment; requirements for human and machine task performance often begin to look similar [7].

Unstructured tasks present to roboticists, as well as cognitive scientists, a major challenge: identifying and modeling the constraints around which the task will be computationally organized. The following sorts of questions must be answered: what "object" constructs should perception derive and maintain? -- how are they made specific and unique to requirements of a particular task? -- how are they made explicit in a particular set of sensing modalities and configurations?-- how are they accessed and used, in concert with motor control and task constraints, to compute a specific set of motor actions?-- overall, is there hope for a modeling approach in which models for perceiving, planning, and acting can be viewed as a common information structure? Roboticists need answers to these questions, not just from a computational viewpoint, but also from the human performance perspective, e.g.: interactive task planning tools will benefit; telerobotic design will reflect better approaches to shared and traded functional control.

In this paper, I suggest that representations of unstructured perceiving, planning, and acting can be made explicit from a group-theoretic decomposition of a task goal. The basic idea is this: transformation groups and their invariants are defined with respect to underlying symmetries of the workspace, observation space, and kinematic and dynamical constraints of robot-workspace interactions. The admissible group operations define solutions to perception, planning, and control; the associated group invariants, and their underlying metrics, categorically structure the solution space. Of the mathematically admissible solutions, some are rooted in more basic physical symmetries than others, and should be inferred as the more projectively/ dynamically stable, globally probable instantiation of the task. The suggested approach, while currently conceptual, offers potentially practical, important insights for robotics, visual psychology, motor performance, and underlying implementations. As one example, it attempts to formally characterize what perception is for, and how this is manifested in a given task, prior to describing how the individual elements of perception are to be generically modeled, computed, and implemented.

Our paper is non-mathematical and self-contained; here, I concentrate on explaining the group representation concept and its motivation, versus its formal development. In Section 2, I provide an epistemological background and motivation for my approach. In Section 3, I outline the approach, and some past related work. In Section 4, I summarize the main points of my idea and discuss some of its possible implications for further work in robotics and cognitive science.

(Selected references)

1) Schenker, P. S. (1988), NASA research & development for space robotics, IEEE Trans. Aerospace Electr. Sys., vol. 24, no. 5, pp. 523-534 (September).

2) Schenker, P. S. (1989), Intelligent robots for space applications, to appear in Intelligent Robotic Systems: Analysis, Design, and Programming (S. Tzafestas, Ed.), Marcel Dekker, New York City, NY.

3) NSF - National Science Foundation (1987). Report of the Workshop on Multisensor Integration in Manufacturing Automation (T. C. Henderson et al., Eds.), Snowbird, UT, 4-7 February (request Techn. Rept. UUCS-87-006, attn: Prof. T. C. Henderson, Department of Computer Science, Univ. of Utah, Salt Lake City, UT 84112).

(cont'd.)

4) AAAI - American Association of Artificial Intelligence (1987). Proceedings of the Workshop on Spatial Reasoning and Multisensor Fusion (A. C. Kak and S. C. Chen, Eds.), St. Charles, Illinois, 5-7 October(available as ISBN 0-934613-59-1 from Morgan Kaufmann Publishers, 95 First Street, Suite 120, Los Altos, CA 94022).

5) SPIE - The International Society for Optical Engineering (1988). Proc. SPIE Conf. Sensor Fusion I: Spatial Reasoning and Scene Interpretation (P. S. Schenker and C. A. McPherson, Chrs.), Vol. 1003, Cambridge, MA, 7-9 November.

6) NATO - North Atlantic Treaty Organization (1989). Forthcoming "NATO Advanced Research Workshop on Multisensor Fusion for Computer Vision" (J. K. Aggarwal, Org.), Grenoble, France, 26-30 June (contact: Prof. J. K. Aggarwal, The Univ. of Texas at Austin, Computer and Vision Research Center, ENS 519, Austin, TX 78712).

7) SPIE - The International Society for Optical Engineering (1989). Forthcoming SPIE conference "Sensor Fusion II: Human and Machine Strategies" (P. S. Schenker and S. C. Chen, Orgs.), Philadelphia, PA, 5-10 November (contact: Dr. P. S. Schenker, Jet Propulsion Laboratory, 4800 Oak Grove Drive, MS 23-103, Pasadena, CA 91109).

N90-2[illegible]

3-D Vision System Integrated Dexterous Hand

Ren C. Luo and Youn-Sik Han

Robotics and Intelligent Systems Laboratory
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC 27695-7911

1. Introduction

The limited capabilities of conventional two-jaw type grippers have prompted research efforts concentrated on the development of a multifingered hand which can grasp an object of arbitrary shape and manipulate it within its grasp.

Most multifingered hands use a tendon mechanism to minimize the size and weight of the hand. Such tendon mechanisms suffer from the problems of stiction and friction of the tendons. This results in a reduction of control accuracy. In order to overcome these problems in control accuracy and to give the hand more flexibility and intelligence, a design for a 3-D vision system integrated dexterous hand using motor control is presented.

The proposed hand is composed of three three-jointed grasping fingers with tactile sensors on their tips, a two-jointed 'eyed finger' with a microcamera in its distal part, and another two-jointed 'laser-emitting finger' with a cross-shaped laser beam emitting diode in its distal part. The two non-grasping fingers allow 3-D vision capability and can rotate around the hand to see and measure the sides of grasped objects and in its task environment.

Little research effort has been focused on the application of 3-D vision-in-hand systems to perform the task of grasping and manipulating an object. In this paper, an algorithm which determines the range and local orientation of the contact surface using a cross-shaped laser light beam is introduced together with some potential applications.

Grasping and manipulating an object with a multifingered hand is a complicated task and there still remain a number of unsolved problems. One inherent and important problem is the determination of the proper internal grasping forces. Some work has been concentrated on grasping or manipulation force analysis, however, finger force determination has not been addressed. This problem can be solved efficiently using the geometric information of the objects to be grasped acquired by the 3-D vision-in-hand system. In this paper, an efficient method for the finger force calculation is presented which uses the measured contact surface normals of an object.

Current industrial manipulators are usually equipped with two-jaw grippers. These grippers not only have difficulty in handling objects with an arbitrarily complex geometry, but also have difficulty in manipulating objects within their grippers.

An alternative solution to this problem is to design and build a flexible gripping device which is capable of a large variety of tasks. Several investigators have developed designs for such devices. A number of multi-fingered multi-jointed hands have been developed, ranging from the nine degree of freedom (DOF) Stanford/JPL hand to the 16

DOF Utah/MIT hand. Most of these multi-fingered hands are driven through a cable transmission to minimize the size and weight of the hand and to give compliance to the finger at the expense of problems such as stiction and friction, control stability, time response, and accuracy. To compensate for the adverse effects of a cable transmission mechanism, it is necessary to develop relatively complex low-level control systems that include cable tension control. These hand and finger configurations tend to mimic the human hand, which is not necessarily the best configuration for a mechanical hand.

In this paper, we have developed a more compact design for a dexterous hand to overcome the difficulties resulting from the use of a cable transmission. Furthermore, we are currently developing a hand-based 3-D laser range finder which can rotate around the hand so that it can perceive the object being grasped.

2. Hand Design

Recently, as a result of an increased research effort in the field of multifingered hands, several hands have been developed. Almost all of the multifingered hands developed are human-like, but this does not seem to be the optimal solution to the problem of grasping objects of various shape and manipulating them for various tasks--even though human-like hands have some advantages when imitating the function of haman hand, especially in a teleoperation system.

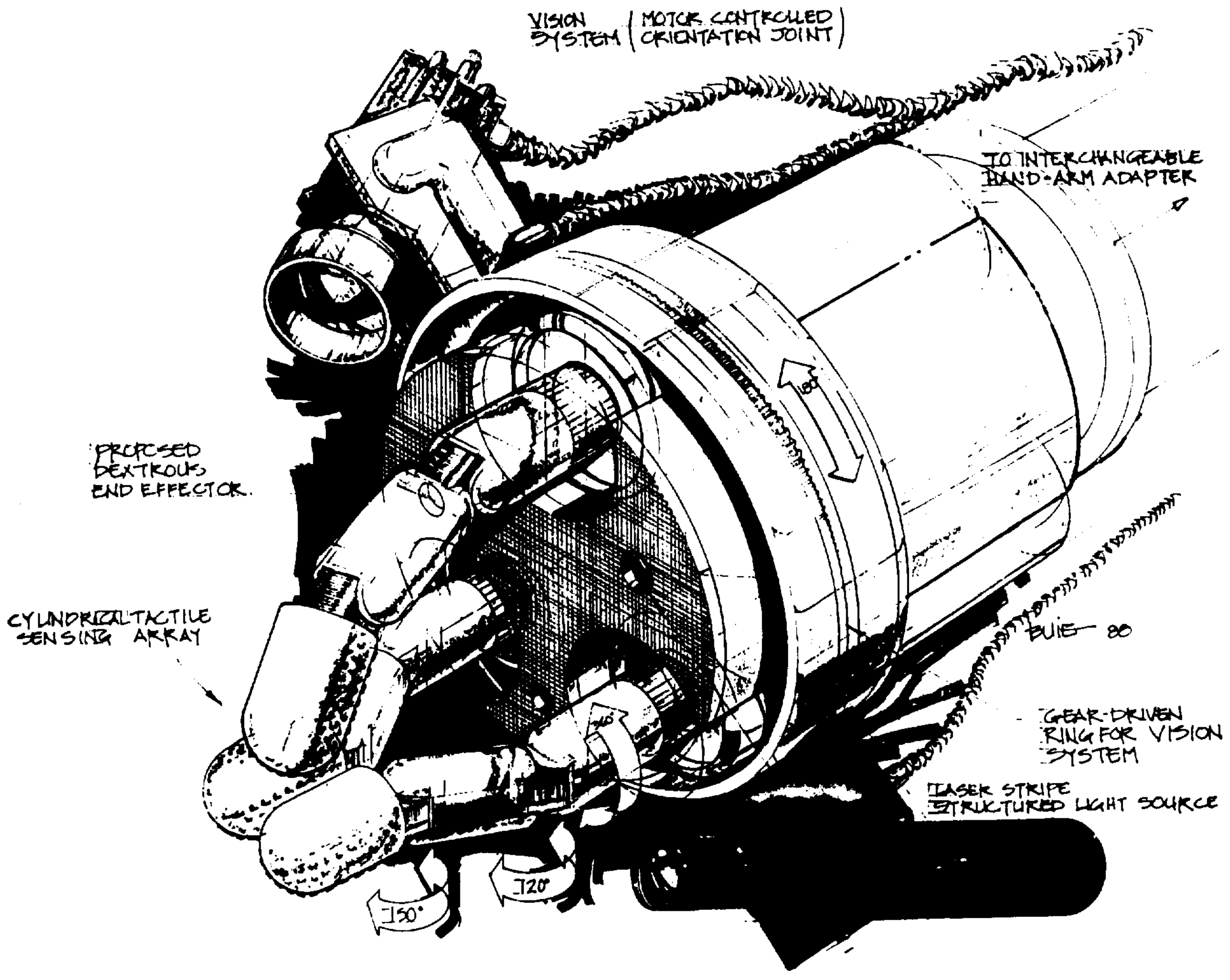
Most of the multifingered hands that have been developed have a relatively small hand workspace [Kerr et al 86] due to their human-like fingers and the finger joint arrangement. The first and middle fingers are placed near to each other, limiting the space to cooperate with each other. In addition, the first joint of each finger is a yawing joint, not a twisting joint. If we allow some distance between fingers and make the first joint of each finger a twisting joint, then the hand workspace will be enlarged and have more dexterity.

Furthermore, most of these multifingered hands have adopted a cable transmission system and their actuators are located remotely. This makes the design of the control system very difficult because of the deflection, friction, and stiction of the cables. Thus it is hard to control the fingertip position and force precisely during manipulation. Therefore, it is better to design the transmission system so that it is rigid and has a shorter transmission length by placing the motors near to the fingers.

For the reasons given above, we have developed a new design for a five-fingered hand where one of the fingers is eyed and one emits a laser beam. The remaining three fingers are used for grasping. The nongrasping fingers can rotate around the hand as shown in the illustrative scheme (see Fig.1).

It is necessary for a hand to have a minimum of three fingers in order to grasp an object of arbitrary shape. Thus, the proposed hand has three grasping fingers. In order to position the fingertip everywhere within the workspace, it is necessary for each finger to have minimum of three degrees of freedom. Thus, each finger has been designed for having three joints. The kinematic arrangement of finger joints are unlike the human hand. Three fingers are equally spaced. The first joint of all three grasping fingers is a revoluting joint which can provide a larger hand workspace and more dexterity.

Each of the nongrasping fingers is composed of two joints so that it can possibly reorient itself in the direction of the object to be grasped in the adjacent task environment. A micro-camera is located in the distal part of eyed finger and a laser diode is implanted in the laser beam emitting finger.



VISION SYSTEM
MOTOR CONTROLLED ORIENTATION JOINT
TO INTERCHANGEABLE HAND-ARM ADAPTER
PROPOSED DEXTROUS END EFFECTOR.
CYLINDRICAL TACTILE SENSING ARRAY
GEAR-DRIVEN RING FOR VISION SYSTEM
LASER STRIPE STRUCTURED LIGHT SOURCE
120°
150°

Figure 1. Proposed illustrative scheme for a 3-D vision integrated dexterous hand

Each joint is driven by a micro dc servomotor with a reduction gearhead through the gear and/or chain transmission. This provides for more rigidity in the control system as compared to the remote cable transmission.

3. Grasping Algorithm using 3-D Visual Data

Manipulating an object with a multifingered hand is a non-trivial task. Not only are the kinematic relationships between the finger joint motions and the object motion complicated, but during the execution of the task the hand must also firmly grasp the object and exert some force and moments to the task environment.

Salisbury has investigated grasping or manipulation force stability [Salisbury 82]. Since the object is overconstrained by the multifinger contacts, the system is statically indeterminate. Several other researchers have determined the optimal internal forces. Kerr has determined that the optimal internal forces are those with minimum norm under an approximated frictional constraints [Kerr 86]. Hollerbach has developed a fast algorithm to calculate the fingertip forces without using the Jacobian for the given external and internal forces [Hollerbach et al 86]. Yoshikawa defined the concept of internal forces as grasping forces [Yoshikawa et al 87]. Here we develop an approach to determine the internal grasping forces and finger forces based on the contact surface information that can be measured by the proposed 3-D laser range finding devices.

3.1 Determination of the Internal Grasping Force

Each fingertip force, $\mathbf{f}_i$, can be decomposed into two components: the internal grasping force, $\mathbf{f}_{gi}$, and the manipulating forces, $\mathbf{f}_{mi}$, where subscript i denotes the finger (i = 1,2,3). The internal grasping force components are defined as these forces that contribute to the grasping of a massless rigid object, and the manipulating force components are defined as those forces that contribute to generate the resultant net forces used to move an object. The main feature of the internal grasping forces is that the forces are equilibrating by themselves and thus do not have any effect on the resultant forces and moments. If we can determine these internal grasping forces, then the manipulating forces and each fingertip force can easily be determined.

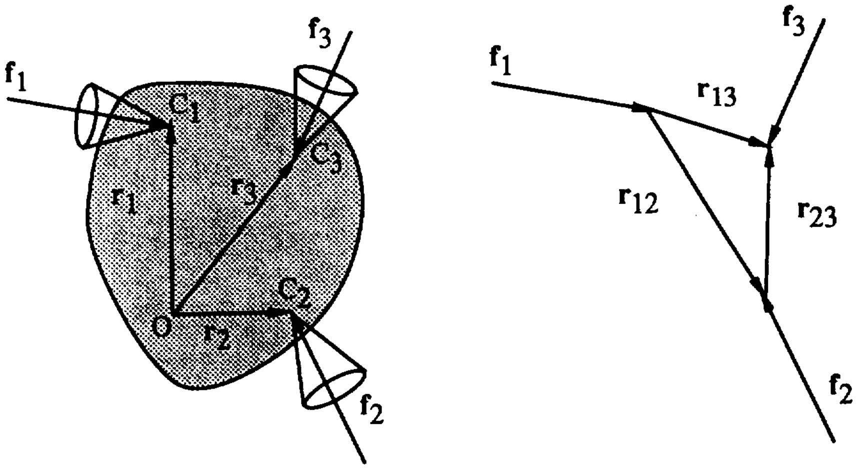


Figure 2. Illustration of three points contact with friction on an object

There are an infinite number of solutions for the internal grasping forces because the system is statically indeterminate. This arbitrariness has three degrees of freedom when we use a hard finger (friction point contact) model in three finger grasping. Howev er, by considering the friction coefficient between the contact points and the contact surface, we can reduce the arbitrariness of the solutions to the degree of one. Also, considering each total fingertip force as being limited by the maximum exertable finger joint driving torque, we can finally determine the internal grasping forces. Since the internal grasping force components of the fingertip forces (the $\mathbf{f}_{gi}$'s) are equilibrating by themselves and do not contribute to the resultant force, we can formulate the equations as follows.

$$\mathbf{f}_{g1} + \mathbf{f}_{g2} + \mathbf{f}_{g3} = \mathbf{0}$$
$$\mathbf{f}_{g1}\cdot\mathbf{r}_1 + \mathbf{f}_{g2}\cdot\mathbf{r}_2 + \mathbf{f}_{g3}\cdot\mathbf{r}_3 = 0 \qquad (1)$$

where

$\mathbf{r}_i$: position vector of the ith contact point

This means that the $\mathbf{f}_{gi}$'s form a closed triangle on the grasping plane, where the contact points (the C_i's) lie on the plane and the lines through the C_i's are intersecting at one point, G (see Figure 3). Let G be called the grasp center point.

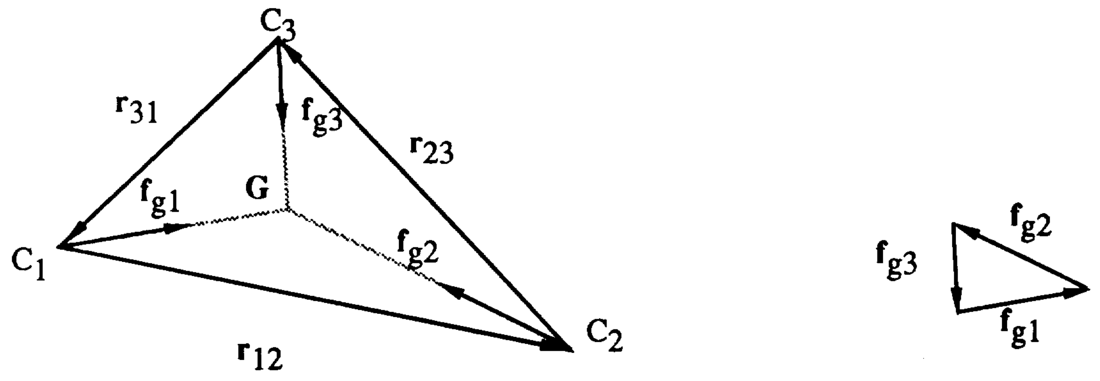


Figure 3. Internal grasping forces and grasp centerpoint

The grasp center point should be inside the common intersection volume of the three friction cones whose center axes are contacting the surface normal vectors, $\mathbf{a}_i$. Also, G should lie on the grasp plane (see Figure 4).

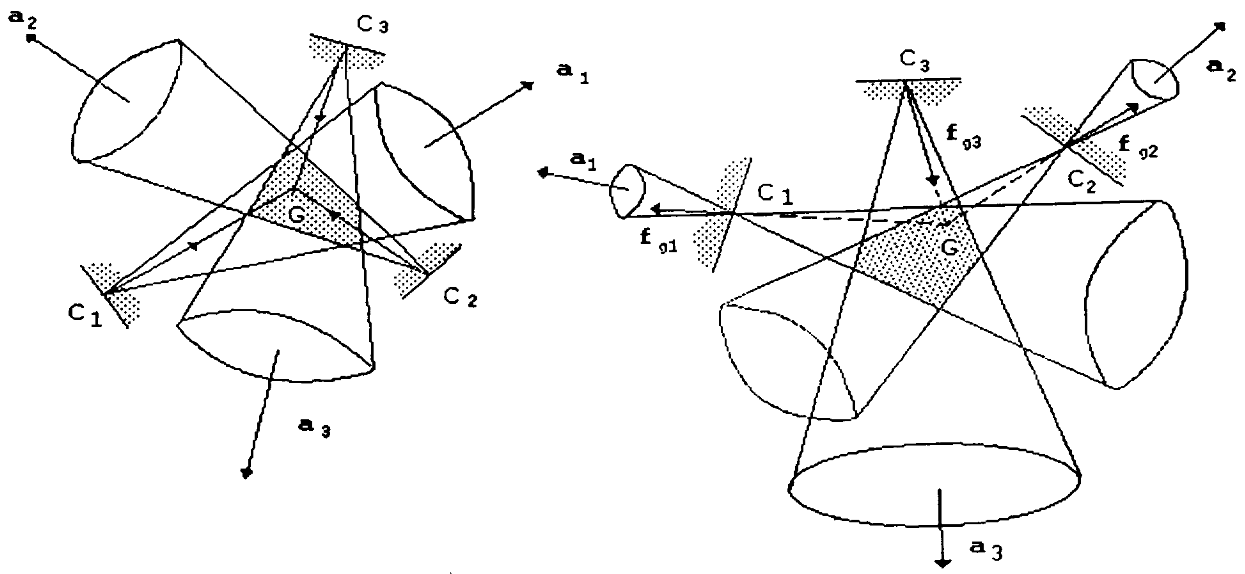


Figure 4. Friction cones and contact normal vectors

Therefore, if we can properly choose the grasp center point, G, for given contact surface normal vectors $\mathbf{a}_i$, we can reduce the three degrees of arbitrariness to one by using the equilibrium equations of the internal grasping forces (see Figure 5).

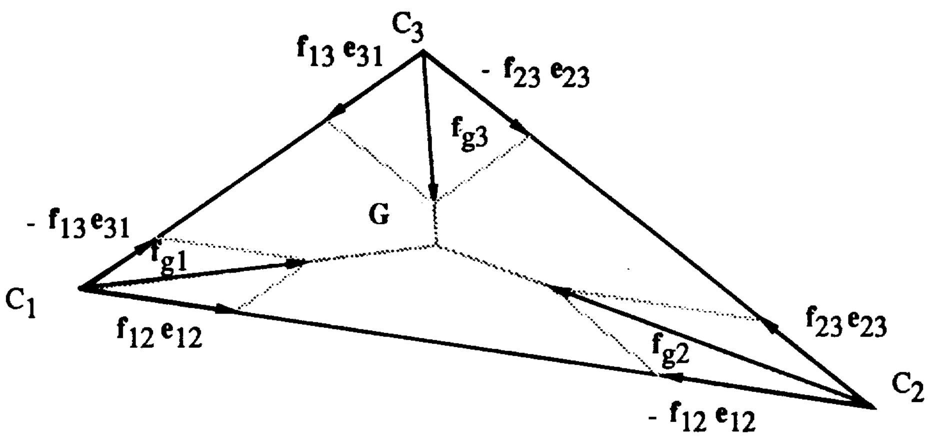


Fig 5. Decomposition of internal grasping force components

Let $\mathbf{a}_i{}^*$ be the projection of the contact surface normal vector $\mathbf{a}_i$ onto the grasp plane. Then, in general, the prolonged lines of the $\mathbf{a}_i{}^*$ form a triangle on the grasp plane. We can determine the grasp center point G to be the center point of the inward tangent circle of this triangle so that the sum of deviations from each line will be minimized (see Figure 6).

Then, the internal grasping forces $\mathbf{f}_{gi}$ act along the unit vector

$$\mathbf{e}_{iG} = (\mathbf{r}_G - \mathbf{r}_i) / \| \mathbf{r}_G - \mathbf{r}_i \| \tag{2}$$

where $\mathbf{r}_G$ is the position vector of the grasp center point G.

Since the internal grasping forces are located inside of the friction cones, the constraints of equation (3)

$$| \mathbf{a}_i \cdot \mathbf{e}_{iG} | \leq \cos(\tan^{-1} u_i) \tag{3}$$

should be satistfied. (Here u_i is the friction coefficient of each finger.) Otherwise, the object cannot be grasped with these contacts.

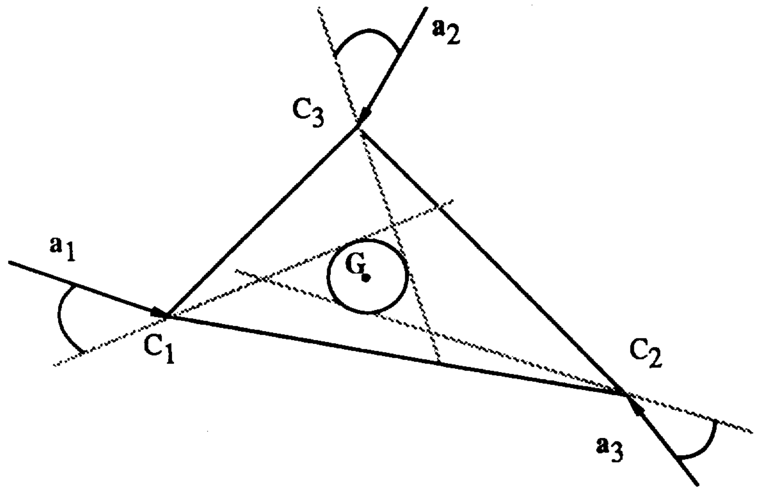


Figure 6. Determination of the grasp center point G

Once we have determined the grasp center point G, we can easily determine the internal grasping force components from the equilibrium equations.

The internal grasping force $\mathbf{f}_{gi}$ can be decomposed into two components which are parallel to the edges of the grasp triangle. Let

$$\mathbf{e}_{ij} = \mathbf{r}_{ij} / \| \mathbf{r}_{ij} \|$$
$$\mathbf{r}_{ij} = \mathbf{r}_j - \mathbf{r}_i$$

Then,

$$\begin{aligned} \mathbf{f}_{g1} &= (\mathbf{f}_{g1} \cdot \mathbf{e}_{31})\, \mathbf{e}_{31} + (\mathbf{f}_{g1} \cdot \mathbf{e}_{12})\, \mathbf{e}_{12} \\ \mathbf{f}_{g2} &= (\mathbf{f}_{g2} \cdot \mathbf{e}_{12})\, \mathbf{e}_{12} + (\mathbf{f}_{g2} \cdot \mathbf{e}_{23})\, \mathbf{e}_{23} \\ \mathbf{f}_{g3} &= (\mathbf{f}_{g3} \cdot \mathbf{e}_{23})\, \mathbf{e}_{23} + (\mathbf{f}_{g3} \cdot \mathbf{e}_{31})\, \mathbf{e}_{31} \end{aligned} \tag{4}$$

3.2 Determination of Fingertip Force $\mathbf{f}_i$

As mentioned in previous section, we can determine the internal grasping force $\mathbf{f}_{gi}$ and its component magnitude f_{ij}. The relationship between the fingertip forces, $\mathbf{f}_i$, and the internal grasping force components can be expressed as

$$f_{ij} = (\mathbf{f}_i \cdot \mathbf{e}_{ij} - \mathbf{f}_j \cdot \mathbf{e}_{ij}) = (\mathbf{f}_i - \mathbf{f}_j) \cdot \mathbf{e}_{ij}$$

This means that the internal grasping force component magnitude f_{ij} is the difference in the projections of the fingertip forces $\mathbf{f}_i$ and $\mathbf{f}_j$ to the intercontact line $\mathbf{r}_{ij}$. The above relationship can be rewritten as

$$(\mathbf{f}_i - \mathbf{f}_j) \cdot \mathbf{r}_{ij} = f_{ij} \cdot \| \mathbf{r}_{ij} \|$$

Then, by applying Hollerbach's method [Hollerbach et al 86], we can determine the fingertip forces (the $\mathbf{f}_i$s). These fingertip forces also should lie inside of the friction cone

$$\mathbf{f}_i / \| \mathbf{f}_i \| \cdot \mathbf{a}_i \leq \cos (\tan^{-1} u_i)$$

where u_i is the friction coefficient between the fingertip and the contact surface. If the fingertip force does not meet this criterion, it can not be applied by the hard finger contact, and thus we should find another place to locate the fingertip.

3.3 Determination of Finger Joint Torques

Taking advantage of the kinematic features of the fingers of the proposed hand, each joint torque t_{ik} can easily be expressed as a function of the grasp intensity g from the fingertip force $\mathbf{f}_i$, where subscript k denotes each finger joint number. Since t_{ik} should not exceed the limit of maximum torque t_{ikmax}, we can determine the value of grasp intensity g.

Since the internal grasping force does not contribute to the resultant force and moment, adjusting the value of grasp intensity (or grasp security) g can result in more flexibility in the control of the fingertip forces.

4. 3-D Vision Integrated with the Hand for Grasp planning and Manipulation

As previously mentioned in the determination of finger force, the role of 3-D vision in giving information about contact locations (contact point vectors $\mathbf{r}_i$ and contact surface normal vectors $\mathbf{a}_i$) is very important in the grasping planning stage. Also, 3-D vision can play a major role during the reorientation of a grasped object during task execution.

The proposed 3-D vision system is composed of one small camera and one cross-shaped laser beam emitter, each of which is mounted on the distal part of two seperate fingers. One finger is called the eyed finger and the other is called the laser beam emitting finger. Each of these two fingers has one bending joint and is mounted on rotating gears around the hand. Thus the camera will be able to see any side of the grasped object, and can measure the distance to any point on surface of the object to be grasped and the surface orientation of the object.

After grasping an object, the status of the grasped object may not be the same as expected because of measurement and control error. Since the main control scheme of the multifingered hand is force control, object motion is subject to the interaction between the fingertip forces. Furthermore, there are many uncertainties in the contact mechanics and the grasped object may not be the expected state. With the proposed 3-D vision system, we can measure the status of the grasped object, namely, its orientation and the position of a particular part of the object with respect to the hand frame. Figure 7 illustrates this basic concept.

We have simulated the operation of this sensor based dexterous hand using an Appolo graphics workstation. Figure 8 shows an example of the simulation.

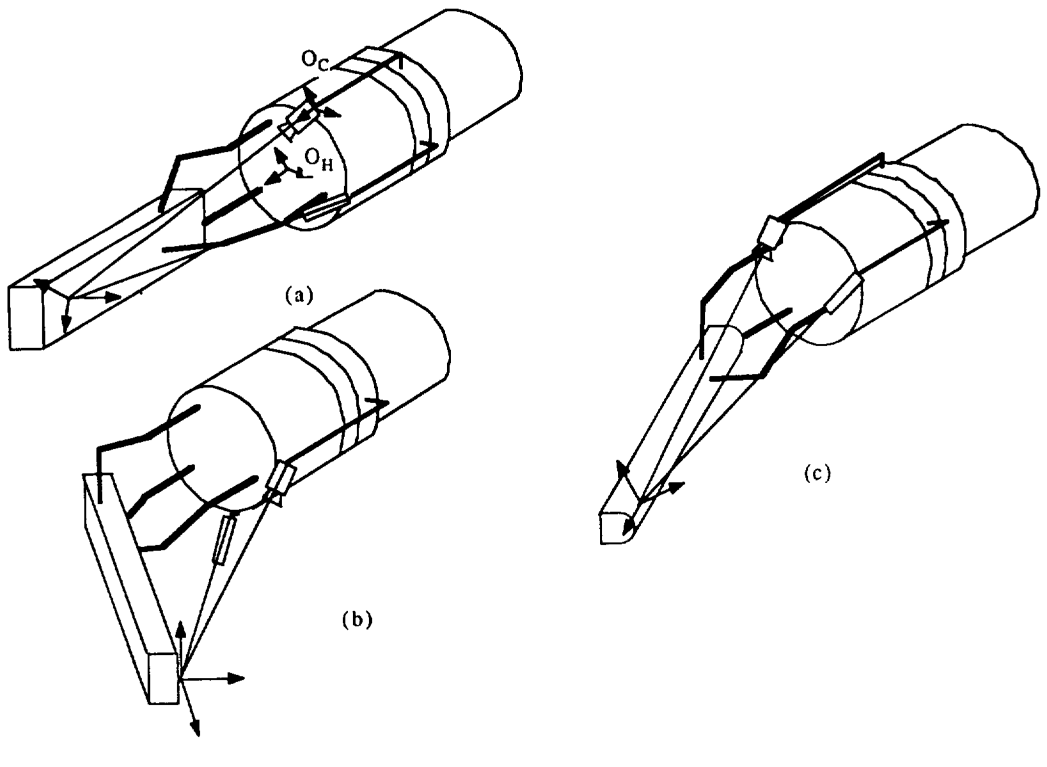


Figure 7. Object posture measurements

Figure 8. Two views of simulated design of sensor based dexterous hand

REFERENCES

Biggers, K.B., Jacobsen, S.C., Gerpheide, G.E. (1986). Low Level Control of the Utah/MIT Hand. 1986 IEEE International Conference on Robotics and Automation, pp. 61-66.

Hollerbach, J.M., Narashman, S., Wood, J.E. (1986). Finger Force Computation without Grip Jacobian, 1986 IEEE International Conference on Robotics and Automation, pp. 871-875.

Jacobsen, S.C., Iversen, E.K., Biggers, K.B., Knutti, D.F., Johnson, R.T. (1986). Design of the Utah/MIT Dexterous Hand. 1986 IEEE International Conference on Robotics and Automation, pp.1520-1532.

Jameson, J.W., Leifer, L.J. (1986). Quasi-static Analysis : A Method for Predicting Grasp Stability. 1986 IEEE International Conference on Robotics and Automation, pp. 876-883.

Kerr, J., Roth, B., (1986). Analysis of Multifingered Hand. International Journal of Robtics Research, vol. 4, no. 4, pp. 3-17.

Narashiman, S., Siegel, D.M., Hollerbach, J.M. (1988). CONDOR : A Revised Architecture for Controlling Utah/MIT Hand. 1988 IEEE International Conference on Robotics and Automation, pp. 446-449.

Salisbury, J.K. (1982). Kinematic and Force Analysis of Articulated Hands. Ph.D. Dissertation, Department of Mechanical Engineering, Stanford University, 1982.

Salisbury, J.K., Craig, J.J. (1982). Articulated Hands : Force Control and Kinematic Issues. International Journal of Robtics Research, vol. 1, no. 1.

Yoshikawa, T., Nagai, K. (1987). Manipulating and Grasping Forces in Manipulation by Multifingered Hands. 1987 IEEE International Conference on Robotics and Automation, pp. 1998-2004.

A Layered Abduction Model of Perception: Integrating Bottom-up and Top-down Processing in a Multi-Sense Agent

John R. Josephson
Laboratory for Artificial Intelligence Research
Ohio State University
Columbus, Ohio 43210

1 February 1989

Abstract

This paper introduces a layered-abduction model of perception which unifies bottom-up and top-down processing in a single logical and information-processing framework. The process of interpreting the input from each sense is broken down into discrete layers of interpretation, where at each layer a "best explanation" hypothesis is formed of the data presented by the layer or layers below, with the help of information available laterally and from above. The formation of this hypothesis is treated as a problem of abductive inference, similar to diagnosis and theory formation. Thus this model brings a knowledge-based problem-solving approach to the analysis of perception, treating perception as a kind of "compiled" cognition.

The bottom-up passing of information from layer to layer defines channels of information flow, which separate and converge in a specific way for any specific sense modality. Multi-modal perception occurs where channels converge from more than one sense.

This model has not yet been implemented, though it is based on systems which have been successful in medical and mechanical diagnosis and medical test interpretation.

Introduction

Computational models of information processing for both vision and spoken language recognition have commonly supposed an orderly progression of layers, beginning near the retina or auditory periphery, where hypotheses are formed about "low-level" features, e.g., edges (in vision) or bursts (in speech perception), and proceeding by stages to higher-level hypotheses. These higher-level hypotheses typically depend largely on hypotheses formed at lower levels, but are also subject to influence from above.

Models intended to be comprehensive often suppose 3 or more major layers, often with sublayers, and sometimes with parallel channels which separate and combine to support higher-layer hypotheses (e.g., shading discontinuities and color contrasts separately supporting hypotheses about object boundary) [31, 28, 29]. Audition, Phonetics, grammar, and semantics have been proposed as layers of interpretation for speech comprehension. Recent work on primate vision appears to show the existence of separate channels for information about shading, texture, and color, not all supplying information to the same layers of interpretation [29].

In both vision and speech understanding most of the processing of information is presumably bottom-up, from information produced by the sensory organ, through intermediate representations, to the abstract cognitive categories that are required for reasoning. Yet top-down processing is significant, as higher-level information is brought to bear to help with identification and disambiguation. Both vision and speech recognition can thus be thought of as "layered interpretation" tasks whereby the output from one layer becomes data to be "interpreted" at the next. Layered interpretation models for non-perceptual interpretive process make sense too, for example medical diagnosis can be thought of as an inference which proceeds from symptoms, to pathophysiological states, to diseases, to etiologies. It is reasonable to expect that perceptual processes have been optimized over evolutionary time, and that the specific layers and hypotheses, especially at lower levels, have been compiled into special-purpose mechanisms. Perceptual learning provides another source of compilation and optimization. Nevertheless, these layered interpretation models all seem to share certain functional similarities.

In particular, it appears that at each layer of interpretation the information processing task is the same: that of forming a coherent, composite (multi-part) "best explanation" of the data from the previous layer. That is, the task is one of performing an *inference to the best explanation*, in other words, an *abductive inference*.

Moreover, it appears that similar types of hypotheses-hypothesis interactions appear in vision, speech understanding, and diagnosis. Here are three important ones:

1. Two hypotheses might partially overlap in what they can account for, but otherwise be compatible (e.g. an edge might be a boundary for two different objects, the /s/ sound acoustically in the middle of "six stones" belongs to both words, the high white blood count is a result of two different infections),

2. Hypotheses might be pair-wise incompatible (e.g. patch X is either part of the figure or part of the background),

3. Hypotheses might be supportive in an associative way, the presence of one giving some evidence for the presence of the other. (Associative support presumably represents the net impact of several distinct types of evidential relationships.)

The functional similarities suggests the posibility of a generic mechanism, and just such a generic mechanism is proposed here. It is hypothesized that that the processing that occurs in vision, hearing, understanding spoken language, and in interpreting information from other senses (natural and robotic) can all be usefully thought of as variations, incomplete realizations, or compilations (domain-specific optimizations) of this one basic computational mechanism, which we may call *the layered abduction model of perception.*

There is a long tradition of belief that perception involves some form of inference [27] [17] [2]. Several researchers have in fact proposed that perception, or at least language understanding, involves some form of abduction or best-explanation inference [10, p.557] [9] [11] [37] [21, pp.87-94] [14, pp.88,104]. Abduction is often thought of as being logically similar to theory formation in science [17] [46] [14, p.104] and to diagnostic reasoning.

Abduction

The logician and philosopher Charles Sanders Peirce introduced the term "abduction" to refer to a kind of plausible inference, which he took to be logically distinct from both induction and deduction [36]. An abduction passes from a body of data, to a hypothesis that *explains* or *accounts for* that data. Thus abduction is a kind of theory-forming or interpretive inference. In fact Peirce says in one place, "Abductive inference shades into perceptual judgment without any sharp line of demarcation between them." [37, p.304].

In their popular AI textbook Charniak and McDermott characterize abduction variously as modus ponens turned backwards, inferring the causal reasons behind something, generation of explanations for what we see around us, and inference to the best explanation [10]. They write that medical diagnosis, story understanding, vision, and understanding natural language are all abductive processes, and they speculate as to whether there might be possible a "'unified theory' of abduction" which will link all of these processes together [10, p.557].

Other AI practitioners have given similar characterization of abduction [26] [38] [39] [40], some have proposed or built systems using similar ideas without actually using the term "abduction" in describing their work [3] [33] [12] [35] [41] [34]. Some attempts have been made to cast the natural language understanding problem explicitly as abduction [11] [9]. Philosophers have written of "inference to the best explanation" [19]

[13] [21] and "the explanatory inference" [30]. Related philosophical traditions are the "hypothetico-deductive" model of the scientific method, and accounts of "the logic of discovery" [18]. Recently Paul Thagard has given abduction an important role in his analysis of the logic of scientific theory formation [46].

We may characterize **Abduction** as a form of inference that follows a pattern like this:

D is a collection of data (facts, observations, givens),
H explains D (would, if true, explain D),
No other hypothesis explains D as well as H does.

Therefore, H is probably true.

The confidence in the conclusion should (and typically does) depend on these factors:

- how decisively H surpasses the alternatives,
- how good H is by itself, independently of considering the alternatives (e.g. we will be cautious about accepting a hypothesis, even if it is clearly the best one we have, if it is not sufficiently plausible in itself),
- how thorough the search was for alternative explanations, and
- pragmatic considerations, including
  - the costs of being wrong and the benefits of being right,
  - how strong the need is to come to a conclusion at all, especially considering the possibility of seeking further evidence before deciding.

I hope my reader recognizes this form of inference as being common in ordinary life, and a part of the "scientific method". What I am proposing here is that it also occurs on many levels in perception.

In general, as Marr pointed out, it is important to distinguish the *goal* of a computation; from the *logic of the strategy* by which that goal can be achieved, from the specific *representations and algorithms* used to describe a specific strategy, and from *implementations* of those representations and algorithms [31, p.25]. Describing a layer of interpretive inference as "abduction" describes the goals of the inference, and suggests strategies to achieve them, as I hope will become clear in what follows. The discussion here will not directly address representation, algorithm, or implementation.

The Layered Abduction Model

Each layer of interpretation, or more precisely, each locus of hypothesis formation (leaving open the possibility of more than one per layer) I call an **agora** after the meeting place where the ancient Greeks would gather for dialog and debate. The picture is that an agora is a place where hypotheses of a certain type gather and contend and where under good conditions a consensus hypothesis emerges. In typical cases the emerging interpretive hypothesis will be a composite hypothesis, coherent in itself, and with different sub-hypotheses accounting for different portions of the data. For example in vision the edge agora can be thought of as the location where a set of edge hypotheses are formed and accepted, each specific edge hypothesis accounting for certain specific data from lower-level agoras.

Our model calls for the information processing at each agora to be decomposed into three functionally distinct types of activity, which we can call *evocation of hypotheses*, *instantiation of hypotheses*, and *composition of hypotheses*.

Evocation can occur bottom-up, a hypothesis being stimulated for consideration by the data presented at the layer below. In diagnosis we would say that the presence of a certain finding *suggests* that certain hypotheses are appropriate to consider. More than one hypothesis may be suggested by a given datum. Evocation can also occur top-down, either as the result of priming (an expectation from the level above), or as a consequence of data-seeking activity from above, which can arise from the need for evaluation. Evocations can in general be performed in parallel, and need not be synchronized.

Instantiation occurs when each stimulated hypothesis is independently scored for confidence (***evaluation***), and a determination is made of what part or aspect of the data the hypothesis can account for (***determination of explanatory scope***). This process is in general top-down, and in order to instantiate itself a hypothesis may seek data which was not part of its original stimulus[1]. The data which are accounted for may or may not be identical to the data upon which the hypothesis was scored, or the data which did the evoking.

In the course of instantiation the hypothesis set may be expanded by including subtypes and supertypes of high-confidence hypotheses [2]. Instantiation is typically based on matching against prestored patterns of features, but instantiating "by synthesis" is also possible whereby the features to match are generated at run time. The result of a wave of hypothesis instantiation is a set of hypotheses, each with some measure of confidence, and each offering to account for some portion of the data. Usually many of the evoked hypotheses can be ruled out, and will not form part of the result. Since in a wave of instantiation hypotheses are considered independently of each other, this too can go on in parallel.

Composition occurs when the instantiated hypotheses interact with each other and (under good conditions) a coherent best interpretation emerges. Note that, as this stage begins, each hypothesis has both a confidence value, and a body of data that it can account for. In the end some hypotheses will have been incorporated into the composite hypotheses, some will have been excluded, and perhaps some will be in limbo as a result of some remaining ambiguity of interpretation.

Strategy for Composition of Hypotheses

For hypothesis composition an overall abduction problem has been set up: to account for all of the (reliable and important) data presented by the agora(s) immediately below. A series of small abduction problems is also set up: to account for each particular datum. A basic strategy is to try to solve the overall abduction problem by solving a sufficient number of smaller and easier abduction problems. We begin by solving the easiest small abduction problems, the ones in which we can have the most confidence. If a certain hypothesis is ***the only plausible explanation*** for some finding (it accounts for the finding and its local-match confidence value is not too low), then it is entitled to high confidence, and entitled to be accepted into the overall composite hypothesis that represents the solution to the overall abductive problem.

Let us call a hypothesis "BELIEVED" when it has been accepted as the correct interpretation for the data it offers to account for. Data accounted for by BELIEVED hypotheses are "ACCOUNTED-FOR" and are considered to be successfully interpreted. Let us call a hypothesis "ESSENTIAL" if it is the only plausible explanation for some reliable datum (which is typically a hypothesis at the next lowest level that is BELIEVED). Thus an ESSENTIAL hypothesis scores positively and accounts for data items for which there are no other good interpretations. ESSENTIAL hypotheses are BELIEVED. Information about the explanatory relationships is thus used to increase the confidence in certain hypotheses.

If not all of the data are yet accounted for, the next step is to propagate the consequences of the initial set of BELIEVED hypotheses. These consequences arise as the result of causal and statistical relationships between hypotheses typically stored as compiled knowledge in advance of processing. There are several kinds of these relationships—I describe them here just briefly. Hypotheses at the same level (in the same agora) can have relationships of compatibility, entailment, or incompatibility, which can be a matter of degree. Propagating the consequences of BELIEVED hypotheses by taking account of these relations requires the appropriate adjustment of scores for related viable hypotheses outside of the BELIEVED set, or other appropriate actions. For example a hypotheses incompatible with a BELIEVED hypotheses can be rejected categorically, and removed from further consideration. Another kind of relationship is where a hypotheses "EXPECTS" the

[1] Under certain data-driven circumstances it is good enough just to score on the basis of voting by the stimulating data from below, and then no top-down processing need occur, at least for scoring.

[2] In general the space of potential hypotheses can be assumed to be hierarchically organized by level of specificity.

presence of data items at the next lower level (this too can come in degrees). Propagating such an expectation requires evoking the hypothesis corresponding to the expectation (priming) if it has not already been evoked. If it has, it can be given an extra measure of confidence. If a strong expectation is ***contradicted*** by the data, an anomaly has occurred, and special handling is appropriate.[3]

A hypothesis is a "CLEAR-BEST" if it is the distinctly best explanation (by confidence level) of some data item. CLEAR-BEST hypotheses are BELIEVED too. Note that an ESSENTIAL or CLEAR-BEST hypothesis is the uniquely best explanation for some data items—it is a local abductive conclusion. If not all of the data has been accounted for, the consequences of CLEAR-BESTs are propagated similarly to the ESSENTIAL hypotheses. Note that this propagation can result in more hypotheses becoming CLEAR-BESTs, e.g., if high-scoring explanatory competitors are removed from consideration, or if propagating consequences readjusts hypothesis scores so that a clear winner emerges.

If the ESSENTIAL hypotheses together with the CLEAR-BESTs do not account for everything, we have done all we can do on the current evidence without resorting to guessing. Generally our best strategy under these circumstances would be to go back for more data. In fact we are in a position to guide the data gathering by focusing on the problem of discriminating between alternative good explanations for significant data items. This is a form of top-down processing we may call "focused disambiguation". Sometimes, however, we have all of the relevant data we are going to get, for example we may be unable to ask the speaker to repeat. Under these circumstances we still have the means available to do some clever guessing. We can begin to include hypotheses which are best explanations for certain findings, but which are not far enough ahead of the alternatives, or not of high enough local-match confidence, to enable them to be accepted confidently. These WEAKLY-BESTs constitute the best guesses we can make under the circumstances. Actually some of them can be accepted with a fairly high degree of confidence. A finding can be made to vote for the hypotheses which best explain it (with voting strength in proportion to the measure by which the hypothesis beats its nearest competitor). The idea is that two different findings, both pointing to the same hypotheses as the best explanation constitute (apparently) independent sources of evidence for the hypothesis, i.e., constitute converging lines of inference for the hypothesis. Hypotheses with more votes can be accepted more confidently than hypotheses with fewer votes, and perhaps enough can be confidently accepted to complete the explanation.

Now in general relationships (spatial, grammatical, etc.) between the parts of a hypothesis are significant and need to be maintained. Some of these relationships can be seen as the filling of related roles in higher-layer interpretive hypotheses, for example a diagnostic hypothesis of a flow going on between A and B would bind A-related and B-related data together into relationships. But some other relationships (e.g. spatial in low level vision) are presumably compiled into the hardware, so that the appropriate constraints are applied between neighboring hypotheses as an automatic result of the operation of the machinery. Still and all, the net impact on hypothesis composition of these relationships can probably be captured by basic relationships of mutual sympathy and antipathy.

At the end of a wave of composition activity certain hypotheses have been accepted as BELIEVED. These constitute a confident best explanation for a portion of the data. Often there will also remain a set of unexplained data, and a set of viable hypotheses which, at various levels of confidence, offer to explain that data, but for which no clear solution is apparent. Nevertheless the BELIEVED hypotheses may be enough data for the next higher layer to do its business; resolving the remaining ambiguities may be unimportant in the context. Alternatively, remaining ambiguities may get resolved later as a result of further processing at that layer stimulated by downward-flowing expectations.

Downward-Flowing Processing

We may distinguish at least four sources or functions of top-down processing. One is that the data-seeking needs of hypothesis evaluation can provoke computation of the data (top-down evocation and evaluation of a hypothesis) as was discussed above. Another that was mentioned is that expectations based on firmly established hypotheses at one layer can prime certain data items (i.e. evoke consideration of them and bias

[3]Throughout the processing various kinds of anomalies can occur. Anomalies are detected and recorded, and typically stimulate special handling; from here on I describe the course of processing only for when everything goes smoothly.

their score upwards). A third way is that hypotheses that are uninterpretable as data at the higher level (no explanation can be found) can be "doubted" and reconsideration of them provoked. Finally data pairs that are jointly uninterpretable, as for example two words, the co-occurrence of which cannot be reconciled syntactically or semantically, can be considered to be incompatible (to some degree of strength) and recomputation of the composite hypothesis can be provoked from above. In these ways higher-level interpretations can exert a strong influence on the formation of hypotheses at lower levels, and layer-layer harmony is a two way street.

Recovering from Mistakes

Mistakes in Initial Hypothesization and Scoring

- Hypothesis suggestions come from above as well as below, thus hypotheses which would be missed on bottom-up processing can still be considered.
- If suggestions are inadequate, e.g. no hypotheses are evoked covering a segment of data, or all suggestions score low, exhaustive search (though hierarchically organized for efficiency) is undertaken to broaden the hypotheses being considered, thus hypotheses that are missed on suggestion-based stimulation can still be considered.
- Hypothesis evaluation is augmented by encouragement and discouragement (resulting from positive associations and incompatibilities) from other hypotheses in the same agora. Thus the local-match confidence score is improved by contextual information.
- Hypotheses evaluation is augmented by encouragement and discouragement based on expectations derived from confident higher-level hypotheses. This constitutes another kind of context-based improvement and check on the the confidence score.
- The acceptance of a hypothesis is based on how well it surpasses explanatory alternatives, thus after recognition-based scoring, a significant additional uncertainty-reducing operation is performed before acceptance.
- Strength of confidence is supported by "the consilience of inductions" whereby converging lines of inference all support the same hypotheses. Thus system performance should be robust.
- Acceptance, when it finally occurs, is still tentative and liable to be overthrown by relationships to the mass of other confident hypotheses.

Mistakes in Choice of Initial Islands of Confidence

- Actually the islands are very strong. They are never based only on a hypothesis having high initial confidence; it is at least required to also be a distinctly best explanation for some datum.
- Inconsistencies lead to detected anomalies, which lead to special strategies that weigh alternative courses of action. Originally accepted hypotheses can collide with others and subsequently called into question.
- Inconsistency collisions can occur laterally, or from above (violation of expectation, or from below (violation of expectation), and can come in degrees of strength. In effect there is broad cross checking of accepted hypotheses.
- An inexplicable datum should be doubted and called into question—it may not really be there. If after re-evaluation the datum remains strong despite the doubt, then the system can detect that it has encountered the limits of its knowledge, and is positioned to learn a new hypothesis category.
- Sometimes two parts of a compound hypothesis are inconsistent in context, where the judgment of higher levels is that they cannot both occur, based upon the inability to form a consistent hypothesis at the next highest level. (It seems that this can account for unstable perceptual objects like the Necker cube.)

Summary of the Control Strategy

We may summarize the control strategy by saying that it employs multi-level and multiple intra-level island-driven processing. Islands of relative certainty are seeded by local abductions and propagate laterally (incompatibilities, positive associations), downwards (expectations), and upwards (firm items of data to be accounted for). Processing occurs concurrently and in a distributed fashion. Higher levels provide *soft* constraints through the impact of expectations on hypothesis evocation and scoring, but do not strictly limit the hypothesis space.

Extension of the Model to Multi-Modal Perception

The basic idea in extending the model to multi-modal perception, i.e. perception that combines the information from more than a single sense, is that combining information from different senses is functionally no different than combining information from different channels within one sense modality. Different channels within the visual system deliver up the data useful at a certain level to form hypotheses about the locations of 3-d objects within the visual space; similarly, different senses deliver up the data useful for forming hypotheses about, say, object identity.

One special processing problem for multi-sense integration is the problem of identifying a "That" delivered up by one sense, with a "That" delivered up by another. Which person is the one that is speaking? Is it the same object being seen in the infrared as that being seen in x-rays? Logically, it should be possible for information derived from one sense to help with resolving distinct objects within the other sense. There is actually some evidence that vision can help hearing to separate distinct streams of tones [32, p.83] and hear the tone stream as two distinct auditory objects.

One useful computational support for cross modal perception is provided by correlated spatial representations, as our visual maps are correlated with our auditory maps of the space surrounding us. Thus, for example, a robot should bring together separate channels of information from its senses of "sight" and "touch" into a unified spatial representation of its immediate surroundings. Moreover this "hot map" of its surroundings should be maintained continually, and updated and revised as new information arrives and is interpreted. This hot map, with its symbols on it, can be viewed as the resulting composite hypothesis formed at "the agora of objects in the immediate surroundings" by a process of abductive interpretation.

Yet some senses are not particularly spatial (e.g. smell). We can envision computational support for cross-modal perception in the form of pattern-based recognition knowledge, where the compiled recognition patterns for an object category rely on features from more than one sense. This is very analogous to medical diagnosis where a disease is recognized from evidence from such disparate sources as lab tests, x-rays, and patient history. Such recognition knowledge can be used to support an "agora of the patient's disease" in much the same manner as the robot mentioned above maintaines its map of objects in its surroundings. Somewhat further along we can envision a robot that maintains an "agora of understanding" whereby it monitors some complex device and continually maintains a causal understanding of it . Much much further along we can imagine building a robot scientist who maintains an "agora of theoretical understanding" whereby its best understanding of the world is maintained.

Summary: Perception as Compiled Cognition

The formation of a composite best-explanation hypothesis at any level in perception is treated as a problem of abductive inference, similar to diagnosis and theory formation. Thus this model brings a knowledge-based problem-solving approach to the analysis of perception, treating perception as a kind of "compiled" cognition.

Acknowledgments

The model described here is an outgrowth of research on diagnostic reasoning that has gone on at Ohio State Laboratory for AI Research over the past decade in the context of a developing theory of generic information

processing tasks [4], [6], [5], [7], [8], [42], [43], [44]. Various facets of abductive reasoning have been investigated, especially the problem of assembling composite explanatory hypotheses [24], [25], [26], [1], [20], [16], [15], [45], [23] [22].

This work has been supported by the Defense Advanced Research Projects Agency under RADC contract F30602-85-C-0010; and the NIH Heart, Lung and Blood institute, grant number 1 RO1 HL38776-01. The development of the layered abduction model has benefited especially from discussions with B. Chandrasekaran, Susan Josephson, Dean Allemang, Ashok Goel, Tom Bylander, Michael Tanner, Terry Patten, and Jordan Pollack, as well many discussions at the Image Understanding Workshop sponsored by DARPA/ISTO in April 1988.

References

[1] D. Allemang, M. C. Tanner, T. Bylander, and J. R. Josephson. On the computational complexity of hypothesis assembly. In *Proc. Tenth International Joint Conference on Artificial Intelligence*, Milan, August 1987.

[2] Jerome S. Bruner. On perceptual readiness. *Psychological Review*, 64(2):123–152, 1957.

[3] B. G. Buchanan, Sutherland, G. L., and E. A. Feigenbaum. *Heuristic DENDRAL: a program for generating explanatory hypotheses in organic chemistry*, pages 54–69. Edinburgh University Press, Edinburgh, 1969.

[4] B. Chandrasekaran. Generic tasks in knowledge-based reasoning: High-level building blocks for expert system design. *IEEE Expert*, pages 23–30, 1986. Fall 1986.

[5] B. Chandrasekaran. Towards a functional architecture for intelligence based on generic information processing tasks. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, 1987.

[6] B. Chandrasekaran. Generic tasks as building blocks for knowledge-based systems: The diagnosis and routine design examples. Technical report, The Ohio State University, 1988.

[7] B. Chandrasekaran, F. Gomez, S. Mittal, and J. W. Smith. An approach to medical diagnosis based on conceptual structures. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 134–142, 1979.

[8] B. Chandrasekaran and S. Mittal. Conceptual representation of medical knowledge for diagnosis by computer: Mdx and related systems. In M. Yovits, editor, *Advances in Computers*, volume 22, pages 217–293. Academic Press, 1983.

[9] E. Charniak. A neat theory of marker passing. In *Proceedings of AAAI-86*, volume 1, pages 584–588. AAAI, Morgan Kaufmann, August 1986.

[10] Eugene Charniak and Drew McDermott. *Introduction to Artificial Intelligence.* Addison Wesley, 1985.

[11] Venugopala Rao Dasigi. *Word Sense Disambiguation in Descriptive Text Interpretation: A Dual-Route Parsimonious Covering Model.* PhD thesis, University of Maryland, College Park, 1988.

[12] Johan de Kleer. Reasoning about multiple faults. *AI Magazine*, 7(3):132–139, August 1986.

[13] R. Ennis. Enumerative induction and best explanation. *The Journal of Philosophy*, LXV(18):523–529, September 1968.

[14] Jerry Fodor. *The Modularity of Mind.* Bradford Book, 1983.

[15] Ashok Goel, J. Ramanujan, and P. Sadayappan. Towards a 'neural' architecture for abductive reasoning. In *Proceedings of the Second International Conference on Neural Networks*, volume 1, pages 681–688, 1988.

[16] Ashok Goel, P. Sadayappan, and John R. Josephson. Concurrent synthesis of composite explanatory hypotheses. In *Proceedings of the Seventeenth International Conference on Parallel Processing*, pages 156–160, august 1988.

[17] Richard L. Gregory. Perception as hypotheses. In Richard L. Gregory, editor, *The Oxford Companion to the Mind*, pages 608–611. Oxford University Press, 1987.

[18] N.R. Hanson. *Patterns of Discovery*. Cambridge University Press, 1958.

[19] Gilbert Harman. The inference to the best explanation. *Philosophical Review*, LXXIV:88–95, January 1965.

[20] John Josephson. A framework for situation assessment: Using best-explanation reasoning to infer plans from behavior. In *Proceedings of Expert Systems Workshop*, pages 76–85, April 1987.

[21] John R. Josephson. *Explanation and Induction*. PhD thesis, The Ohio State University, 1982.

[22] John R. Josephson. Reducing uncertainty by using explanatory relationships. In *Proceedings of the Space Operations Automation and Robotics Conference-1988 (Soar-88)*, pages 149–151, 1988.

[23] John R. Josephson. Towards a generic architecture for layered interpretation tasks—implications for plan recognition. Technical report, The Ohio State University, 1988.

[24] John R. Josephson, B. Chandrasekaran, and Jack Smith, Jr. Assembling the best explanation. In *Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems*, pages 185–190, Denver, Colorado, December 1984. IEEE Computer Society. A revised version by the same title is available as a technical report.

[25] John R. Josephson, B. Chandrasekaran, Jack Smith, Jr., and Michael C. Tanner. Abduction by classification and assembly. In *PSA 1986 Volume One*, volume 1, pages 458–470, East Lansing, Michigan, 1986. Philosophy of Science Association.

[26] John R. Josephson, B. Chandrasekaran, Jack Smith, Jr., and Michael C. Tanner. A mechanism for forming composite explanatory hypotheses. *IEEE Transactions on Systems, Man and Cybernetics, Special Issue on Causal and Strategic Aspects of Diagnostic Reasoning*, SMC-17(3):445–54, May,June 1987.

[27] Immanuel Kant. *Critique of Pure Reason*. St. Martins Press, (1968), New York, 1787. tr. Norman Kemp Smith.

[28] V.R. Lesser, R.D. Fennell, L.D. Erman, and R.D. Reddy. The Hearsay II speech understanding system. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-23:11–24, 1975.

[29] Margaret Livingstone and David Hubel. Segregation ofform color, movement, and depth: Anatomy, phisiology, and perception. *Science*, 240, May 1988.

[30] William G. Lycan. Epistemic value. *Synthese*, 64:137–164, 1985.

[31] David Marr. *Vision*. W. H. Freeman and Company, 1982.

[32] Dominic William Massaro. *Speech Perception by Ear and Eye:A Paradigm for Psychological Inquiry*. Lawrence Erlbaum Associates, New Jersey, 1987.

[33] Randolph A. Miller, Harry E. Pople, Jr., and Jack D. Myers. Internist - i, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, 307:468–476, 1982.

[34] R. S. Patil. *Causal representation of patient illness for elecrolyte and acid-base diagnosis*. Ph. D Thesis, 1981.

[35] J. Pearl. Distributed revision of composite beliefs. *Artificial Intelligence*, 33(2):173–215, October 1987.

[36] Charles S. Peirce. Abduction and induction. In Justus Buchler, editor, *Philosophical Writings of Peirce*, chapter 11, pages 150–156. Dover, 1955.

[37] Charles S. Peirce. Perceptual judgments. In Justus Buchler, editor, *Philosophical Writings of Peirce*, pages 302–305. Dover, 1955.

[38] H. Pople. On the mechanization of abductive logic. *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 147–152, 1973.

[39] James Reggia. Abductive inference. In Kamal N. Karna, editor, *Proceedings of The Expert Systems in Government Symposium*, pages 484–489. IEEE Computer Society Press, 1985.

[40] James A. Reggia, Barry T. Perricone, Dana S. Nau, and Yun Peng. Answer justification in diagnostic expert systems–part 1: Abductive inference and its justification. *IEEE Transactions on Biomedical Engineering*, BME-32(4):263–267, April 1985.

[41] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.

[42] Jack Smith, Jr., John R. Svirbely, Charles A. Evans, Pat Strohm, John R. Josephson, and Michael C. Tanner. Red: A red-cell antibody identification expert module. *Journal of Medical Systems*, 9(3):121–138, 1985.

[43] Jack W. Smith, Jr. *RED: A Classificatory and Abductive Expert System.* PhD thesis, Ohio State University, August 1985.

[44] Jon Sticklen. *MDX-2: An Integrated Medical Diagnostic System.* PhD thesis, The Ohio State University, 1987.

[45] Mike Tanner and John Josephson. Abductive justification. Technical report, Ohio State University Laboratory for AI Research, 1988.

[46] Paul Thagard. *Computational Philosophy of Science.* Bradford Books / MIT Press, 1988.

N90-29852

RCTS: A FLEXIBLE ENVIRONMENT FOR SENSOR INTEGRATION AND CONTROL OF ROBOT SYSTEMS - THE DISTRIBUTED PROCESSING APPROACH

R. Allard, B. Mack, M.M. Bayoumi

Department of Electrical Engineering
Queen's University
Kingston, Ontario, Canada K7L 3N6

Abstract

Most robot systems lack a suitable hardware and software environment for the efficient research of new control and sensing schemes. Typically, engineers and researchers need to be experts in control, sensing, programming, communication and robotics in order to implement, integrate and test new ideas in a robot system. In order to reduce this time, the Robot Controller Test Station (RCTS) has been developed. It uses a modular hardware and software architecture allowing easy physical and functional reconfiguration of a robot. This is accomplished by emphasizing four major design goals: flexibility, portability, ease of use and ease of modification. This paper mainly reviews an enhanced distributed processing version of RCTS. It features an expanded and more flexible communication system design. Distributed processing results in the availability of more local computing power and retains the low cost of micro-processors. A large number of possible communication, control and sensing schemes can therefore be easily introduced and tested, using the same basic software structure.

1.0 Introduction

As part of the space station, the Mobile Servicing System (MSS) will be utilized to perform tasks such as carrying out basic repairs, assembling structures, cleaning surfaces and servicing satellites as discussed in [1-3]. Such capabilities in a robot system will require significant advances in the state of the art. A flexible development environment is therefore required. The Robot Controller Test Station (RCTS) has been developed as a tool to facilitate development, integration, testing and verification of robot sensors and controllers. Some other test stations currently exist; however, none of these stations highlight the four basic design goals of RCTS which are: flexibility, portability, ease of use and ease of modification. The user interface consists of a series of menus. The user chooses a number of application routines within a special purpose library and specifies operating parameters. The system features a simulation interface in addition to a robot interface. Graphic facilities are also provided to display the results of the run-time data analysis routines. RCTS is based on a three level hierarchical approach (high, intermediate and low levels). This subdivides the control and sensing problem into different priorities and levels according to their processing times and their functions. The low level control is the direct interface with the robot, and the high level initiates the robot tasks to be carried out (Figure 1). This three level structure permits sensor integration to be carried out on several levels, which allows the robot to respond to sensor input more rapidly. Detailed reviews of the initial version of RCTS are provided in [4-6].

This paper reviews an expanded and more general distributed processing version of RCTS, which is presently being implemented. It features an updated communication interface and a variety of utilities which help the user to easily modify the system and enhance flexibility. In this paper, the basic structure of RCTS is first reviewed, followed by details of the updated distributed processing version and an overview of the hardware implementation.

2.0 Overview of Basic Structure of RCTS

2.1 Arrangement of modules

The RCTS software is organized into six different modules, as outlined in Figure 1. This approach decomposes the robot control and sensing problem into three separate levels (high, intermediate, and low levels) and also separates control from sensor processing. It improves the flexibility and modularity of the system since each part is clearly defined and bounded. A detailed analysis of the module arrangement is provided in [4]. The following is a review of the functions of each basic module:

Robot Task Generator (high level control): This module specifies what task the robot should accomplish. The task may be altered with information received from the high level sensing module.

High Level Sensor Interface: This module processes data from a sensor interfaced to this module or from the intermediate level sensor interface module. It transforms the data into a form which may be used by the robot task generator to alter its operation.

Trajectory Generator (intermediate level control): This module provides the position or torque setpoints used by the low level controller when the information is requested.

Intermediate Level Sensor Interface: This module processes data from a sensor interfaced to this module or from the low level sensor interface module. The data is converted to a form suitable for the trajectory generator. This module may also send data to the high level sensor interface module.

Low Level Controller: This module calculates the drive signals for the robot from the setpoints provided by the trajectory generator and from data provided by the low level sensor interface module.

Low Level Sensor Interface: This module processes data from a sensor interfaced to this module. It sends the information to the low level controller or to the intermediate level sensor interface. This module interfaces sensors which usually have a rapid execution time, since the information is required at a high rate by the low level controller.

2.2 Module structure

Each RCTS module has an identical structure, consisting of a pre-process, process and post-process (Figure 2). The pre-process and post-process are the communication interfaces with other modules. The objective is to separate the communication interface from the actual application processing so that any section can be easily modified. The pre-process section receives commands, data and status signals from the other modules. The process uses a state table approach which consists of a set of conditions testing the internal logic states and inputs to a module. If its associated test result is positive, an application routine will execute. The routines executed in the main process update the logic state and determine the module's output signals. When all the tests have been conducted, control is then transferred to the post-process. The post-process transmits information to other modules or the robot. All the communication signals (commands, module status and internal states) are standardized for all of the modules. This makes the software easier to comprehend and modify.

3.0 Distributed Version

3.1 Principle

A laboratory or advanced robot systems similar to the MSS may require rapid integration of new sensors and reallocation of the modules to different processors. In the context of the MSS, for example, a substantial expansion of capabilities can be expected over the lifespan of the system. This may require significant changes to the processing architecture of the system. The RCTS software design philosophy can accommodate a wide range of communication schemes, both internal and external. Distributed processing is incorporated into RCTS to increase the processing power available to different modules by executing them on different processors. This also promotes parallel execution. It is particularly useful when modules are computationally intensive or need to execute at high rate (e.g., vision processing and low level control). A review of communication issues in robotics can be found in [7]. The initial version of RCTS features some distributed processing capabilities and is able to share information between processors, using locally controlled communication. This could typically be interrupt driven communication or messaging. Interrupt driven communication requires that interrupt lines be wired every time a sensor is added. These interrupt lines customize the system and may complicate the system design for the user; this violates the original design goals of RCTS (flexibility, portability, ease of use and ease of modification). The initial version is also not efficient for centrally controlled communication (e.g., polling), which may use a master--slave hierarchy, for example. In centrally controlled schemes, all information transfers are controlled by a single node. In the case of polling, for example, only the processor with the status of master can send information and interrogate other processors.

The upgraded version of RCTS features a modified communication software structure and a variety of enhanced capabilities. This eases system reconfiguration, the reallocation of processing power and the introduction of new communication interface schemes. The original pre-process and post-process sections are generalized and expanded. A new utility (Communication Relay Utility--CRU) is added and provides some synchronization and intermediate information storage functions. This permits the transfer of information between modules which cannot communicate due to a lack of direct physical link. Other functions are added to allow the user to define the network topology through the menu system (as opposed to a system designer changing the source code). The task of adding a new sensor, relocating a module to another processor or using a different communication scheme can therefore be carried out by the user.

3.2 Initial communication interface design

In the initial implementation, the pre-process and post-process sections of the RCTS modules are divided into three layers: upper, middle and lower as discussed in [5]. The upper communication interface layer determines the information to be transferred and its destination. The middle interface layer formats the information to suit the transfer method and the lower interface layer is responsible for the actual physical transfer. This layered approach emphasizes modularity, which permits each section to be designed and modified separately. It can accommodate both synchronous (e.g., memory sharing) and asynchronous (e.g., message passing) communication. For example, the following functions are carried out in the case of internal messaging: the upper layer determines the data content of the message and the destination information (target module). The middle layer formats this message to suit the mailbox size and affixes a command byte. The lower layer uses system calls or customized drivers to accomplish the physical transfer by loading the destination mailbox. The pre-processor is structured similarly, but it is executed in the reverse order. Since all the operating system calls are located into one section of the software, modifications are easily carried out.

3.3 Updated communication interface design

In the updated version of RCTS, the three communication layers (upper, middle and lower) are expanded into four layers (upper, middle, linking and lower layers) (Figure 3). The system contains both internal (between the modules residing on the same processor) and external (between modules located on different processors) communication. Examples of internal schemes are memory sharing and internal messaging (using mailboxes). Examples of external schemes are polling, interrupts and multiple access with collision detection (such as ethernet). These are different and pose special requirements. Network topology information is provided by the user from the menu system. The network topology files thus generated define which processors are linked together, the type of link, the processor communication status and the physical location of each RCTS module. In the event that an RCTS module is added or re-allocated to a new processor, the user modifies the files through the menu. The source code does not require any modifications. The system carries out an automatic reconfiguration by downloading all the necessary information to the correct node and readjusting the communication system.

Within the post-process, the upper communication layer is identical to the initial version and determines the data content and destination of the information. In the external case, when using a centrally controlled system, the communication routines used may depend on the network status of the sender and receiver (e.g., master or slave). Before the middle layer is accessed, a test is therefore made to determine this status. This uses the network topology information supplied by the user. It is a useful addition to the initial version, since it permits easier reconfiguration of the network (the source code is not modified), use of centrally controlled communication and information transfer to modules that do not have a direct link with the sender. When the communication status has been determined, the correct communication routines within the middle layer are accessed according to the relationship between the sender and the receiver. This allows the selection of any communication scheme, providing the required communication drivers are included in the system. The network topology files contain information specifying the correct driver. Similarly to the initial version of RCTS, the middle communication interface layer then formats the data to suit the communication method and adds a command byte (containing internal module states, commands and module status). For example, if the information transfer is determined to be internal and the system uses internal messaging, the middle layer will format the data to suit the mailbox size. In the case of an external message, with a system using polling, the middle layer will format the data according to the packet size (multiple packets may be generated). The linking layer is then accessed. This is an addition to the initial version. The function of the linking layer is to establish a communication link with the receiving processor. This may require sending of an interrupt signal, or detecting a possible collision within an ethernet network. The lower communication interface layer performs the actual physical transmission of the message. Usually, only the linking and lower layers will contain operating system calls. This simplifies any modifications required when using a different operating system. In the pre-process section, the linking layer also has the added function of sending any acknowledgement signals required by the sending node (fully synchronous external communication).

Some applications may require that two processors must communicate through a intermediate processor, because they are not directly linked. For example, two sensing modules may be located on different communication buses and need to share information. In this case, the information is received from the sending processor, and is retransmitted to the correct destination. For this purpose, a new utility has been created ("communication relay utility"--CRU). It is created as a separate entity and has an identical structure to the other RCTS modules (pre-process, process, post-process and uses state tables). The process section of the CRU will typically alter the message header bits to reflect the final destination. The CRU is triggered internally by an RCTS module embedded on the same processor.

4.0 Hardware and Software Implementation

The current hardware implementation is based on the iRMX-II real-time operating system (Figure 4). This is an object oriented, multitasking operating system. It permits the transfer of information between tasks using a series of user-defined mailboxes. RCTS is implemented on Intel 310 and 320 microcomputers (based on the 80286 and 80386 microprocessors, respectively) and controls a Puma 550 robot. The system also utilizes Intel single board controllers (8044 based) to interface simple devices such as proximity sensors. The updated version of RCTS is implemented with a Bitbus network linking all the processors. Bitbus is a serial communication bus which uses a polling scheme. With this scheme, a processor is either a master or a slave. Only a processor which has the master status can initiate communication. The ease of use, flexibility and the low cost of Bitbus are its major advantages. This section reviews the details the implementation of a centrally controlled communication scheme in the updated version of RCTS. The low level control and low level sensing modules are located on the same processor. Because of speed requirements, the robot and the low level sensors are interfaced to the processor by a parallel interface.

In a polling based environment, a processor is either a master or a slave. Only the processors having the status of master can initiate communications. A slave cannot initiate any communications on its own, but must wait for a handshaking signal (called a poll) from the master. When a slave wants to transmit a message, it must first store it in a buffer which will later be accessed by the master. If a master wants to transmit to a slave under its jurisdiction, it must first send a poll message containing the data and wait for a message reception acknowledgement signal. The presence of acknowledgement signals complicates the communication system. It can cause large delays to a processor wanting to send messages to different processors. Bitbus, however, has the advantage of permitting the formation of a multi-layer network. Each level is able to communicate with its neighbour. The current system uses two buses (Figure 4) communicating through the processor hosting the intermediate level control module. This structure is more suited to the hierarchy of RCTS modules (high, intermediate and low level control and sensing). In a centrally controlled bus, the demands on a single master to poll every other processor and then transfer the message to the correct destination would introduce excessive communication delay. Since control information flows from the high to low level (sensing information flows from low to high level), a master node containing a control module is therefore able to request information to the control module situated on a higher level, or to sensors on the same level. This, therefore, prioritizes the control modules over the sensing modules, and permits them to meet their real-time requirements. For the sensing modules, it increases the execution time and results in a slower response. If no new sensing data is generated between two polls sent by the control modules, old data is transmitted. This ensures that data is always available and eliminates the need to send multiple poll signals to capture new data.

(1) Master to slave communication on the same bus level

Message transmission by the master: In common with all the situations explained in this section, a verification of the functional status of the sender and receiver processors is first carried out. For example, in the present case, the sender is a master and the receiver is a slave under its jurisdiction. The correct communication routines are then chosen using this information. The message is formatted by the middle communication interface layer, according to the format used by Bitbus (13 bytes of data and 7 bytes of header). The linking layer receives a request to send a message from the middle communication interface layer and interrupts the application routine at the slave by sending a poll signal. The master then waits for an acknowledgement signal (coming from the slave pre-processor's linking layer). The message then goes to the pre-processor of one of the slave modules. Upon completion of the message transmission, control is passed back to the linking layer which waits for a reply message.

Message reception by the slave: The slave's pre-process receives a polling signal and interrupts its processing to access the linking layer of the pre-processor. The message is first transferred to the pre-process lower layer. The upper interface communication layer then decodes the message in a form usable by the process (the header bytes are removed and the data is assigned to the correct variables).

(2) Slave to master communication on the same bus level

Message transmission by the slave: The slave cannot transmit a message to the master without having first been interrogated by the latter. It first waits to receive a poll signal and then returns a reply message containing the data. Normally, the master will only issue a poll when it wants to receive the message. This results in some synchronizing problems and causes the slave task to lock up until the message transmission can be completed. To circumvent this problem, a temporary memory storage area is used to unload the module's post-process. When a poll is received, the buffer (located in the lower communication interface layer) is accessed and the data is then transferred externally to the receiving module.

Message reception by the master: When a request for data or command is encountered in the pre-process of the receiving module (at the master), a poll signal is issued to the slave module. If no new data or command is available (the sending module has not finished executing once), the old message is retransmitted. This ensures there is no synchronization problem and that timing requirements are met. If the old message was not retransmitted (or a code indicating the data has not changed), the receiving module (master) would have to reissue another poll signal later. This could cause processing delays or lock-up. This approach allows minimal delays of the control modules by providing them with data (new or old) at set interval times.

(3) Slave to slave or master to slave communication on different levels

An example of this case occurs when the intermediate level sensor interface module requests communication with the high level sensor interface (both are on different bus levels). The two modules do not have any direct links, so the messages have to be temporarily stored in a node that has communication access to both. Moreover, both are slaves, so they cannot initiate the communication. This results in a sizeable transmission delay, because of the time expended in waiting for poll signals generated externally. For this purpose, the communication relay utility (CRU) is used in each master node for message relaying. The absence of the CRU would not prevent the current implementation from being used. It would only limit sensor integration opportunities, since the different levels of sensing would not be able to communicate. The function of the CRU is to poll all the slave processors located on the level under its jurisdiction, for messages that would have to be re-transmitted to another node. It is triggered by one of the RCTS module embedded on the same processor. It has the same structure as the RCTS modules (pre-process, process and post-process, with all the sub-layers), and uses the same standard command and status input signals. The sending slave module must modify the header bytes (done in the middle interface layer), to reflect the temporary destination of the message. Some timing functions are also added to make sure the real-time constraints are respected.

5.0 Conclusion

The RCTS system is proving valuable to test and compare new control and sensing schemes. A researcher with limited knowledge of robot hardware and software can easily perform system integration duties that now require hours, instead of days or weeks. The distributed version allows greater computing power to be allocated towards RCTS, therefore permitting the development of more complex routines. This updated version integrates a larger variety of communication schemes (e.g., centrally controlled with master--slave hierarchy) with added flexibility and ease of use. The reconfiguration is also more easy and does not require a system designer. Future work regarding RCTS will concentrate on increasing the computing capacity and adding several utilities to the system. A new communication bus providing a larger bandwidth will likely be implemented in the near future (such as multibus II or S/NET). Parallel processing will also be investigated, using a new operating system (e.g., Harmony). The portability of RCTS permits such major changes to be carried out easily.

6.0 Acknowledgements

The authors would like to thank: John Griffioen and Peter Luscher for their contributions in creating software; Intel Corp. for some equipment donations; and funding from NSERC and the Manufacturing Research Corporation of Ontario.

References

[1] Werstiuk, H. and Gossain, D., "The Role of the Mobile Servicing System on Space Station," Proc. of the 1987 International Conference on Robotics and Automation, Raleigh, NC, USA, March 30 - April 3, 1987.

[2] Hunter, D., "An Overview of the Space Station Special Purpose Dexterous Manipulator (SPDM)," National Research Council of Canada, Technical Report No. 28817 (issue A), Ottawa, Canada, April 7, 1988.

[3] Vigneron, F., Caswell, R., Sachdev, S., and Ravindran, R., "Technology Research and Development Associated with the Mobile Servicing System," Presented at the Fourth CASI Conference on Astronautics, Ottawa, Canada, November 3 - 4, 1987.

[4] Mack, B. and Bayoumi M.M., "A Robot Controller Test Station (RCTS)," Proc. of the 19th International Symposium on Industrial Robots, Sydney, Australia, November 6 - 10, 1988.

[5] Mack, B., Allard R., and Bayoumi, M.M., "An Environment for the Development of Sensor-based Robot Software," Proc. of SPIE Conference on Intelligent Robots and Computer Vision, Cambridge, MA, USA, Vol. 1002, November 6 - 11, 1988.

[6] Mack, B. and Bayoumi, M.M., "Analysis of Sensing and Control Algorithms Using the Robot Controller Test Station," Presented at the Fifth CASI Conference on Astronautics, Ottawa, Canada, November 15 - 16, 1988.

[7] Gauthier, D., Freedman, P., Carayannis, G., and Malowany, A., "Interprocess Communication for Distributed Robotics," IEEE Journal of Robotics and Automation, Vol. RA-3, No. 6, December 1987, pp. 493 - 504.

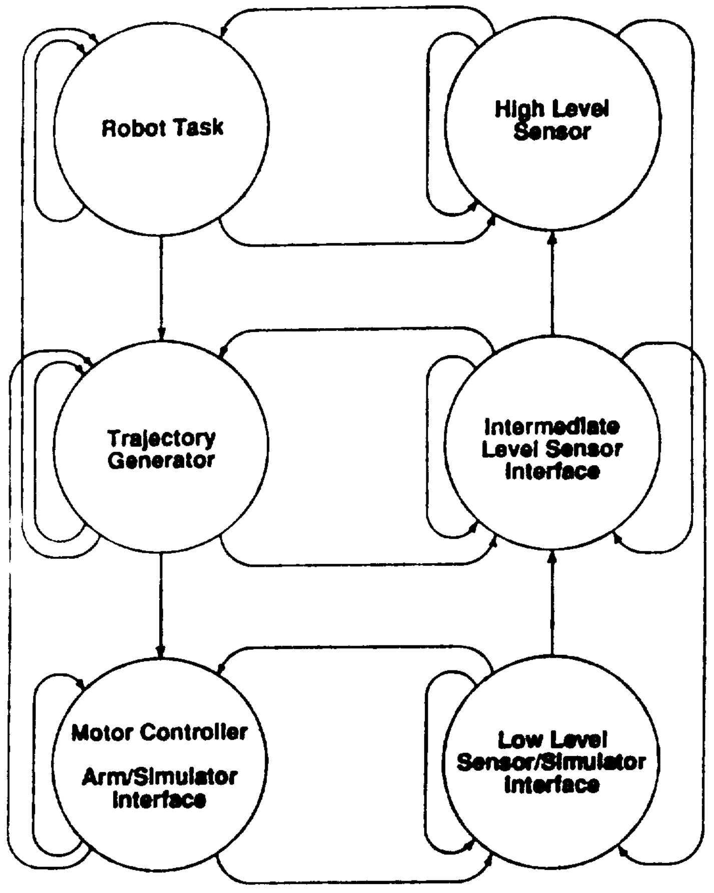


FIGURE 1: MODULE LAYOUT FOR RCTS

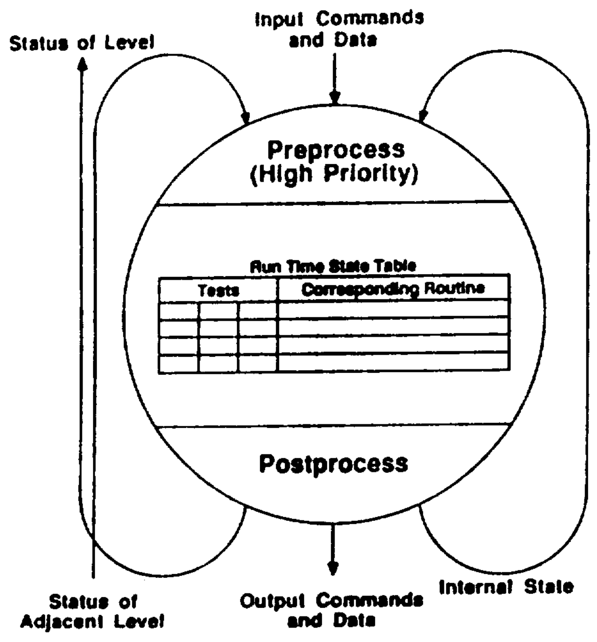


FIGURE 2: RCTS MODULE STRUCTURE

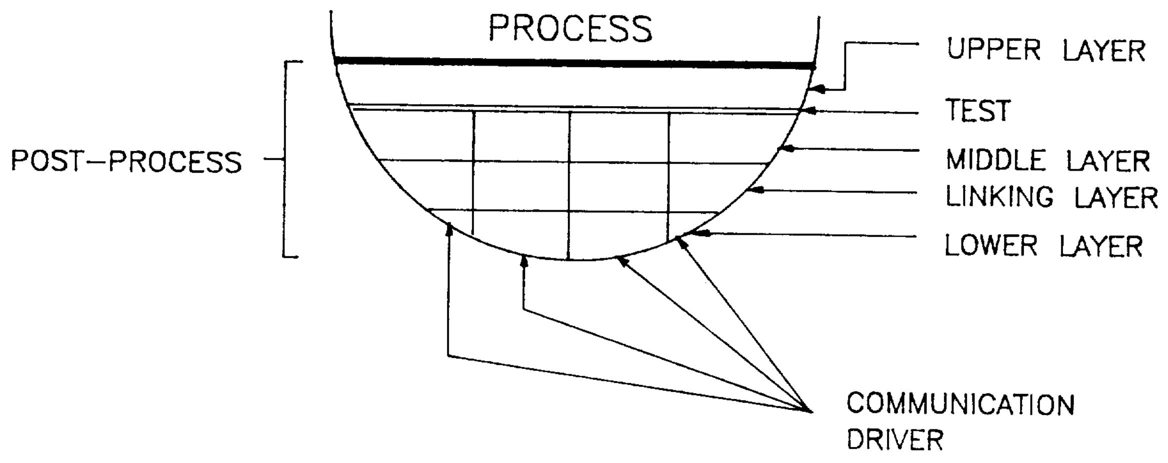


FIGURE 3: POST-PROCESS STRUCTURE

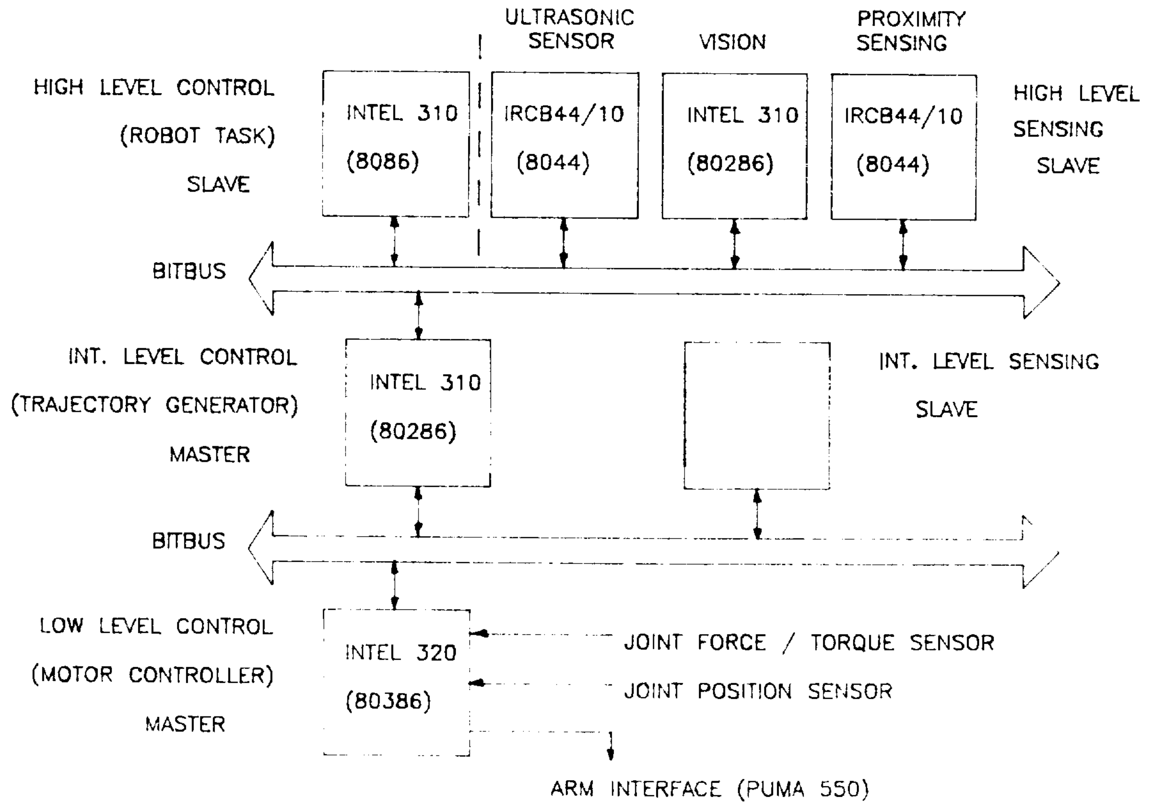


FIGURE 4: HARDWARE IMPLEMENTATION

N90-2[illegible]53

Vehicle path-planning in three dimensions using optics analogs for optimizing visibility and energy cost

Neil C. Rowe and David H. Lewis
Department of Computer Science, code 52Rp
U.S. Naval Postgraduate School
Monterey, CA 93943

Path planning is an important issue for space robotics. Finding safe and energy-efficient paths in the presence of obstacles and other constraints can be complex although important. We have been investigating high-level (large-scale) path planning for robotic vehicles in three-dimensional space with obstacles, accounting for (1) energy costs proportional to path length, (2) turn costs where paths change trajectory abruptly, and (3) "safety costs" for the danger associated with traversing a particular path due to visibility or invisibility from a fixed set of observers. We find paths optimal with respect to these cost factors.

We have in mind autonomous or semi-autonomous vehicles operating either in a space environment around satellites and space platforms, or aircraft, spacecraft, or smart missiles operating just above lunar and planetary surfaces. One class of applications concerns minimizing detection, as for example determining the best way to make complex modifications to a satellite without being observed by hostile sensors; another example is verifying there are no paths ("holes") through a space defense system. Another class of applications concerns maximizing detection, as finding a good trajectory between mountain ranges of a planet while staying reasonably close to the surface, or finding paths for a flight between two locations that maximize the average number of triangulation points available at any time along the path.

1. Our approach

Our major innovation is to view free space as a set of irregular polyhedra for path-planning purposes. Most previous explorations have divided space into a uniform lattice of cubes, and plan transitions between the centers of cubes. We think this leads to unnecessarily inefficient programs since large portions of space are very homogeneous in traversal characteristics while other portions are very inhomogeneous, so a uniform-resolution representation can be wasteful. And by grouping space into irregular meaningful regions (like space visible from a particular set of observers), the problem becomes more intuitive to a human; debugging is simplified, and human heuristics are easier to obtain. To further simplify matters, we assume energy expenditure is a proportional to path length, and that visibility danger is binary (either yes or no) so it only changes abruptly and at the boundary of what we call "visibility regions". We have already investigated a similar problem in two dimensions, that of finding routes across overland terrain with forests, shrub, grasslands, and other terrain types of varying traversability, using the idea of formulating "weighted" regions, regions of homogeneous characteristics [5, 4, 3, 6]. Apparently the only similar attempts such as [7] to find paths through irregular regions in three dimensions have been limited to obstacle-avoidance criteria, and much of that work is highly theoretical.

Since our goal is to find paths good with respect to energy, turn, and visibility cost, a path optimal with respect to those criteria is desirable if obtainable without excessive effort. The calculus of variations is the general solution to optimal-path problems; its methods are complex and few of its problems can be solved in a closed form. Fortunately, the problem we have described does not require it since our optimal paths must be piecewise-linear, where the pieces correspond to visibility regions. This follows from an important general principle we have used in many areas of path planning, the "shortcut meta-heuristic":

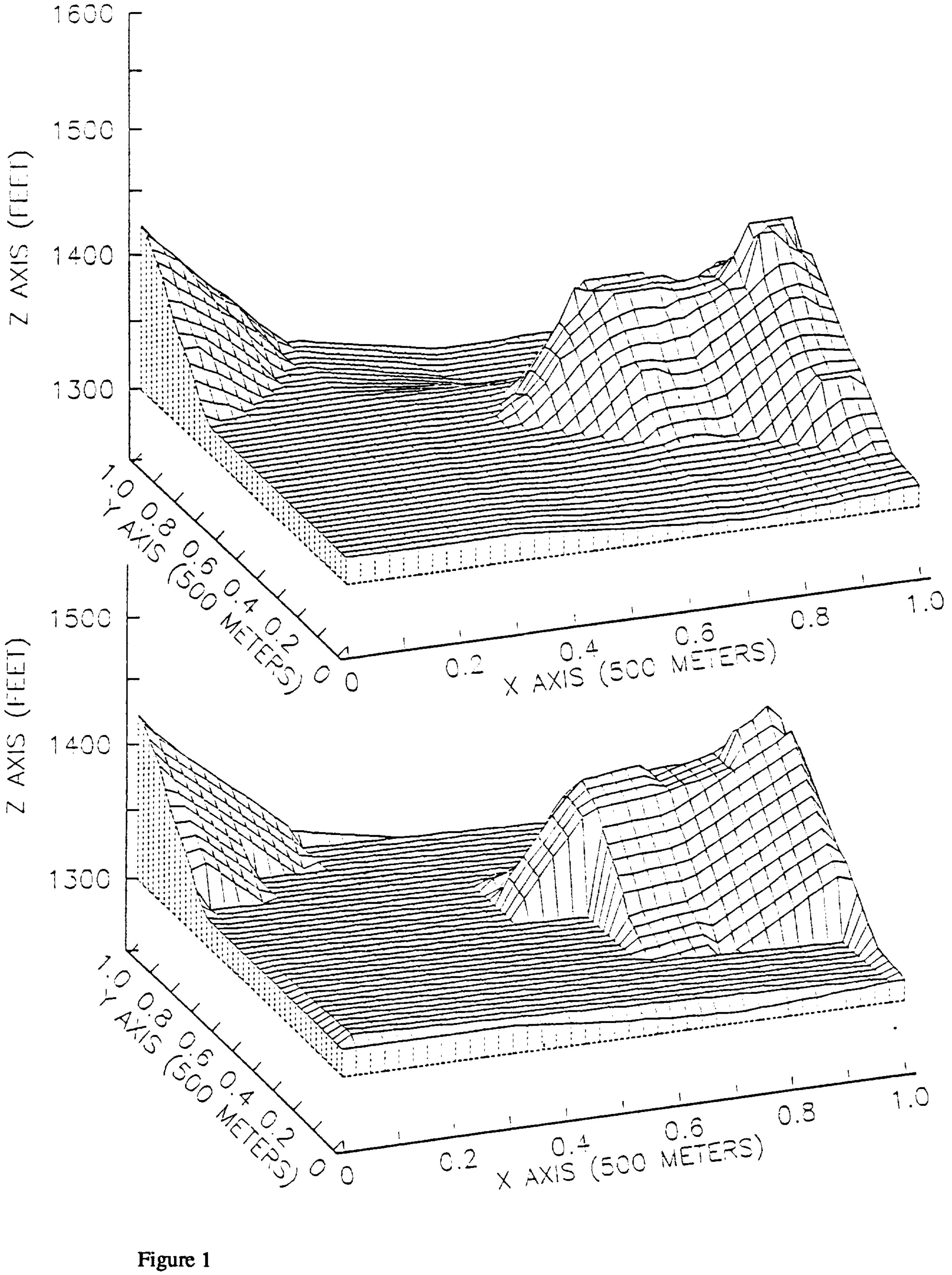

1600
1500
1400
1300
Z AXIS (FEET)
1.0 0.8 0.6 0.4 0.2 0
Y AXIS (500 METERS)
0 0.2 0.4 0.6 0.8 1.0
X AXIS (500 METERS)
1500
1400
1300
Z AXIS (FEET)
1.0 0.8 0.6 0.4 0.2 0
Y AXIS (500 METERS)
0 0.2 0.4 0.6 0.8 1.0
X AXIS (500 METERS)

Figure 1

optimal paths only turn or curve when any shortcut across the turn would be worse. No such conditions are present, in the problem described above, in interiors of regions of homogeneous visibility and homogeneous cost per unit traversal.

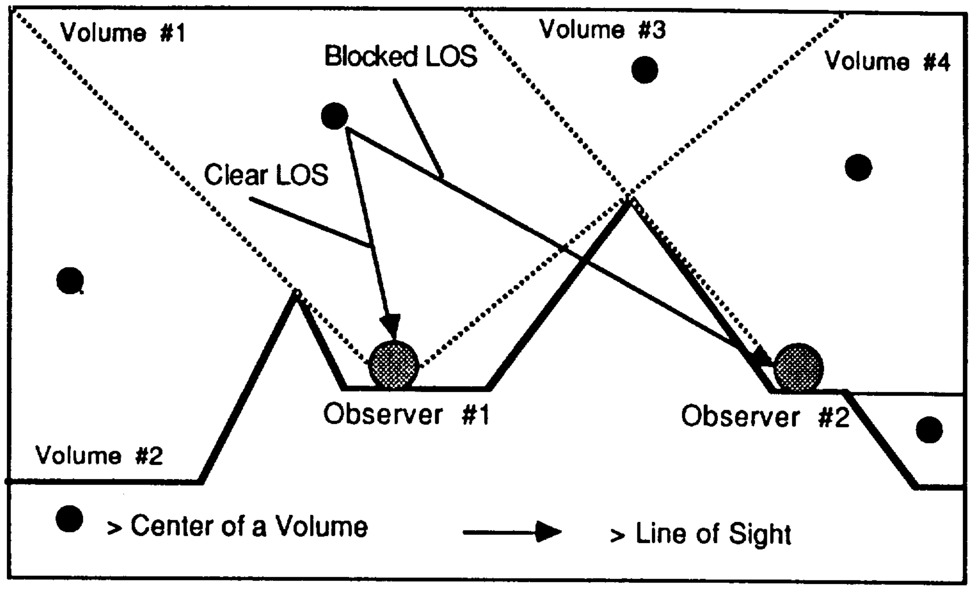
Furthermore, we can show that the piecewise-linear optimal paths in our problem must obey Snell's Law from optics when they do turn on the boundaries of regions, if we make a few reasonable assumptions. We assume that path energy cost is proportional to the length of the path, as for example with uniform-thrust and uniform-speed aircraft. We assume that visibility danger or benefit from a single observer is a utility, positive or negative, also proportional to path length, as with a uniform-speed vehicle that has a uniform probability of being noticed anytime when visible. A positive utility means it is desirable to be seen, a negative utility the opposite. Any path-optimization problem in which cost is proportional to path length within homogeneous regions of space is equivalent to the optics problem of tracing light rays through optical media of locally-homogeneous indices of refraction, as for example a lens system. It follows that Snell's Law applies on the region boundaries in our problem.

The main challenge for our path planning becomes the geometric construction of visibility regions. We assume a fixed set of point observers. We model all obstacle (impermissible-traversal) regions as polyhedra as in [8], methods of which are illustrated in Figure 1, which shows example terrain near Jolon, California (top picture), and one polyhedral fit to it (bottom picture). (To ensure paths stay at least K units from obstacles, we can further offset the polyhedral facets by K.) We also model visibility regions as polyhedra representing regions of space within which all the same things can be seen; they are polyhedra because each point observer and each convex edge of an obstacle determine a plane wherein two facets of two adjacent visibility regions must lie. Figure 2 illustrates the visibility-region construction in cross section. Each region can be assigned a uniform probability of detection per unit traversal distance within it; if more than one observer can see a region, we assume observational independence and the total probability of detection is the inverse of the product of inverses of the separate probabilities of detection. This approach leads to recursive resubdivision of regions, and can be managed by an object-oriented system storing information about regions, facets, edges, and vertices.

This approach is surprisingly general. Work done against gravity can be ignored, since we are assuming a fixed start and goal, and the work between them must be independent of the path. Work against atmospheric resistance to the object traversing the path can also be ignored whenever the resistance is linearly proportional to the velocity. (The traversal speed then need not be constant, since the total work done against air resistance depends only on the path length). The only remaining major factor is turn cost. Since we are doing high-level path planning, we can assume the object traversing the path is negligible in size with respect to the geometry of the path, and we can assume turns are abrupt, not gradual. Then turns at a particular velocity require a momentum-vector change proportional to the sine of half the angle of the heading change. In an atmosphere, a well-designed craft requires minimal energy expenditure to make this momentum change, but in space or thin atmospheres, all this must be done by thrust. Then by using Lagrange multipliers it is straightforward to show the following generalization of Snell's Law must hold for the turn on the boundaries of regions:

$$\mu_1\sin\theta_1 + (K/l_1)\cos\theta_1\cos(|\theta_1-\theta_2|/2)\,sgn(\theta_2-\theta_1) = \mu_2\sin\theta_2 + (K/l_2)\cos\theta_2\cos(|\theta_2-\theta_1|/2)\,sgn(\theta_2-\theta_1)$$

where the μ terms are the cost per unit distance in the two regions per friction and visibility factors, the θ terms are the entering and leaving path angles with respect to the normal to the boundary between the two regions, K is the constant of proportionality between traversal cost and turn cost, and the l terms are the lengths of the path within each of the two regions (recall the path must be linear within these regions). When K=0 this becomes Snell's Law. This equation is almost equally straightforward to apply in "ray tracing" of optimal paths as Snell's Law.



- Volume #1 is visible from Observer #1 and not visible from Observer #2
- Volume #2 is not visible from Observer #1 and not visible from Observer #2
- Volume #3 is visible to both observers
- Volume #4 is visible to Observer #2 and not visible to Observer #1

Figure 2

2. Search methods

Given a start point, a goal point, and a set of polyhedral regions of space homogeneous in traversal characteristics, search for an optimal path breaks naturally into two parts (see Figure 3): finding a good sequence of adjacent volumes through which the path can go, and optimizing the path through those volumes by iterative adjustment. The first step can be done by an A* search using the centers of the regions as nodes, an idea used before for nonoptimal-path finding [1, 2], but which we enhance by requiring Snell's-Law turns on region boundaries. The length of path segment within each volume determines its contribution to the cost of the path. A* search can provide the K best volume sequences, or all sequences with cost within C of the best cost. By a straightforward extension of the two-dimensional case considered in [3, 4, and 5], the optimal path must lie within an ellipsoid whose foci are the start and goal, whose major axis is the distance D between start and goal times the ratio R of the cost of the highest-cost region to the lowest-eost region, and whose minor axes are $D\sqrt{R^2-1}$.

For a sequence of volumes, we must then find the best path through it. Such an optimal path must be piecewise-linear, turning only on region boundaries, and turning only in accordance with Snell's Law

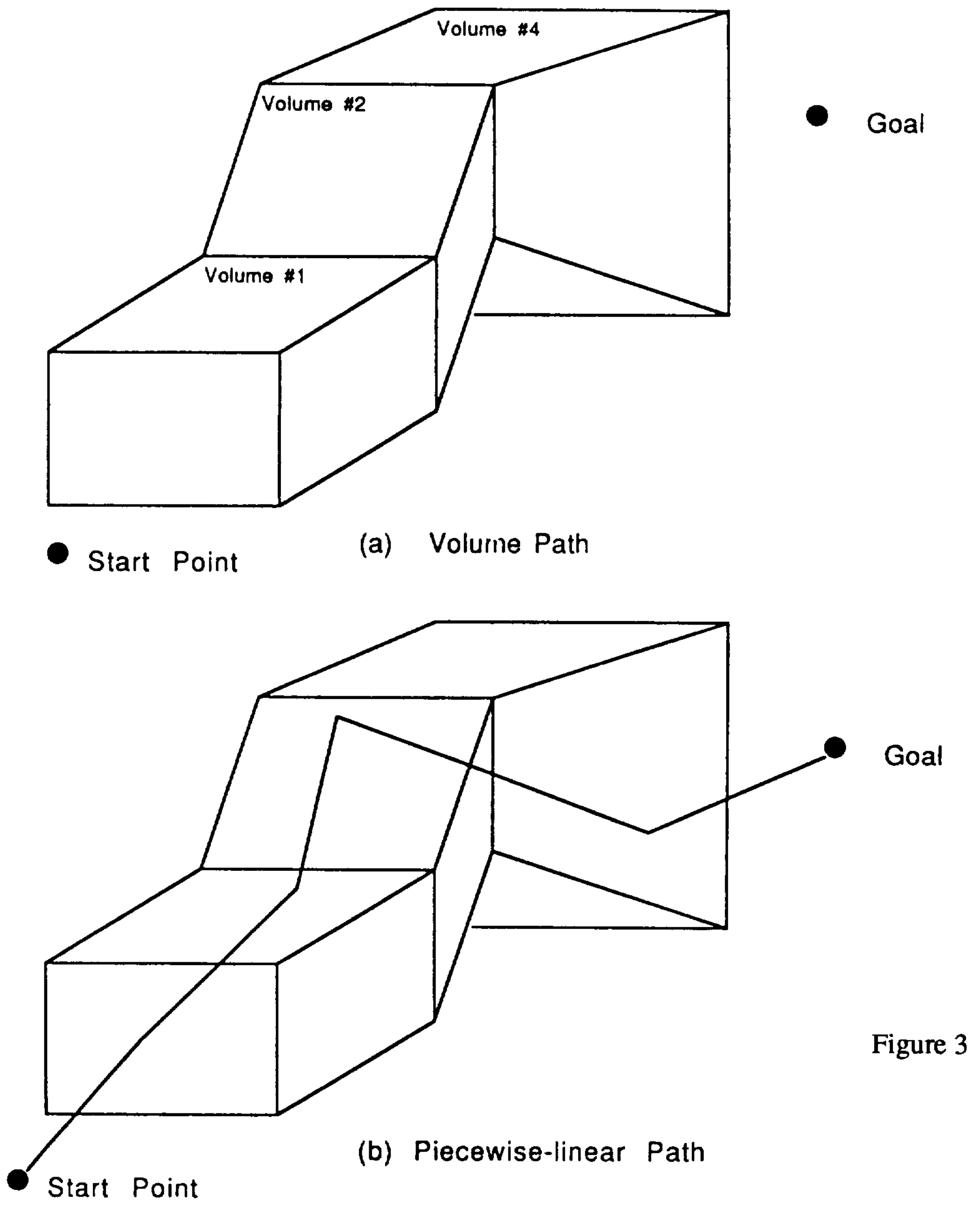


Figure 3

except when the boundary is an obstacle (when it must be a obstacle edge, which follows from the shortcut meta-heuristic). This is a convex optimization problem if we ignore turn cost, as is provable from Snell's Law. Thus almost any iterative optimization technique can be used without fear of converging to the wrong local optimum. (Even with turn cost included, we can use the same techniques heuristically.) The optimization variables represent points on the facets of polyhedra, points with only one degree of freedom each because there are only two cases for path turns: (1) points on obstacle edges, where the turn must enclose the obstacle; and (2) points on non-obstacle-region facets, where Snell's Law must hold for the turn, and thus where all the successive such path segments must be coplanar between obstacle encounters. We can thus partition a path into alternating episodes of successive Snell's-Law turns and successive obstacle-following turns, and each episode can be optimized separately with each variable varying along one dimension. The iteration need not be run to convergence for every volume sequence under consideration, since we may be able to see after a while that the result could not better that of best volume sequence found so far. Lower bounds on costs help for this; they can be obtained by considering the minimum distance between two facets of a polyhedron.

3. Implementation

To explore these ideas we did a partial implementation in Common Lisp with Flavors on a Texas Instruments Explorer II. To simplify path planning, we created more polyhedral visibility regions than strictly necessary (for instance, we subdivided all regions until they were convex, to avoid checking that paths between two facets of a region stay within the region). In the second (optimization) phase, we only examined the best region sequence found by the first search. And to simplify the optimization itself, we did not include turn costs (though we did in the first phase or A* search). Figure 4 is the block diagram of our implementation.

Figure 5 shows solutions to some simple problems given our program. A single observer is stationed in a "box canyon" in the middle of the far side of the terrain, and visibility by this observer is taken as undesirable; the goal point is outside the canyon to the right in the back; and a variety of start points are chosen along the left side of the picture outside the box canyon. While we did not continue the optimiza-

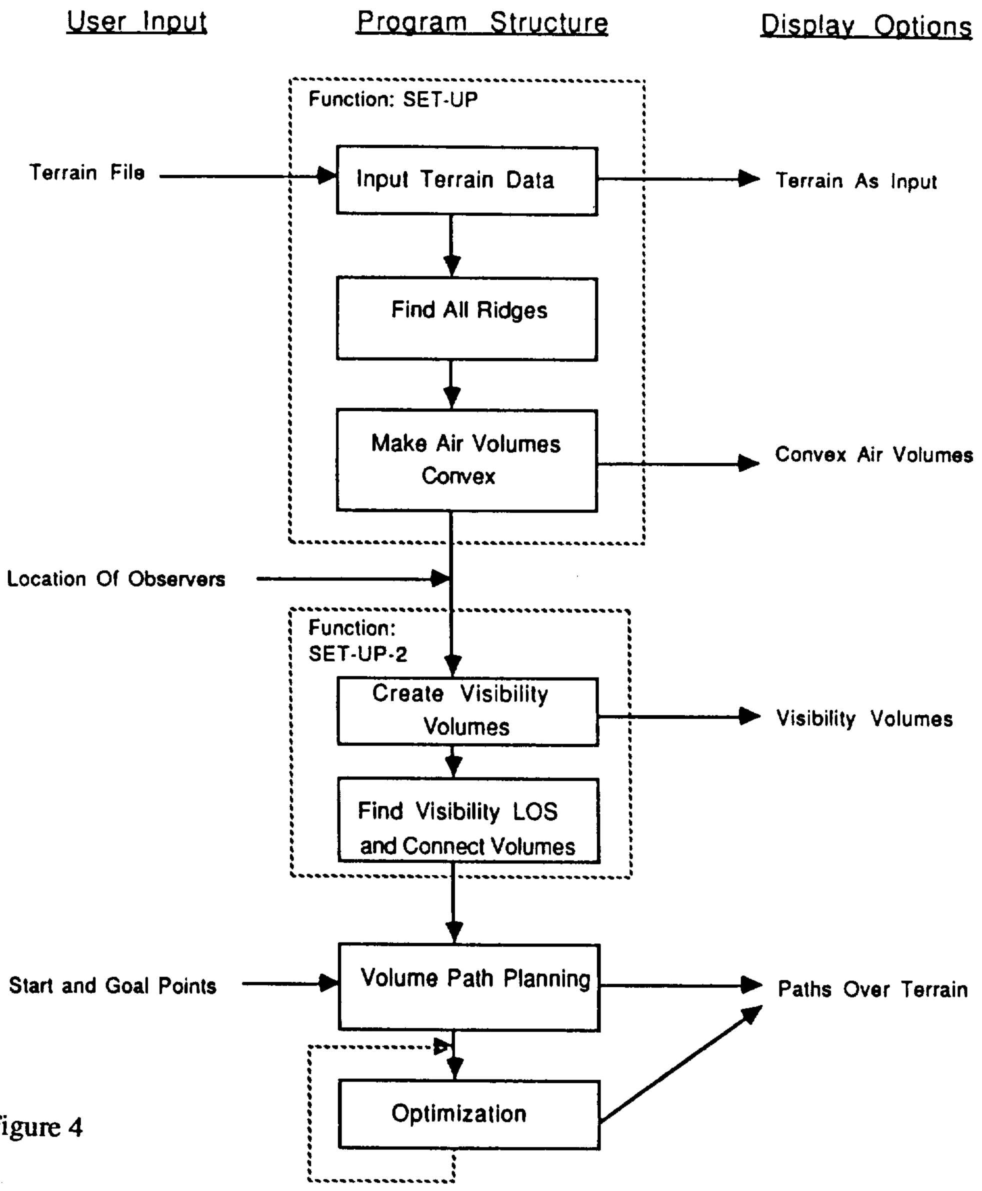


Figure 4

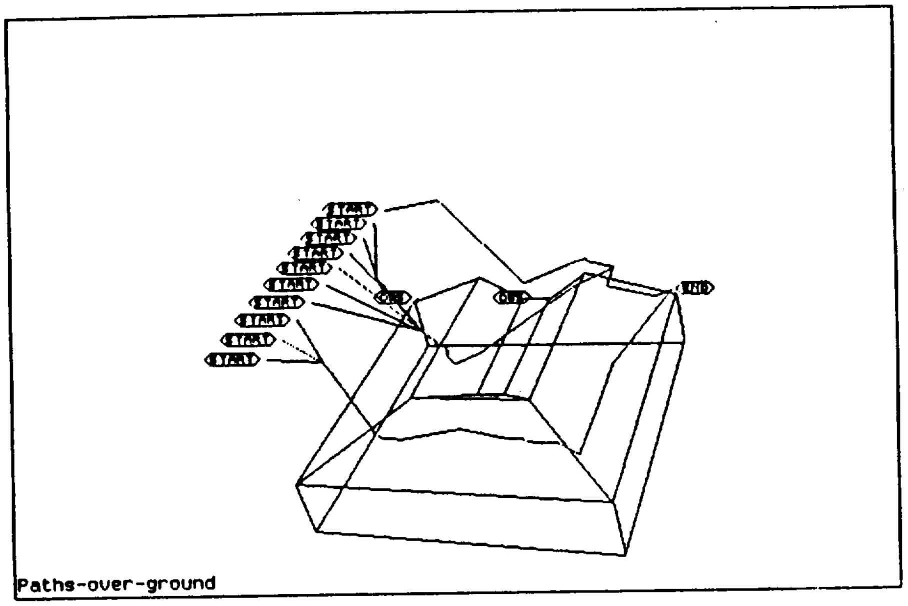


Figure 5

tion long enough to completely smooth out the paths, note that optimal paths starting toward the front stay hidden from the observer, but paths starting toward the back find it better to be seen by the observer for a short time rather than make a long and costly-energy detour.

To see a little of what the program does to get such results, Figure 6 shows the convex air regions created,

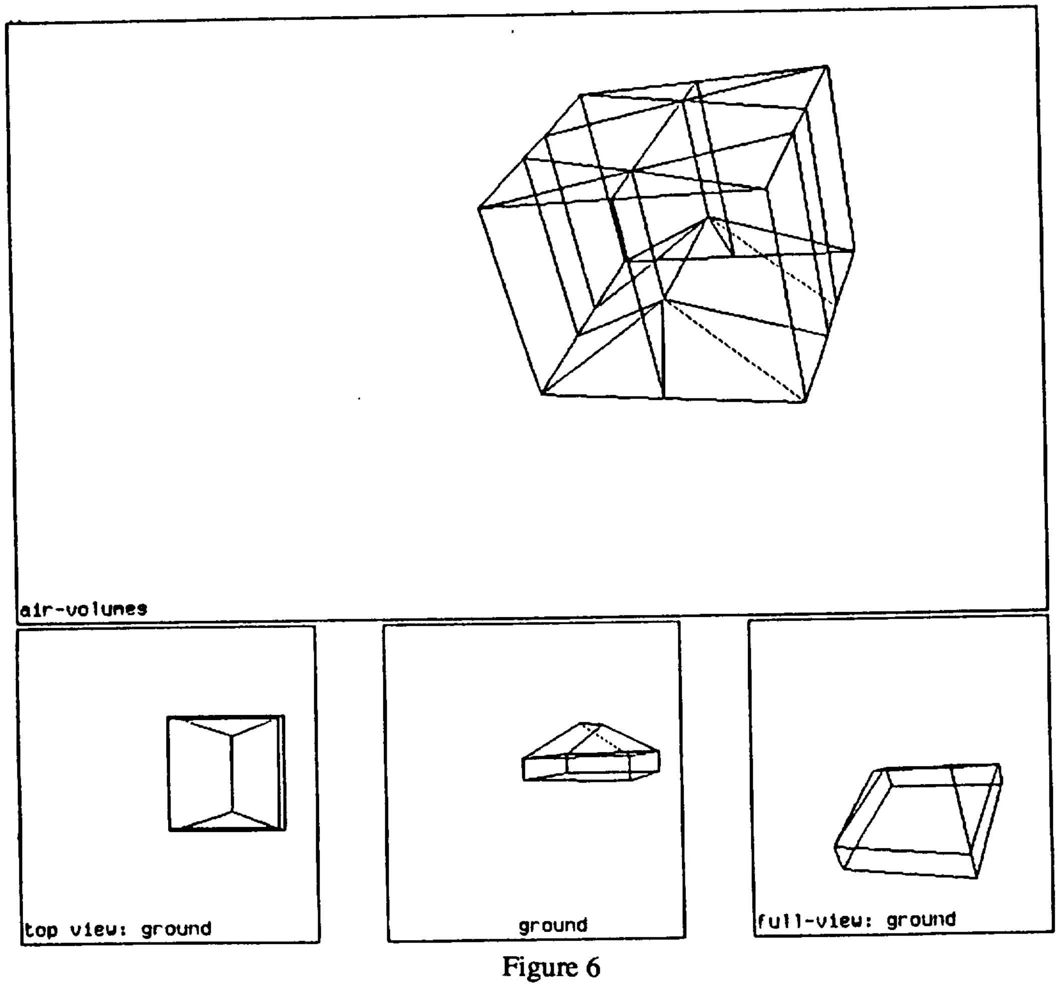


Figure 6

preliminary to visibility analysis, for an obstacle polyhedron representing a hill or ridge. To illustrate the search phases, Figure 7 shows terrain with two ridges, start and goal points near the ground on the far right and the far left sides of the two ridges, and the result of fitting a piecewise-linear path through the centers of the facets found in sequence by the region-sequence (A*) search; Figure 8 is the result after optimization. While the difference was not dramatic, we did average a ten percent improvement in path cost by optimizations in our experiments.

Most of the computation time by far for our implementation was in the computation of the visibility regions. But this could be easily improved significantly by better algorithms; for instance, we use only a very simple method to intersect planes and polyhedra. However, region-computation time may often not be important because regions usually need only be computed once, like the surface of the planet or the surface of a space object; once computed, they can be used repeatedly for all path planning on that same terrain.

4. Conclusions

The work so far is just a first step in what we hope will be a continuing research effort. We have yet to integrate well our terrain planar-patch modeling into the path planning. Clearly we need a more comprehensive path-cost calculation, incorporating atmospheric air resistance, a less-absolute notion of

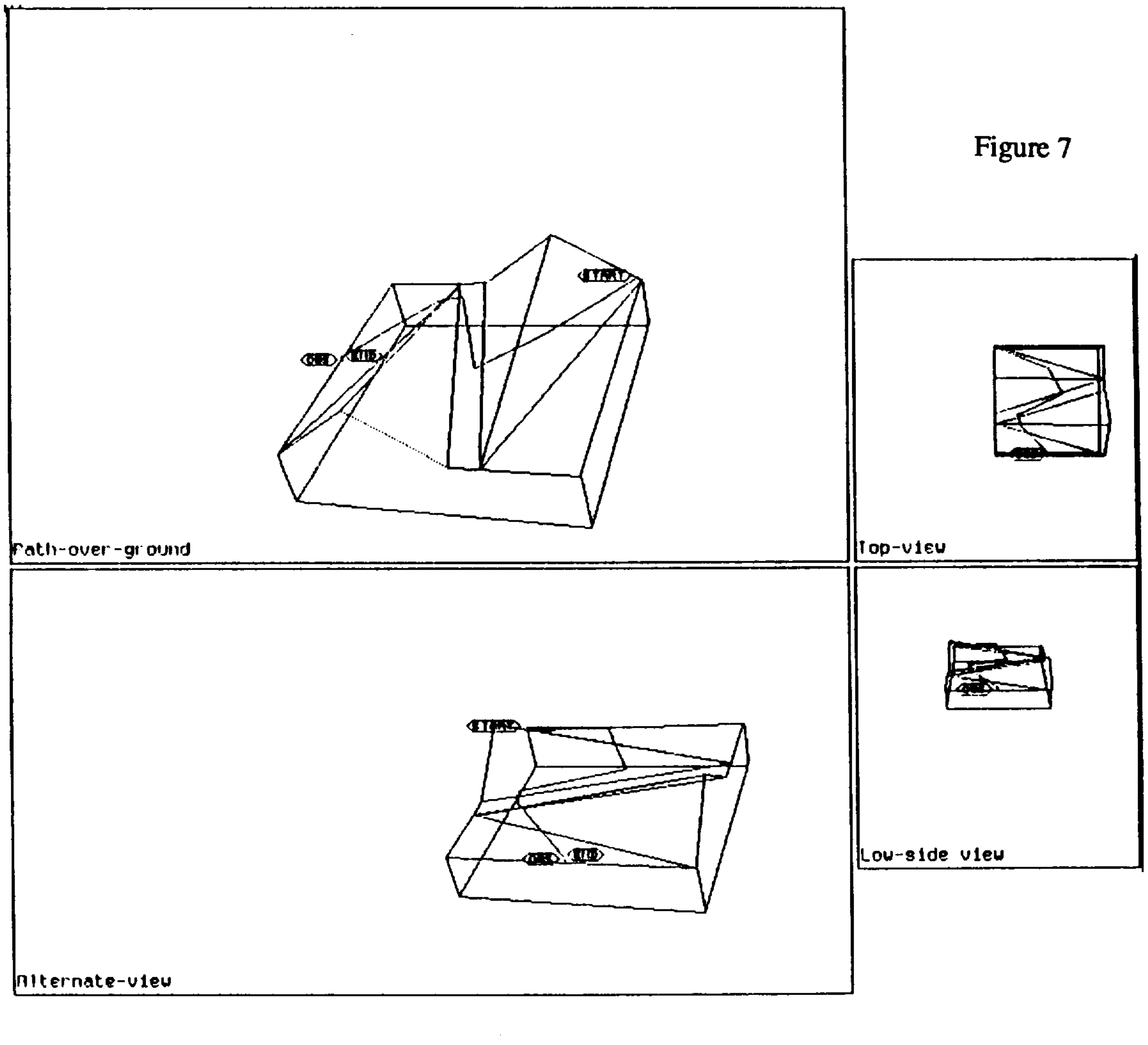


Figure 7

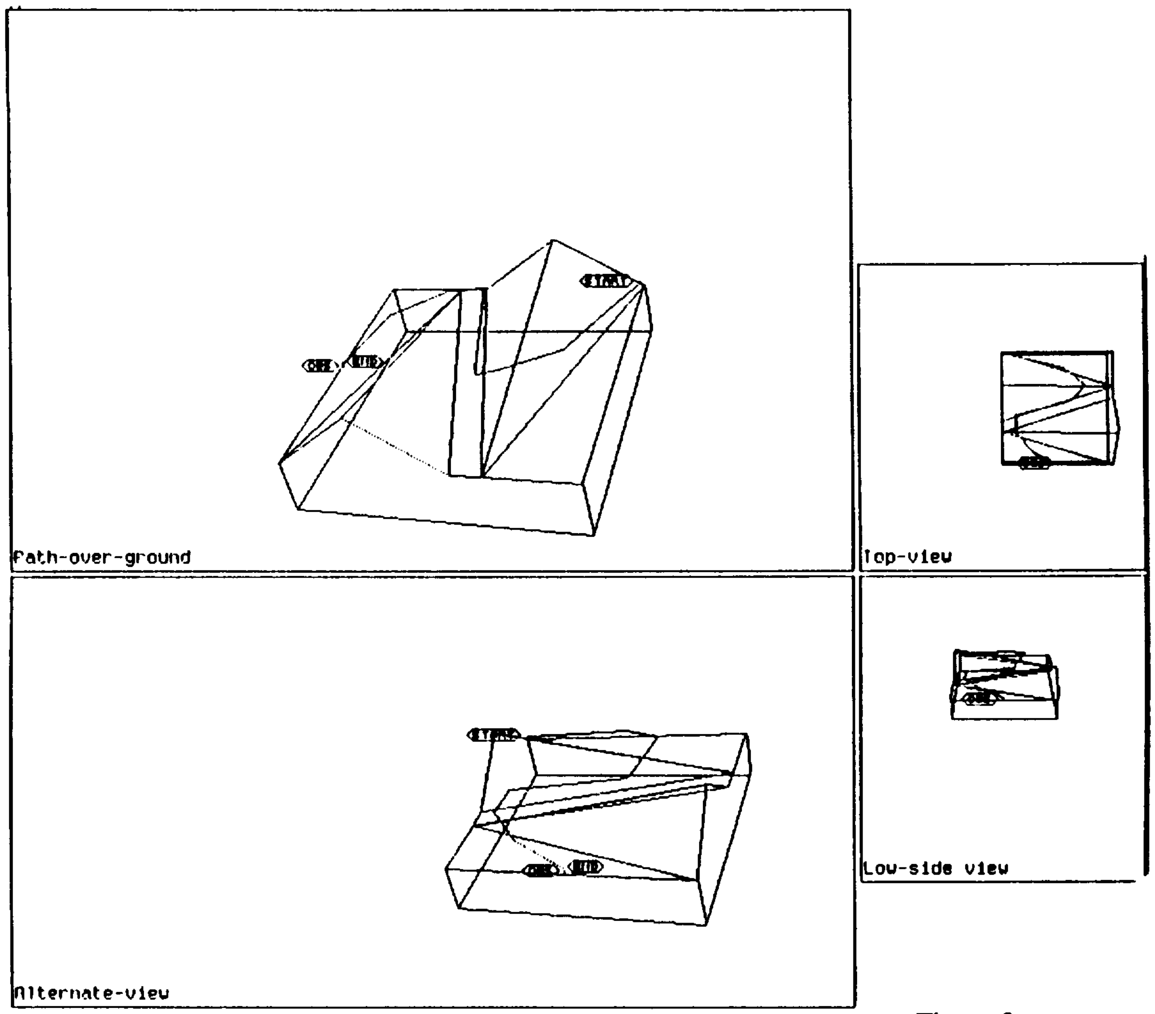


Figure 8

visibility, wind effects, and temperature-variation and other weather effects. Some kinds of aircraft and spacecraft show significant nonlinearity in the tradeoff between power expenditure and thrust, and this needs to be modeled. We intend to explore simple nonhomogeneous regions, for which optimal paths can be curves. Currently we are having problems with numerical errors in long-and-narrow regions, whose centers are misleading, so fixing that problem is a high priority. But the methods we are exploring are more intuitive than current methods, as well as being more efficient for many problems because of the fewer regions of space that need to be created. They clearly deserve further investigating.

Bibliography

[1] R. Chavez and A. Meystel, "Structure of intelligence for an autonomous vehicle", IEEE International Conference on Robotics, Atlanta, Georgia, 1984, 584-591.

[2] K. Rueb and A. Wong, "Structuring Free Space as a Hypergraph for Roving Robot Path Planning and Navigation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9, 2, March 1987, 263-273.

[3] R. Richbourg, "Solving a class of spatial reasoning problems: minimal-cost path planning in the Cartesian plane", Ph.D. thesis, U. S. Naval Postgraduate School, June 1987, published as technical reports NPS52-87-021 and NPS52-87-022.

[4] R. F. Richbourg, N. Rowe, N. Zyda, and R. McGhee, "Solving global, two-dimensional routing problems using Snell's Law and A* search", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, February 1987, 1631-1636.

[5] N. Rowe, "Roads, rivers, and rocks: optimal two-dimensional route planning around linear features for a mobile agent", accepted to *International Journal of Robotics Research*, July 1988.

[6] N. Rowe and R. Ross, "Optimal grid-free path planning across arbitrarily-contoured terrain with anisotropic friction and gravity effects", technical report NPS52-89-003, Department of Computer Science, Naval Postgraduate School, Monterey, California, November 1988.

[7] M. Sharir and A. Schorr, "On shortest paths in polyhedral spaces", *SIAM Journal on Computing, 15*, 1 (February 1986), 193-215.

[8] S. H. Yee, "Three algorithms for planar-patch terrain modeling", M. S. thesis, U. S. Naval Postgraduate School, June 1988.

Acknowledgement: This work was prepared in conjunction with research conducted for the Naval Air Systems Command and funded by the Naval Postgraduate School.

SPECIAL TOPICS

VACUUM MECHATRONICS

Susan Hackwood, Steven E. Belinski and Gerardo Beni

Center for Robotic Systems in Microelectronics
University of California, Santa Barbara

Abstract

The discipline of vacuum mechatronics is defined as the design and development of vacuum-compatible computer-controlled mechanisms for manipulating, sensing and testing in a vacuum environment. The importance of vacuum mechatronics is growing with an increased application of vacuum in space studies and in manufacturing for material processing, medicine, microelectronics, emission studies, lyophylisation, freeze drying and packaging. The quickly developing field of vacuum mechatronics will also be the driving force for the realization of an advanced era of totally enclosed clean manufacturing cells. High technology manufacturing has increasingly demanding requirements for precision manipulation, *in situ* process monitoring and contamination-free environments. To remove the contamination problems associated with human workers, the tendency in many manufacturing processes is to move towards total automation. This will become a requirement in the near future for e.g., microelectronics manufacturing. Automation in ultra-clean manufacturing environments is evolving into the concept of self-contained and fully enclosed manufacturing. At the CRSM we are developing a Self Contained Automated Robotic Factory (SCARF) as a flexible research facility for totally enclosed manufacturing. The construction and successful operation of a SCARF will provide a novel, flexible, self-contained, clean, vacuum manufacturing environment. SCARF also requires very high reliability and intelligent control. In this paper we will review the trends in vacuum mechatronics and discuss some of the key research issues.

1. Introduction

Vacuum mechatronics involves the design and development of vacuum compatible computer controlled mechanisms for manipulating, sensing and testing in a vacuum environment. Vacuum mechatronics is becoming important due to the increased use of vacuum in applications for space studies and manufacturing for material processing, medicine, microelectronics, emission studies, lyophylisation, freeze drying and packaging. As the benefits of the vacuum environment, e.g. low pressure, long mean free path length and cleanliness, become better defined and understood, the desire to implement more processes in vacuum will increase. The vacuum environment is therefore important in many operations requiring a controlled, contamination-free environment.

Vacuum mechatronics plays a particularly important role in the microelectronics industry. Microelectronics manufacturing has increasingly demanding requirements for precision manipulation, *in situ* process monitoring and contamination-free environments. To remove the contamination problems associated with human workers, there is a need to move towards total automation for IC manufacturing. This will become a requirement in the near future as dimensions decrease below 1μm and circuit complexities increase. There is also a trend toward the use of self-contained manufacturing systems since clean rooms are no longer adequate. It has been shown that vacuum, once achieved, is inherently superior to the best clean room environments. Automation in ultra clean manufacturing environments is evolving into the concept of self contained and fully enclosed manufacturing. At the CRSM we are developing a Self Contained Automated Robotic Factory (SCARF) as a flexible research facility for totally enclosed manufacturing. The SCARF system will be used for prototyping application-specific IC's (ASIC's) e.g., 1μm CMOS and NMOS. The construction and successful operation of a SCARF will provide a novel, flexible, self contained, clean manufacturing environment. A self contained manufacturing environment is appealing for IC manufacturing as it allows the implementation of fast cycle times, high yield, low cost and flexible prototyping. It also requires very high reliability and intelligent control. Already, a number of equipment manufacturers have chosen to isolate processes in self-contained vacuum environment manufacturing cells, using small robots as wafer transfer devices (e.g. Applied Materials Precision 5000 Etch, Precision 5000 CVD and 9000 Ion Implanter, Varian 5103 CVD system and M2000 Sputtering System). Such systems take advantage of the superior cleanliness properties of vacuum and indicate the eventual direction of microelectronic (and other cleanliness-intensive) manufacturing.

Many manufacturing steps are understandably dependent upon atmospheric pressure conditions, especially those which presently require an operator. Total in-vacuum manufacturing systems will not be realized unless a concentrated effort is made to develop and *integrate* the vacuum-compatible system components. These include robots, sensors, vision inspection systems, particle detectors and various testing and measuring devices. In the following sections we will discuss some key research problems in vacuum mechatronics and describe ongoing research projects in this area.

2. Vacuum Mechatronics: Scope and Goals

a. Vacuum Mechatronic Applications

Vacuum can be classified into natural (space) and artificial (vacuum chamber). Vacuum, as an environment for various processes, can provide many advantages over an atmospheric environment, such as low particle contamination level, collision-free space, and long monolayer forming time [1]. These properties are currently used in advanced research projects in particle physics, material science, microelectronics, biotechnology, etc. There are opportunities for developing new vacuum systems for these fundamental technologies. However, it is the applications of vacuum mechatronics to manufacturing are becoming interesting. From the time of the first artificial closed vacuum systems, there has undoubtedly been a desire to manipulate objects inside the chamber with as much ease as those outside the closed system.

The space program has provided much of the forward momentum in vacuum mechatronics due to the numerous vacuum problems which had to be solved for space missions [2]. Some of these solutions have recently been applied and extended for use in chamber-based production environments, such as those used for coating (e.g. evaporation or sputtering). In this and other vacuum production applications, the transfer and/or positioning functions provided by the mechatronic equipment is critical to the overall process.

b. Vacuum Mechatronics Design

Mechatronics design for vacuum poses design constraints on the selection of <u>materials</u>, choice of <u>lubricants</u> and on modes of <u>energy transfer</u> [3,4]. Materials should have the standard design properties e.g machinability and ease of fabrication etc., and in addition must have surface vapor pressures lower than the operating pressure and temperature. Desirable physical properties of lubricants for vacuum include low vapor pressure over a wide temperature range, low contamination level and low coefficient of friction. Energy transfer in vacuum needs to focus on heat dissipation and energy input to a mechanism in vacuum. Natural convection is absent in vacuum and thus dissipation must be achieved by conduction, radiation or forced convection.

The effective use of the vacuum environment will depend on the availability of these <u>vacuum components</u>. Mechanisms and machine design research should include joints, bearings, energy transmission/control devices, linkages, fasteners, etc. for vacuum [5,6]. Actuators e.g., vacuum rated motors, piezoelectric devices will need to be developed. The need for and methods of sensing in vacuum (e.g., encoders for vacuum motors, force sensors, vision sensors) will also be needed.

An intelligent controller which can deal with limited sensory information/limited control action possessing fault detection/tolerance capability must be designed for vacuum mechatronics control [7]. Real-time multi-sensory data fusion is desired[8,9]. A computationally very efficient world model is important, because it can be used with active sensing, in working space understanding and model adaptation, as well as in the expectation and sensory data interpretation during operation.

Since the usual teaching method is no longer adequate for vacuum mechatronics, real-time simulation capability is highly desirable to assist program control. Some new criteria for optimal trajectory and task scheduling must be introduced. Reliability is another important issue in vacuum mechatronics, besides component design, emphasis must also be placed on the controller, i.e., fault tolerance ability, since frequent repair is undesirable.

c. System Design and Integration
Although there are many problems inherent in system design common to both atmospheric and vacuum applications, there are problems associated with designing mechatronic systems for vacuum that warrant special attention. Outgassing, heat transfer, and particle emissions are issues that must be addressed in vacuum work [10,11]. Reliability, always a concern when designing mechatronic systems, becomes especially important when the system is enclosed in a vacuum chamber. The overall size of the finished system can be very important in vacuum applications. Often systems must be constructed to fit into existing vacuum chambers; in any case the size of the system and therefore the surrounding chamber must be kept small to keep the costs of the chamber and pumping system down. Another difficulty in designing mechatronic systems for vacuum use is a lack of vacuum compatible subassemblies (e.g. robots, stages, etc.), the building blocks of system design.

3. Vacuum Mechatronics: Current Research Projects

The current research program at the CRSM is focussed in three areas:

1: **INTELLIGENT SYSTEM DESIGN, SIMULATION AND CONTROL**
   - VACUUM-COMPATIBLE ACTUATORS
   - VACUUM ROBOTS
   - SELF-CONTAINED SYSTEMS

2: **SENSORS IN VACUUM**
   - VISION
   - MULTIPLE SENSING SYSTEMS

3: **IN-VACUUM CLEANLINESS AND PARTICULATE CHARACTERIZATION**
   - MEASUREMENT TECHNIQUES
   - MECHANISM TESTING

Several of these research projects will be discussed in more detail below. In particular, the development of vacuum compatible robots, self contained systems, vision and particulate characterization will be described.

3.1 INTELLIGENT SYSTEM DESIGN, SIMULATION AND CONTROL

VACUUM-COMPATIBLE ACTUATORS

New Actuator Design

Application of conventional electric motors in vacuum leads to problems. At high vacuum the gas density is so low that conduction and convection can no longer take place, thermal exchange is carried out mainly by radiation. If power is applied to a motor in vacuum, and no sink is provided, it will heat up until losses due to radiation cause an equilibrium. A temperature of 125°C can be reached in several minutes with the application of the maximum rated voltage to a thermally isolated motor. This problem may be minimized by designing appropriate heat sinking, limiting the voltage necessary to drive the load and reducing or eliminating the holding current when the motor is not running[12]. Even if the temperature effects are controlled, the motor must be constructed of suitable materials and employ appropriate lubrication. The CRSM is cooperating with Yaskawa Electric to develop motors specifically for high-vacuum robot applications. They have been developing an axial gap pulse motor, which will withstand temperatures to 300°C and vacuum levels of 10^{-11} Torr[13].

Magnetically Levitated Systems for Clean Vacuum Operation

Magnetically levitated systems have great potential for vacuum applications[14]. Lack of surface contact in such devices can reduce the particle load significantly.

Motion Control for In-Vacuum Motors

Some unique considerations exist with respect to the control of in-vacuum motors. Due to the lack of conduction through air and convection in a vacuum, optimized temperature control is desirable. Also, the currently available vacuum motors are of the stepper motor variety, making feedback control and smooth motion difficult for precision actuators and robots.

VACUUM ROBOTS

Vacuum Robot Development for Industrial Manufacturing

A robot capable of operating in high vacuum (to 10^{-7} Torr) has been developed for ultra-clean manufacturing of gyroscopes in a self contained manufacturing environment. This was a two year effort in collaboration with Delco Systems Operations. The availability of vacuum-compatible robots is presently limited, although this is likely to change in the near future[15]. A modified commercially available robot was used for use in the assembly task[16]. Although it is desirable to use a robot which was designed and built specifically for the vacuum environment, the first step was to obtain a vacuum-compatible robot.

The vacuum robot is a GMF model E-310 cylindrical coordinate robot, originally designed for use in clean rooms to class 10. The principal design requirements for the modification of the GMF E-310 robot for vacuum compatibility were:

•Modification of axes movement range:
-Z-axis: maintain 300mm stroke if possible
-R-axis:maintain 500mm stroke if possible; if reduced, resulting stroke must be useful in the vacuum chamber
-q-axis: maintain ±150° rotation
-a-axis: maintain ±180° rotation
•Limit negative effects on the vacuum environment (outgassing, etc)
• Design for ≤ 100°C operating environment

The first decision in the modification of the GMF E-310 was between two methodologies. The robot could either be totally exposed to the vacuum environment or it could be sealed in a type of "suit" which would allow the inside components to operate at atmospheric pressure, as they were originally designed to do. In order to expose the entire robot to a pressure of 10^{-8} Torr, a number of key changes would have to be made. The major ones would be in the lubrication systems, the surface finish and materials, and the motors. After examining this choice, it was concluded that it would entail a substantial amount of redesign work, and that a total exposure robot would be better designed from scratch. The goal then became one of designing a new housing for the robot which would seal it from the vacuum environment, while accomplishing the design goals. The sealing "suit" would have to be as leak-tight as the walls of a high-quality vacuum chamber, yet must also allow the desired motions by sealing two linear (R and Z) and two rotary (Theta and Alpha) motions. The completed robot is shown in Figure 1.

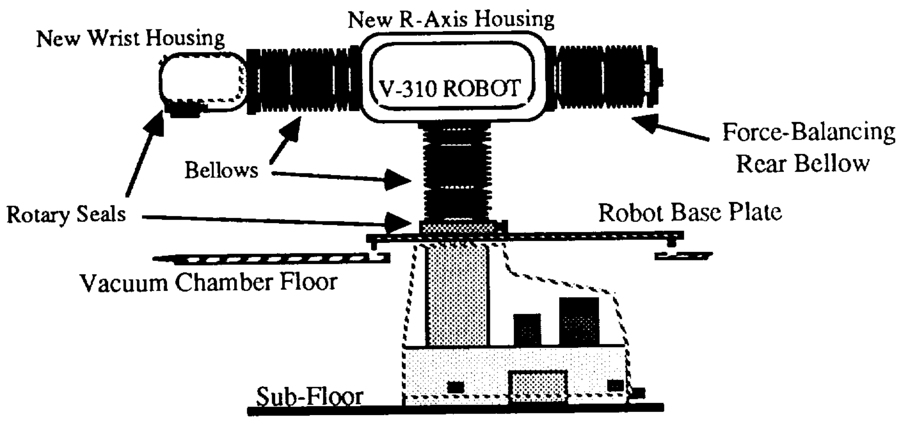


Figure 1. Modified E-310 Vacuum Robot

SCARF Vacuum Robot and Controller Development
The modified GMF vacuum robot described above is useful, but is not ideal. A robot designed to be fully exposed to the vacuum is more difficult to build but has greater implications for vacuum mechatronics. The CRSM, in cooperation with Yaskawa Electric, has designed and built a vacuum-compatible robot for use in the SCARF vacuum chamber (Figure 2). The robot has many advanced features not currently found in the small vacuum-compatible pick-and-place robots used in microelectronic processing stations. Some key features are:

- The robot is of cylindrical coordinate design, with a linear reach axis. This configuration is inherently suited to a *cylindrical* vacuum chamber.
- The robot's stepper motors are completely vacuum-compatible and use vacuum-compatible magnetic encoders. This eliminates the need for any motion feedthroughs, which are potential leak sources. It also allows for a significant vertical stroke (120mm) which is missing in other vacuum robots due to the sealing problems of a linear feedthrough.
- The controller is based on the Motorola 68020 processor and the TMS320 digital signal processor, and fully programmable in a high level Pascal-like language.
- The controller is easily interfaced to a host computer. The robot then falls under the authority of the overall system controller, easing system integration.

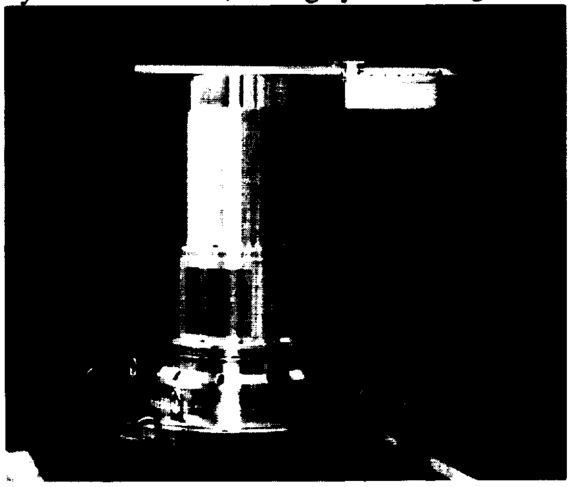


Figure 2. SCARF Robot

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

Basic specifications are as follows:

	TRAVEL RANGE	RESOLUTION	REPEATABILITY	MAX SPEED
S-axis: (base rotation)	360°	.013°	±.013°	90°/s
Z-axis: (vertical stroke)	120mm	0.1mm	±0.1mm	60mm/s
H-axis: (horizontal stroke)	657.66mm	0.25mm	0.25mm	250mm/s
W-axis: (off-robot wafer rotation)	360°	0.25°	±0.25°	90°/s
Payload:	0.4kg			
Vacuum Compatibility:		• Vacuum-compatible to 10^{-7} Torr • Total leak rate less than $5x10^{-9}$ Torr liters/s He • Bakeable to 100°C		

Table 1. SCARF Robot Specifications

SELF-CONTAINED SYSTEMS

Vacuum Mechatronics in the IC Processing Environment.

The semiconductor industry is rapidly evolving to produce the high variety and short cycle times demanded by its customers. Application Specific Integrated Circuits (ASIC's) are proliferating [17,18]. As the demands for flexibility increase, the fabrication process sequences themselves are becoming longer with more levels and complexity. Dimensions and design rules are expected to be reduced below 0.5 μm in the next few years. The corresponding allowable particle sizes (using the one-tenth rule) are less than 500Å. Not only can we not directly measure these sizes, but present day clean rooms have approximately a $1/d^2$ law for particle densities vs. particle sizes [19] and therefore very large densities of small particles cannot be avoided by using currently designed airborne clean room systems.

It was clear even in the early 80's that an integrated manufacturing capability would be needed by the microelectronics industry [20]. By early 1987, several equipment manufacturers already displayed self-contained stand alone process tools that are fore-runners of larger tool integration yet to come. Drytek (General Signal) and Applied Materials Technology market dry etch and Chemical Vapor Deposition (CVD) equipment, respectively, that are single-wafer-at-a-time tools with multiple process chambers and thus multiprocess capability. Also MTI-Sypher has now marketed a unit with combined deposition (2 stations) and etching (1 station). The wafers are fed by robots and these tools suggest tool architectures for the further evolution of integrated processes.

Factories of the future will have facilities architecture where cells are linked together. If the operations needed to make an entire integrated circuit are combined under the envelope of one unit tool, then we ultimately have a self-contained factory. If the wafers are transported by automation and robotic manipulation, controlled by a computer, we have a self-contained-automatic-robotic-factory (SCARF)[21]. Many large companies have embarked on similar paths. IBM [22] and Texas Instruments [23] have similar programs.

SCARF System Description

The SCARF project was initiated at the CRSM in mid 1987. We are essentially placing the clean room inside a relatively small envelope, evacuating that envelope, maintaining low particle densities and controlling pressure to quickly allow transfer and load locking between wafer storage areas and process chambers. A specific implementation has been designed, as shown in Figures 3 and 4. A large number of IC fabrication processes are currently being performed in vacuum. The

SCARF design integrates small footprint vacuum tools together around the central chamber. It is convenient to bring certain process tools together locally, especially those which will be used serially in the process architecture.

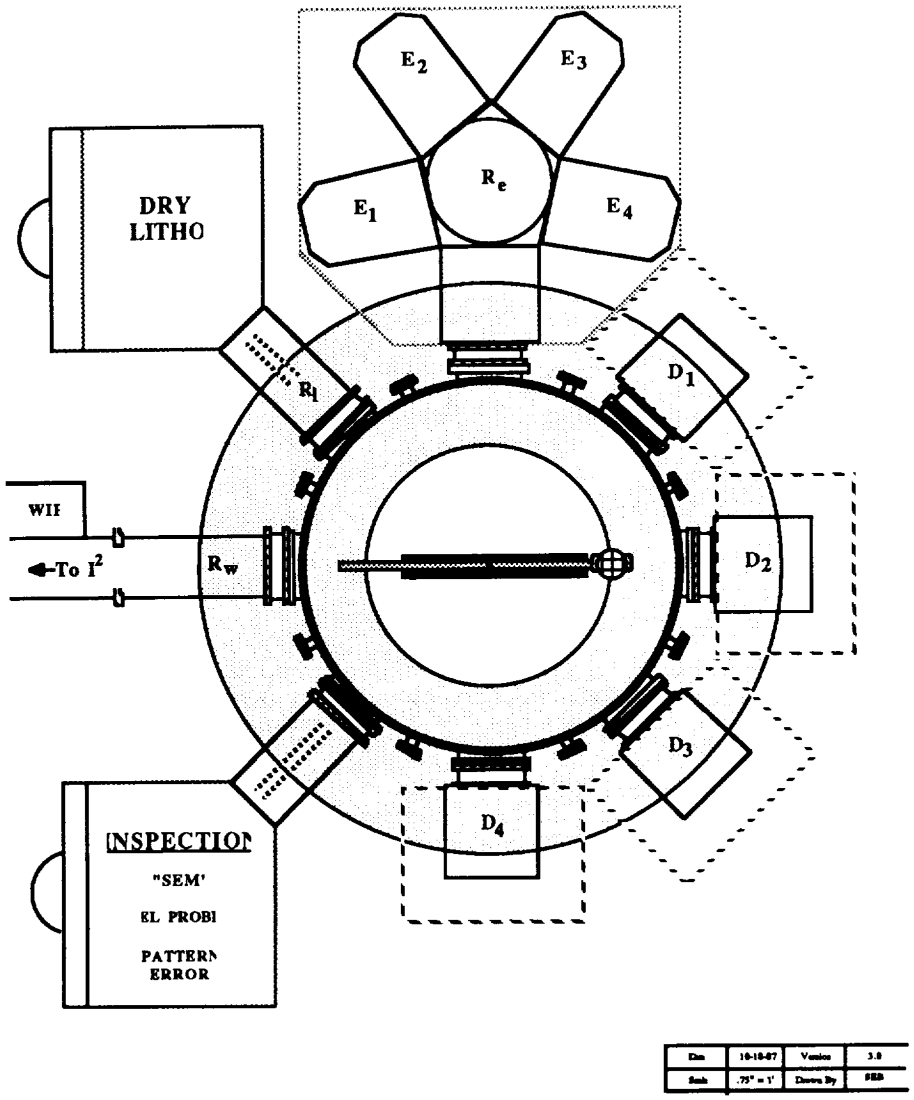


Figure 3. Self-Contained-Automated-Robotic-Factory Layout.
The four deposition chambers in the SCARF are dedicated to a specific process or at least dedicated to a compatible class of chemicals. The central vacuum system has a pumping system that allows base pressures of 10^{-6} Torr. Both rough and controlled limited pumping as well as rough and controlled venting are required for the system. It is important to be able to equalize the pressure between low pressure process chambers and the central vacuum chamber in order to avoid particle transport between chambers.

The chamber is now completed and testing is progressing. The operational parameters of the SCARF facility dictated the design of the central vacuum chamber. The chamber is 50 inches in diameter to provide room for several processing tools around its circumference. There are eight ports around the circumference of the chamber to attach wafer processing equipment. Seven view ports, four on the top and three on the bottom, provide for *in situ* inspections. A 24 inch diameter port on the bottom of the chamber allows quick access to the robot used for wafer transportation.

The entire lid of the chamber is removable to provide greater access to the interior of the chamber. Eighteen small Conflat ports allow electrical and mechanical feedthroughs to be quickly attached to the chamber. The mechanical design of the central vacuum chamber provides the flexibility required in a research environment.

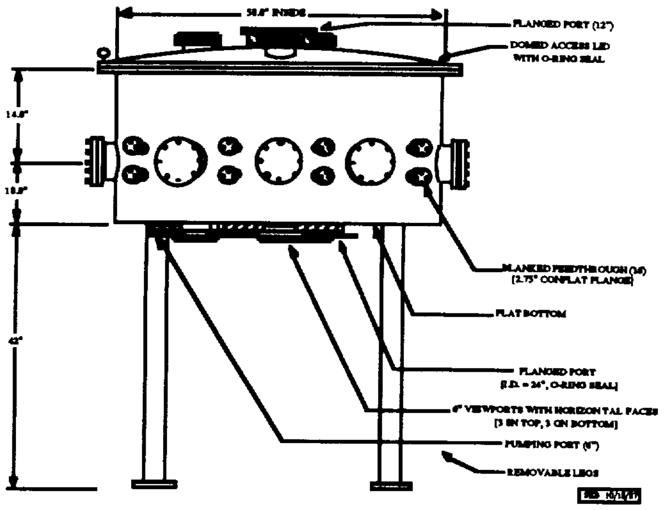

58.8" INSIDE
FLANGED PORT (12")
DOMED ACCESS LID WITH O-RING SEAL
14.0"
10.0"
42"
BLANKED FEEDTHROUGH (18) [2.75" CONFLAT FLANGE]
FLAT BOTTOM
FLANGED PORT [I.D. = 24", O-RING SEAL]
6" VIEWPORTS WITH HORIZONTAL FACES [3 ON TOP, 3 ON BOTTOM]
PUMPING PORT (8")
REMOVABLE LEGS

Figure 4. Cross Section of SCARF Chamber

SCARF System Integration
The SCARF system falls under the control of a central host, presently a SUN 3/110 workstation. The SCARF Host Controller is responsible for control and monitoring of the SCARF Chamber functions: pumpdown and vent cycles and rates, gate valve and load-lock sequencing, and acquisition of data from pressure gauges. The next level of control involves the SCARF Robot and the in-vacuum particulate monitor. The SCARF Host Controller can act as a terminal for the SCARF Robot Controller during program development, and will communicate with the SCARF Robot Controller during the test phase and actual process runs. In addition, the SCARF Host Controller will be responsible for data analysis and acquisition. It will serve as the loop control when clean load-locking, transfers and processing steps are accomplished using information fed back from in-situ particle detection. Control over in-vacuum vision inspection tasks is also planned.

Intelligent Operation
Self-contained manufacturing environments are generally characterized in having: a) reduced accessibility and visibility in a crowded workspace making operation by an external operator difficult; b) even when the visibility is possible, access is often costly as it requires exposing the internal environment to atmosphere; c) the work environment is often hazardous. These characteristics require the system to depend on sensors to achieve higher autonomy. The operations must also have a robustness to process variation. Operations such as robot motion within this environment therefore require the development of algorithms for automatic planning of motion so that smoothness can be achieved (to avoid particle generation) and so that obstacles can be avoided[7]. The smooth collision-free trajectory control is required for many mechanisms.

3.2 SENSORS IN VACUUM

VISION
In-Vacuum Color Vision Inspection

As more processes are integrated into a vacuum environment manufacturing system, in-situ inspection will also be required. By performing the inspections in the same vacuum station rather than transferring the wafers to a standard inspection station in clean room, the chance of fatal contamination can be dramatically reduced. Color vision has high potential for process monitoring, metrology and control in IC manufacturing. The increased complexity and decreased lateral and vertical dimensions of semiconductor circuits necessitates accurate, reliable process monitoring. Computer vision, i.e. automated optical inspection, is an important component of automated process inspection and monitoring [24]. Recently, we have designed and built a color vision workstation suitable for automated inspection of integrated circuits [25]. The workstation can readily identify defects that could not be distinguished by black and white processing, even by using gray scale imaging [26]. Furthermore, semiconductor fabrication is in large part a thin film technology. Not only are some materials intrinsically colored, but optical interference effects of semi-transparent layers give films a color characteristic of the film thicknesses.

Color vision can therefore be used in inspection for isolating defects not normally visible in black and white processing. In addition, we have used the relationship between film thickness and color to show the feasibility of a system that can rapidly (~100 milliseconds) measure thin film thicknesses to approximately 20Å accuracy [27]. This can be done by use of a color matching scheme or by incorporating analytical relationships that allows identification of samples of unknown oxide thickness.

Robot Positioning via End-Point Detection in Vacuum
As totally enclosed vacuum processing systems for microelectronics become more advanced, the repeatability with which wafers can be placed for processing becomes a more critical issue. Currently, robots of various sizes and configurations are being used as transfer mechanisms to move wafers between processing stations, with repeatability of placement determined by either motor-mounted encoders or stepper motor drive systems. However, the usual uncertainty of placement position is accentuated greatly in a vacuum chamber, due to the slightly changing shape of the chamber and movement of target areas with respect to each other and the robot over time. To overcome this, it is necessary to implement an end-point feedback system for wafer positioning in the process or inspection chamber.

3.3 IN-VACUUM CLEANLINESS AND PARTICULATE CHARACTERIZATION MEASUREMENT TECHNIQUES

Understanding particle behavior and contamination control in vacuum interface technology is critical to the progress of vacuum-based processes[28]. The dynamic measurement of particles generated during a vacuum operation has to date been difficult to accomplish. The recently developed PM-100 particle monitor made by High Yield Technology is a new type of particle counter and is presently the only one that can be used under vacuum[29]. The system includes a sensor head, a preamplifier, and a controller, and has some unique features. This unit measures particle flux through a light net, which gives information on particle motion as well as the number of particles flowing through during a certain time interval so that real-time monitoring is easily achieved. Sensors such as this are key to monitoring particulate counts in self-contained manufacturing processes.

The probabilistic behavior of this sensor have been studied[30]. The measuring mechanism can be modeled by a Poisson stochastic process with the particle flux to be measured as a parameter of the distribution function. Based on this model, the probability of counting error is estimated. It is shown that when the actual particle flux is significant, the probability of counting error becomes very high. When the product of particle flux and sampling time is small, this probability is approximately a second order function of the sampling time. This sensor, while very useful, gives an intrinsic error in the total particle count. A Bernoulli experiment model can be set up and the formula for recovering the actual total particle count derived.

The cleanliness characteristics of the vacuum environment has been investigated through the use of load-lock chambers and vacuum-compatible particle monitors[31]. It has been demonstrated that most particles will occur at the beginning stage of the rough pumping when the air flow is the maximum and turbulence is expected. The particle count has been related to the turbulence through the time dependent instantaneous Reynolds number. Experimental results indicate a strong relationship between particle count and Reynolds number.

An unexpectedly large number of particles are counted at the rough pumping stage when the chamber is backfilled with clean room ambient air. A nucleation hypothesis proposes that during pumping, the moisture in the air will tend to condense onto fines, and the presence of turbulence will trigger and enhance the condensation process, causing the fines to quickly grow into particles of supermicron sizes[32]. Backfilling with dry nitrogen has led to a dramatic reduction in particle count, although before pumping nitrogen has a very similar particle distribution to that of the clean room air.

In summary, this work characterizes the cleanliness level of vacuum and uses real-time particulate information to minimize contamination levels. Results indicate that vacuum, once achieved (to 10^{-3} Torr and higher) is clearly superior to clean room technology. In general, since there is no air to support particle flow, only newly generated particles will be detected.

4. Conclusion

The problems of clean, contamination free vacuum environments, where complex processes are performed, monitored and verified without human intervention, are not limited to space applications. The microelectronics industry, materials processing, biotechnology and pharmaceutic manufacturing are all tending towards the same direction. In order to produce these new technologies, new manufacturing strategies have to be sought. In particular, the concept of modular, self contained intelligent manufacturing systems offers the possibility of coping with complex processing tasks with high reliability. As many of these manufacturing processes are carried out under vacuum, the design and development of computer controlled mechanisms for manipulating, testing and sensing in this environment become necessary. Vacuum mechatronics is a rapidly developing field of research aimed at solving some of the problems.

Acknowledgements

We would like to thank Tom Seidel who helped develop the concept of SCARF. We also thank the CRSM Vacuum Mechatronics group. This work was partially supported by NSF, contract number CDR8421415. The color vision work was sponsored the Semiconductor Research Corporation, contract number 88-MP-121.

References

[1] A. Roth, "Vacuum Technology", North-Holland Publishing Co., 1976.
[2] H. R. Ludwig, "Six Mechanisms used on the SSM/I Radiometer", 19th Aerospace Mechanisms Symposium, NASA CP-2371, pp. 347-362, May 1-3, 1985.
[3] S. Belinski and S. Hackwood, "Manufacturing in a Vacuum Environment", IEEE International Electronic Manufacturing Technology Symposium, p.269, Anaheim, Oct. 12-14, 1987.
[4] K. G. Roller, "Lubricating of Mechanisms for Vacuum Service", Journal of Vacuum Science and Technology A 6 (3), pp. 1161-1165, May/June, 1988.
[5] T. J. Patrick, "Space Environment and Vacuum Properties of Spacecraft Materials", Vacuum, vol. 31 (8), pp. 351 - 357, 1981.

[6] P. V. Head, W. Allison and R. F. Willis, "Specimen Manipulators for High Resolution in Ultra-high Vacuum", Vacuum, vol. 32 (10/11), pp. 641 - 644, 1982.
[7] B. Paden, "Robot Motion Planning for Self-Contained Manufacturing Systems", Proc.Vacuum Mechatronics International Workshop, University of California Santa Barbara, Feb 2-3, 1989.
[8] E. Hu, S. Magiarcina, M.Peters, S. Hackwood, G. Beni, "Inference in Intelligent Machines," in Proc IEEE Conf. on Robotics and Automation, p.1966, San Francisco, Apr. 7, 1986, p1966
[9] S. Mangiarcina and G Beni, "On the Logical and Physical Combinations of Evidence in Intelligent Machines," J. Intelligent Systems, Vol 1, 1986, p143
[10] T. Hatsuzawa, Y. Tanimura, H. Yamada and K. Toyoda, "Piezodriven Spindle for a Specimen Holder in the Vacuum Chamber of a Scanning Electron Microscope", Rev. Sci. Instrumentation, 57(12), pp. 3110-3113, 1986.
[11] J. O'Hanlon, "Advances in Vacuum Contamination Control for Electronic Materials Processing", Journal of Vacuum Science and Technology, A5(4), p. 2067, July/Aug. 1987.
[12] S. Belinski, W. Trento, R. Imani and S. Hackwood, "Robot Design for a Vacuum Environment", Proceedings of the NASA Workshop on Space Telerobotics, vol 1, p.95, Pasadena, Jan. 20-22, 1987.
[13] S. Nio, T. Suzuki, H. Zenpo, K. Yokoyama, H. Wakizako and S. E. Belinski, "Vacuum Compatible Robot Design for Self-Contained Manufacturing.
[14] T. Matsumoto, "Magnetic Levitation for Vacuum-Compatible Wafer Transfer Systems", Proc.Vacuum Mechatronics International Workshop, University of California Santa Barbara, Feb 2-3, 1989.
[15] S. Hackwood, "Automated Manufacturing in a Vacuum Environment", Proc. 21st Aerospace Mechanisms Symposium, NASA Johnson Space Center, Houston, April/May 1987.
[16] M. Shirazi, "Development and Testing of a Vacuum-Compatible Robot", accepted for publication in ASME Manufacturing Review.
[17] S. Runyon, "The Great ASIC Wave Gathers Force", Electronics, 60, 58 (1987).
[18] J.G.Toetch, "Super Integration: Using Standard Products as Megacells," in VLSI System Design, June 1987, p.752.
[19] B. Hardigan and A. Lane, "Testing Particle Generation by Wafer Handling Robots," Solid State Technology, March 1985.
[20] H. H. Schicht, "Clean Room Technology: the Concept of Total Environmental Control for Advanced Industries", Vacuum, Vol. 35, No. 10/11, pp. 485-491, 1985.
[21] S. Hackwood, "Robotics in Microelectronics Manufacturing", Proc. IEEE International Workshop on Intelligent Robots and Systems, Tokyo, Japan, Oct. 31 - Nov. 2, 1988.
[22] J. G. Bednorz, et al, U.S. Pat. 4,643,627: "Vacuum Transfer Device", Feb 17, 1987.
[23] G. R. Larrabee, private communication, and J.M.Blasingame, "Microelectronic Manufacturing Science and Technology", Air Force Wright-Patterson, PRDA 87-7 PMRR
[24] K. L. Harris, et al., "Automated Inspection of Wafer patterns and Applications in Stepping, Projection and Direct Write Lithography," Solid State Technology, Feb. 1984, p.159.
[25] M. Barth, J. Wang, S. Parthasarathay, E.L.Hu, S. Hackwood, and G. Beni, "A Color Vision System for Microelectronics: Application to Oxide Thickness Measurements", Proc.IEEE Int. Conf. on Robot. and Auto., San Francisco, CA., April 1986, pp.1242-1247.
[26] S. Parthasarathay, D. Wolfe, S. Hackwood, E.L. Hu, and G. Beni,"Color Vision for Microelectronics Inspection", Proc. SPIE Conf. on Intelligent Robots and Computer Vision, vol 726, pp. 125-130, Oct. 26-31, 1986.
[27] S. Parthasarathay, D. Wolf, E.L. Hu, S. Hackwood, and G. Beni, "A Color Vision System for Film Thickness Determination", Proc IEEE Int. Conf. on Rob. and Auto, p.515, Raleigh, NC, Mar. 31-Apr.3, 1987.
[28] R. A. Bowling and [9] G. R. Larrabee, "Particle Control for Semiconductor Processing in Vacuum Systems", Microcontamination Conference Proceedings, Canon Communications, Inc., Santa Monica, CA pp161-168, 1986.
[29] P. G. Borden, Y. Baron and B. McGinley, "Monitoring Particles in Vacuum-process Equipment", Microcontamination, pp. 30-34, Oct., 1987.

[30] Degang Chen, T. E. Seidel, S. Belinski, and S. Hackwood, "Dynamic Particulate Characterization of a Vacuum Load Lock System", submitted for publication.
[31] Degang Chen, S. Belinski ans S. Hackwood, "Effect of Moisture Condensation on Particle Count During Pumpdown", submitted for publication.
[32] Degang Chen and S. Hackwood, "Vacuum Particle Dynamics and the Nucleation Phenomenon During Pumpdown", in progress.

N90-29855

UNIFORM TASK LEVEL DEFINITIONS FOR ROBOTIC SYSTEM PERFORMANCE COMPARISONS

Charles Price, Johnson Space Center, NASA
Delbert Tesar, University of Texas at Austin

ABSTRACT

The following is an initial effort to tabulate a series of 10 task levels of increasing difficulty on which to make comparative performance evaluations of available and future robotics technology. It is not certain that these 10 levels are sufficient or that they are in the correct order. Further, the specific detailed parameters of the tasks will have to be carefully outlined (perhaps by a workshop) by the affected community. These parameters have to do with relative scales, size of obstacles, size of task boards, oscillations of the task board support (frequency and amplitude), disturbance levels in process tasks, etc.

Note that each level has a breakdown of 10 additional levels of difficulty to provide a layering of 100 levels. It is assumed that each level of task performance must be achieved by the system before it can be appropriately considered for the next level.

Obviously, the community will wish to react to this as a point of departure. Nonetheless, something of this nature is necessary to drive the technology forward. Note that questions of mobility, portability, etc. are implied but perhaps should be set aside as additional requirements depending on the implied task regime. For example, zero G would be such a task regime as would 50 G, but they are so unique as to require a special community response.

Some members of the community would like to bypass or step over some of the task levels. This could be achieved by using the notation of 8.3 - 4, meaning that the system was proven at level 8.3 but all levels 5 through 7 were untested. Also, the community might like an up-to-10 level designation such as 1.6, 2.3, 3.4, 4.7, 5.4, which would imply testing within each listed task level to the decimal level of difficulty.

1. EXISTING STANDARD TASK BOARD SCALE (20% of Work Volume)

   1.0 Simple Placement
   1.1 Low Tolerance Peg-in-Hole
   1.2 High Tolerance Peg-in-Hole
   1.3 Bayonet Lock Assembly
   1.4 Toggle Lock Clamping Device
   1.5 Screwing of Nut on Bolt
   1.6 Force Fit Disassembly
   1.7 Force Fit Assembly
   1.8 Pin Connector Assembly
   1.9 Bending of Tube to Shape

2. ENLARGE TASK BOARD TO TAKE UP MOST (80%) OF ROBOT WORK VOLUME INCLUDING SPACES WITH SINGULARITIES

   2.0 Simple Placement
   2.1 Low Tolerance Peg-in-Hole
   2.2 High Tolerance Peg-in-Hole
   2.3 Bayonet Lock Assembly
   2.4 Toggle Lock Clamping Device
   2.5 Screwing of Nut on Bolt
   2.6 Force Fit Disassembly
   2.7 Force Fit Assembly
   2.8 Pin Connector Assembly
   2.9 Bending of Tube to Shape

3. TASK BOARD WITH SEVERAL PERPENDICULAR SURFACES WITH TASKS ON ALL SURFACES

   3.0 Simple Placement
   3.1 Low Tolerance Peg-in-Hole
   3.2 High Tolerance Peg-in-Hole
   3.3 Bayonet Lock Assembly
   3.4 Toggle Lock Clamping Device
   3.5 Screwing of Nut on Bolt
   3.6 Force Fit Disassembly
   3.7 Force Fit Assembly
   3.8 Pin Connector Assembly
   3.9 Bending of Tube to Shape

4. INCLUDE MOVEMENT OF THE TASK BOARD

   4.0 Small Sinusoidal Motion (1 DOF)
   4.1 Small Sinusoidal Motion (2 DOF)
   4.2 Variable (medium scale) Sinusoidal Amplitudes
   4.3 Random Small Motions (1 DOF)
   4.4 Random Small Motions (1 DOF and 2 DOF)
   4.5 Random Medium Scale Motion (1 DOF)
   4.6 Random Medium Scale Motion (2 DOF)
   4.7 Random Medium Scale Motion with Superimposed Small Shocks
   4.8 Large Scale Motion As in Floating in Ocean Currents or on Ocean Waves ($\omega <$ time scale)
   4.9 Random Large Scale Motion

Small	- 1% of scale
Medium	- up to 5% of scale
Large	- up to 20% of scale
Sinusoidal	- $\omega \geq 5\text{x}$ time scale

5. TRACKING OF PRESCRIBED SPATIAL PATHS

   5.0 Point-to-Point Tracking in Discrete Steps
   5.1 Straight Line Following
   5.2 Straight Line Following Along Major Workspace Dimension
   5.3 Circular Arc Following
   5.4 Circle Tracking With Round Out

5.5 Conic Section Curve Following
5.6 Flat Plane Following in 80% of Workspace
5.7 Spherical Surface Following in 80% of Workspace
5.8 Tracking Error of 0.1% of Scale
5.9 Tracking Error of 0.01% of Scale

6. OCCLUSION-VISUAL ACCESS IS DEGRADED

6.0 5% Visual degradation
6.1 15% Visual degradation
6.2 25% Visual degradation
6.3 35% Visual degradation
6.4 45% Visual degradation
6.5 55% Visual degradation
6.6 65% Visual degradation
6.7 75% Visual degradation
6.8 85% Visual degradation
6.9 95% Visual degradation

7. OBSTACLE STREWN ENVIRONMENT

7.0 One spherical obstacle in the environment, 10% of workspace in size
7.1 One spherical obstacle in the environment, 30% of workspace
7.2 One spherical obstacle in the environment, 50% of workspace
7.3 Straight line obstacle through workspace
7.4 Cylindrical obstacle with diameter 10% of workspace scale
7.5 Sphere of 10% and cylinder of 10% scale in workspace
7.6 Two spheres and two cylinders of 10% scale in workspace
7.7 Two spheres and two cylinders at 10% scale in workspace moving at 0.1 time scale of needed task performance time in 10% scale range of motion
7.9 Robot must reach through a ring of 20% scale of the workspace within 20% scale of the task objective
7.9 Robot must reach through a ring of 20% scale of the workspace within 50% scale of the task objective

8. OPERATION IN POTENTIAL FIELD FORCES

8.0 No external forces acting on robot structure
8.1 Effect of stream forces on end-effector - 10% of payload
8.2 Effect of stream forces on end-effector - 50% of payload
8.3 Effect of stream forces on end-effector - 90% of payload
8.4 Stream forces acting on all robot links - 10% effective payload
8.5 Stream forces acting on all robot links - 50% effective payload
8.6 Stream forces acting on all robot links - 90% effective payload
8.7 Base excited inertia forces acting on robot - 10% effective Payload
8.8 Base excited inertia forces acting on robot - 50% effective payload
8.9 Base excited inertia forces acting on robot - 90% effective payload

9. MULTIPLE ARM OPERATION

9.0 One arm fixed as holder
9.1 Both arms equal in performance
9.2 One arm 5 times greater in payload
9.3 Relative forces between end-effectors 10% of payload of weakest
9.4 Relative forces between end-effectors 50% of payload of weakest
9.5 Relative forces between end-effectors 90% of payload of weakest
9.6 Relative positioning precision 2% of intersecting workspace scale
9.7 Relative positioning precision 1% of intersecting workspace scale
9.8 Relative positioning precision 0.1% of intersecting workspace scale
9.9 Relative positioning precision 0.01% of intersecting workspace scale

10. DISTURBANCE REJECTION FROM THE PROCESS

10.0 Sinusoidal force at end-effector of 1% of payload (n=10)
10.1 Sinusoidal force at end-effector of 5% of payload (n=3)
10.2 Sinusoidal force at end-effector of 25% of payload (n=2
10.3 Sinusoidal force at end-effector of 50% of payload (n=1)
10.4 Random force disturbance at end-effector of 1% of payload
10.5 Random force disturbance at end-effector of 5% of payload
10.6 Random force disturbance at end-effector of 25% of payload
10.7 Random force disturbance at end-effector of 50% of payload
10.8 Sudden shock of 10% of payload
10.9 Sudden shock plus random force disturbance

Sinusoidal - ω > nx time scale

N90-29856

Linear Analysis of a Force Reflective Teleoperator

Klaus B. Biggers Stephen C. Jacobsen Clark C. Davis

Center for Engineering Design
University of Utah
Salt Lake City, Utah

Abstract

Complex force reflective teleoperation systems are often very difficult to analyze due to the large number of components and control loops involved. This document describes one model of a force reflective teleoperator and presents an analysis of the performance of the system based on a linear analysis of the general full order model. Reduced order models are derived and correlated with the full order models. Basic effects of force feedback and position feedback are examined and the effects of time delays between the master and slave are studied. The results show that with symmetrical position-position control of teleoperators, a basic trade off must be made between the intersystem stiffness of the teleoperator, and the impedance felt by the operator in free space.

1. Introduction

As man continues to expand into the extreme and dangerous environments of space and undersea, as well as having an increasing requirement to perform tasks in man-made hazardous environments such as nuclear reactors, it has become obvious that there is a need for systems which allow the manipulation of objects and effecting of the environment from a remote location. These systems range from mobile systems with limited ranges of motion, degrees of freedom and sensory capabilities to highly dextrous, force-reflecting, multi-arm and hand systems with a high degree of sensory capability. These more complex systems are often called teleoperation or telepresence systems. In unstructured environments in which the tasks to be performed are not known *a priori*, the manipulation of objects and the execution of complex tasks have shown themselves to be very formidable problems. While the concepts and basic theories involved in teleoperation systems have been investigated and studied for decades, there is still no highly dexterous, high performance system capable of reliably performing complex manipulation or assembly operations. And there are still many more barriers to overcome before reliable, robust, and effective systems are technologically and economically feasible.

2. The "Stick" Analogy for a Single-Degree-of-Freedom Teleoperator

What is the basic behavior we are trying to reproduce in the implementation of an actuated (active) force reflective teleoperation system? If we abstract the problem to that of the single-degree-of-freedom linear case, we are have simplified the problem to that depicted in Figure 1. This figure shows a simple "stick". This can be thought of as the most simple mechanical telemanipulator (1 DOF, linear, mechanically coupled). The behavior we are trying to reproduce when we separate the master and slave and attempt to actively couple them with actuation systems is the behavior of the single molecular layer (δx) between the two mechanically coupled systems. These desired behaviors include:

- low operator input impedance in free space
  - inertia
  - viscous drag
  - friction

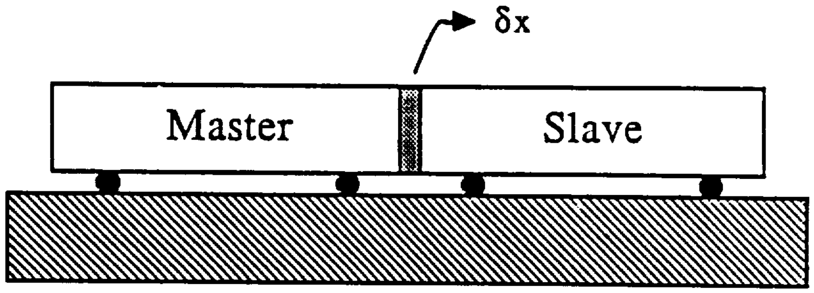


Figure 1: The simple passive stick teleoperator

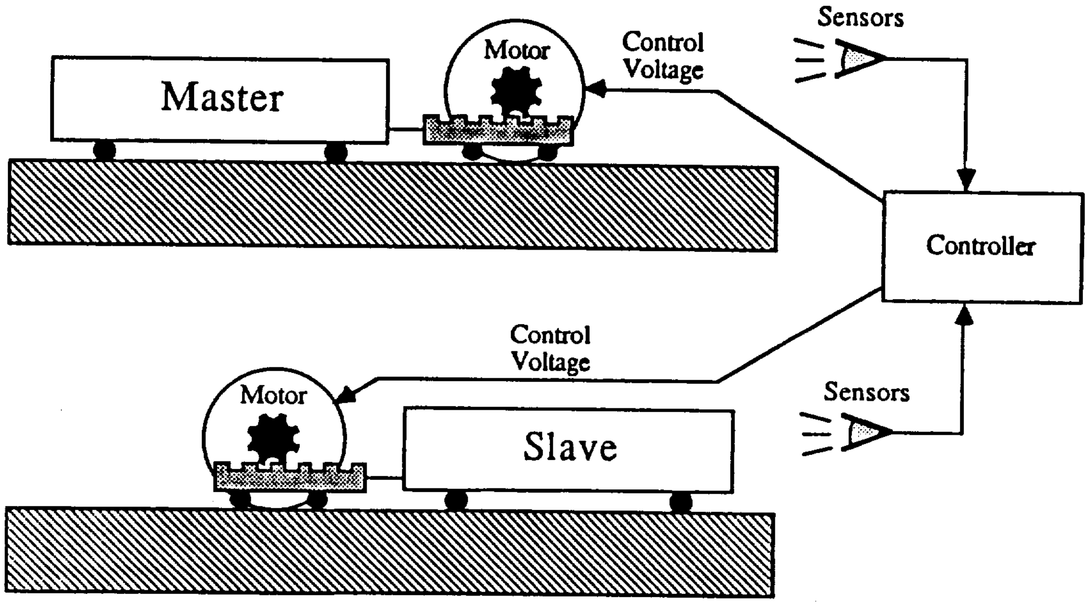


Figure 2: The active "stick" model of a simple one-degree-of-freedom teleoperator

- high intersystem stiffness
- high bandwidth force reflection
- stability for a wide range of contact impedances

These behaviors are some of the most difficult to actively implement in any effector system and reveal the severe demands placed on high performance teleoperation systems. These desired behaviors, however, can lend great insight into the performance and characteristics required of the individual components which comprise the system. This research will use these desired behaviors to determine the necessary characteristics which actuators, structures, sensors, and controllers must exhibit in order for the teleoperation system to perform as desired.

If the two ends of the "stick" are separated and an actuation system, controller, and sensors are integrated in each part, the model becomes that shown in Figure 2. Note that the basic subcomponents of the teleoperator are two symmetrical effector systems. In order to understand the behavior of such a system, we must first fully understand these effector subsystems which make up the teleoperator.

3. The Effector Model

Since a force reflecting teleoperator is really two effectors which have been connected via control systems to behave as a single system, the basic element which we must consider and evaluate in depth is the individual

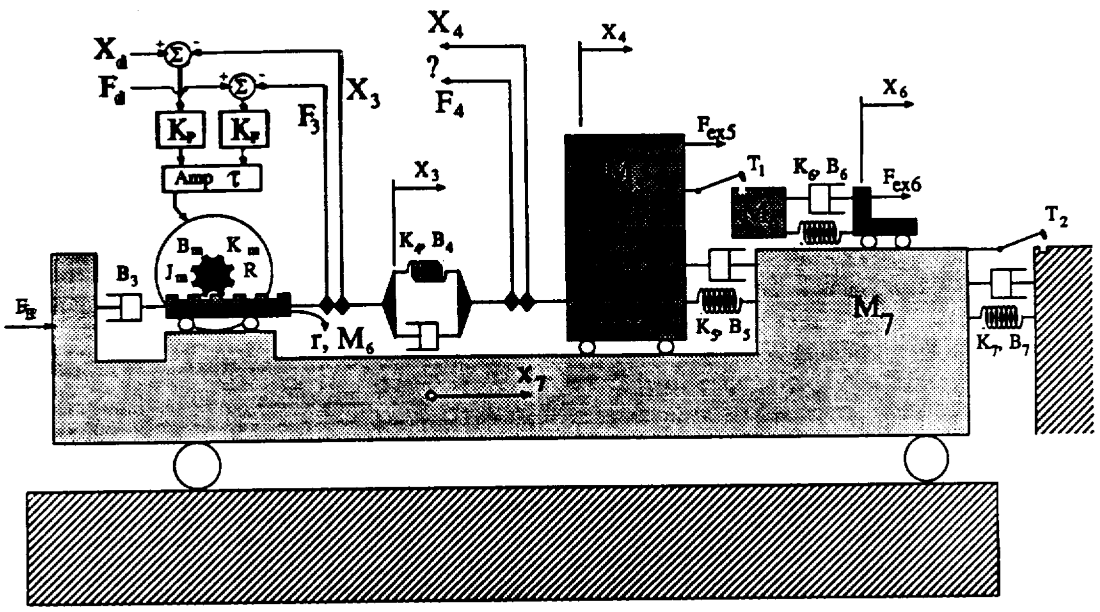


Figure 3: The general model of a linear effector

effector. The effector model we choose to analyze must be complex enough to exhibit what is observed in reality, yet it must be simple enough to allow intuitive understanding of the observed behaviors. The basic effector model which will be used in this study will be the model used in [6] and depicted in Figure 3. This model is presented as a fifth order system but reduced order second and third order simplifications are possible to describe system behavior in specific applications. The fifth, second, and third order models are derived from this general model by setting specific parameters as described in [6]. See [6] for a detailed description of how this may be accomplished.

This model can be configured as a bilateral teleoperation system by properly connecting the control systems and sensors. Then the effects of the various system components on overall teleoperator performance can be analyzed. Also, by using different control schemes in the connection of the two systems, the performance of the different control strategies can be rigorously studied and insight will be gained as to how a force reflecting teleoperator is to be properly controlled.

3.1 The Full Order Effector Model

By setting the parameters as shown in Table 1 [6], the simple fifth order effector model is generated. Two typical root locus plots for this configuration are shown in Figure 4. [1] Figure 4(A) shows a plot in which the force gain (K_f) is set to zero and the position gain (K_p) is varied from 0 to infinity. Figure 4(B) shows the same system with the position gain fixed at 10,000 volts/newton while the force gain is varied from 0 to infinity. Note that as the position gain is first increased, the poles which arise due to the interaction of the actuator and the structure migrate toward the imaginary axis becoming very much less damped. The lower magnitude poles represent an oscillation of the the load and the actuator. This interaction occurs at a lower frequency and while these poles do become less damped and increase in magnitude, they do not cross the imaginary axis, but approach the open loop zeros due to the damping within the structure. Note from Figure 4(B) that as the force gain is increased with a fixed position gain, the poles due to the actuator/load interaction become less damped and decrease in magnitude indicating a "softening" of the actuated system. The high frequency complex poles due to the actuator/structure interaction continue to move toward the imaginary axis, becoming less damped and eventually going unstable. The softening of the low frequency poles gives an indication of the backdriveability of the system. As these poles move toward the real axis, the system becomes more free and compliant and will more easily be "pushed around" by a disturbance. These

[1] Note that the scale of the real and imaginary axes are markedly different in order to more easily see system behavior. Care should be exercised when attempting to extract exact frequency or damping ratio information from the plots. Note also that in many of the plots, the high frequency poles and zeros on the negative real axis are not shown on the plots since thier magnitudes are so great in comparison to the other frequencies of the system's components.

K_m	51.9×10^3	$N\cdot m/a$	Motor[†]
R	.955	Ω	Motor[†]
J_m	38.1×10^{-6}	$kg\cdot m^2$	Motor[†]
B_m	17×10^{-6}	$N\cdot m\cdot s/rad$	Motor[†]
L	0.188	mH	Motor[†]
τ	0	s	Amplifier
G	108.7	$m^{-1}\rightarrow(1/r)$	Reducer
B_3	3.46	$N\cdot s/m$	Drive
M_3	0	kg	Drive
B_4	3.464	$N\cdot s/m$	Structure
K_4	2.83×10^5	N/m	Structure
M_5	2.65	kg	Load
B_5	17.32	$N\cdot s/m$	Load
K_5	113.2	N/m	Load
M_6	0	kg	Touch Load
B_6	0	$N\cdot s/m$	Touch Load
K_6	0	N/m	Touch Load
T_1	0		Touch Load
K_p	*Variable*	V/m	Controller
K_f	*Variable*	V/N	Controller
K_d	*Variable*	$V\cdot s/m$	Controller
Sf, Sp	*Unity*		Sensors
M_7, B_7, K_7, T_2	*Fixed*		Base

[†]Pitman Corporation Motor, Model 5113, winding #1 [1]

Table 1: The parameters used in the general model.

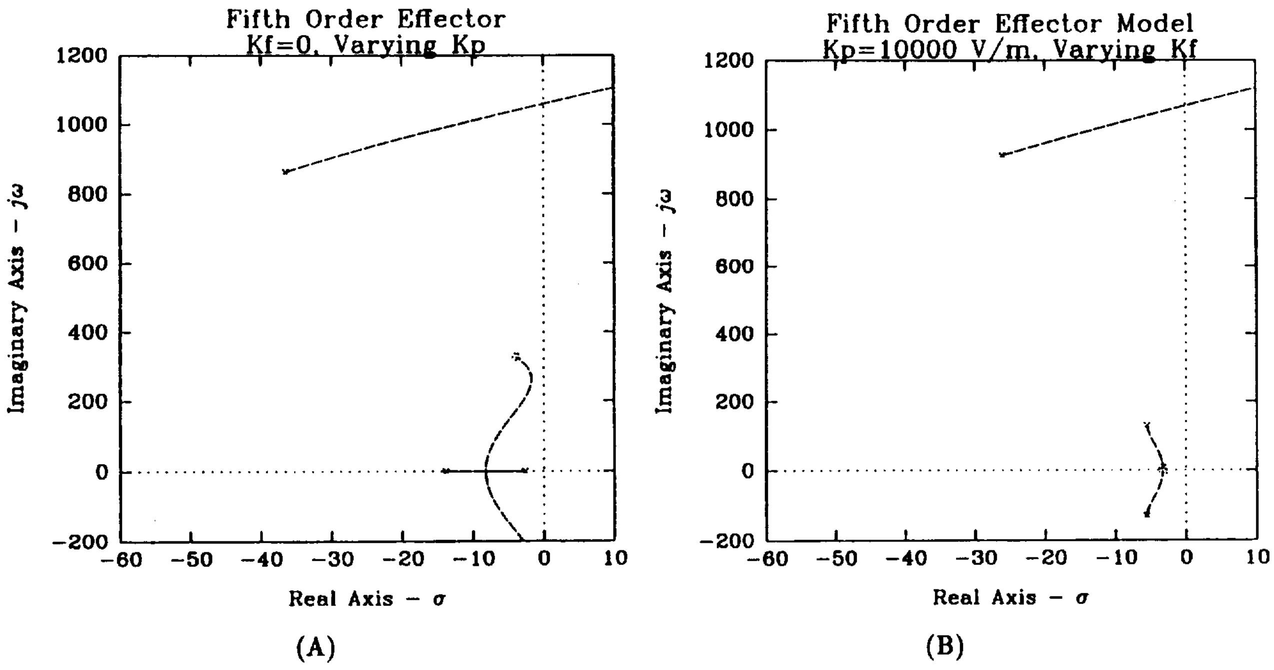
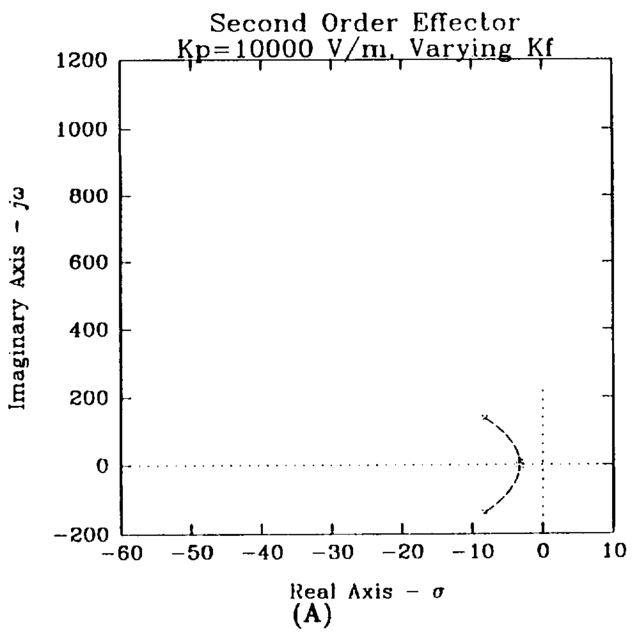


Figure 4: Root locus plots of the fifth order system. (A) shows the root locus with K_f set to 0, varying K_p. (B) shows the plot for $K_p = 10000V/N$, varying K_f.



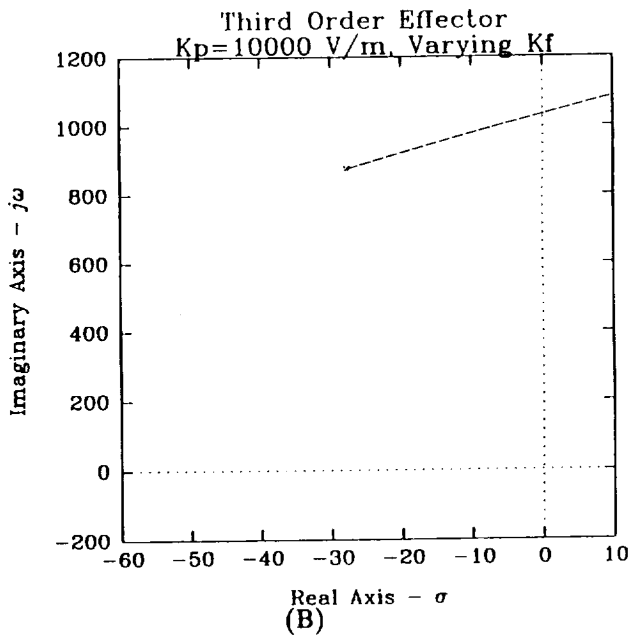


Figure 5: (A) is the root locus plot of the second order system varying K_f, K_p = 10000 V/m. (B) shows the same root locus for the third order system

poles approach the open loop zeros which would be the performance attained if the actuators became pure force sources with no acceleration or velocity dependent impedances.

3.2 Reduced Order Models

The fifth order effector model can be simplified for specific applications or to study specific interesting interactions. These reduced order models are described in detail in [6] and will only be reviewed for clarity here.

Rigid Structure with Moving Load and No Motor Inductance - If we assume that the structure is very rigid (K_4 *large*) and the motor inductance is zero ($L = 0$), the model becomes a second order system used to examine interactions between the actuator and load. Figure 5(A) show a typical root locus for this system with $K_f = 0$, varying K_p.

Compliant Structure with a Slow Load - If we assume that the load mass (M_5) is large and its motions are slow with respect to the dynamics of the actuator/structure interactions, the load mass can be assumed fixed and a third order model is generated which focuses on interactions of the actuator and structure. Figure 5(B) shows a typical root locus with K_p constant , varying K_f.

These simplified models show that we can use restricted generality models to understand specific behaviors of the full order system in particular situations. These lower order models allow a more intuitive understanding of the interactions among system elements and often make closed form solutions for particular performance criteria [6] possible.

4. The Teleoperator Model

4.1 The Tenth Order Teleoperator Model

By connecting two effector models, we can derive a model for a force reflecting teleoperation system. In this paper, we will restrict our analysis to a symmetrical position/position control law. In this method of control, the actual position of the master is used as the desired position for the slave(with no delay), and vice-versa. This control scheme leads to a tenth order model for the force reflective system. Figure 6 shows a simplified block diagram of the entire system.

Figure 7 shows two typical root locus plots corresponding to those in Figure 4. Note that the migration of the varying poles are very similar to those of the fifth order system, however the poles migrate more rapidly

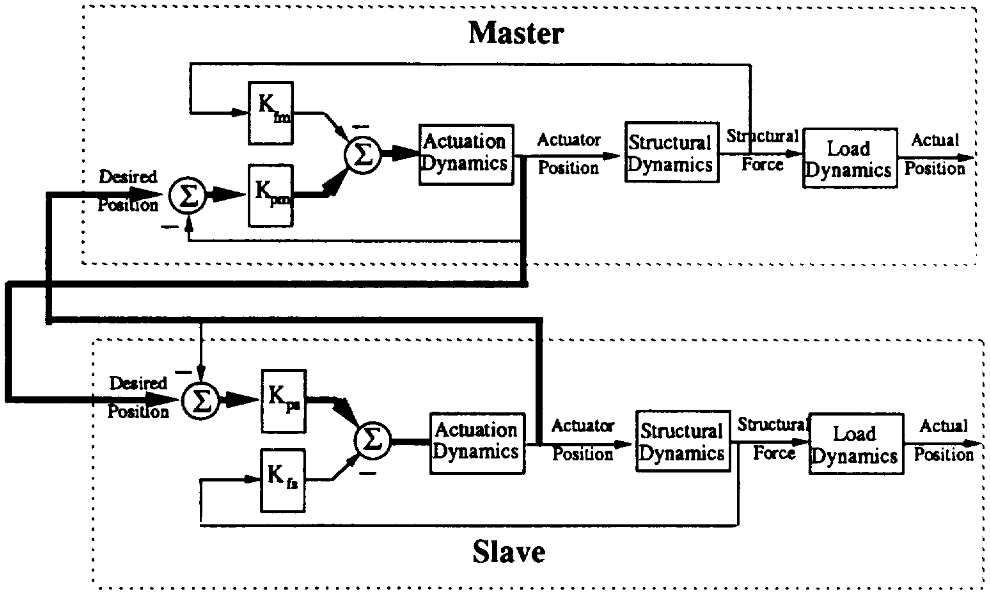


Figure 6: Block diagram of the tenth order teleoperation system composed of two fifth order effectors.

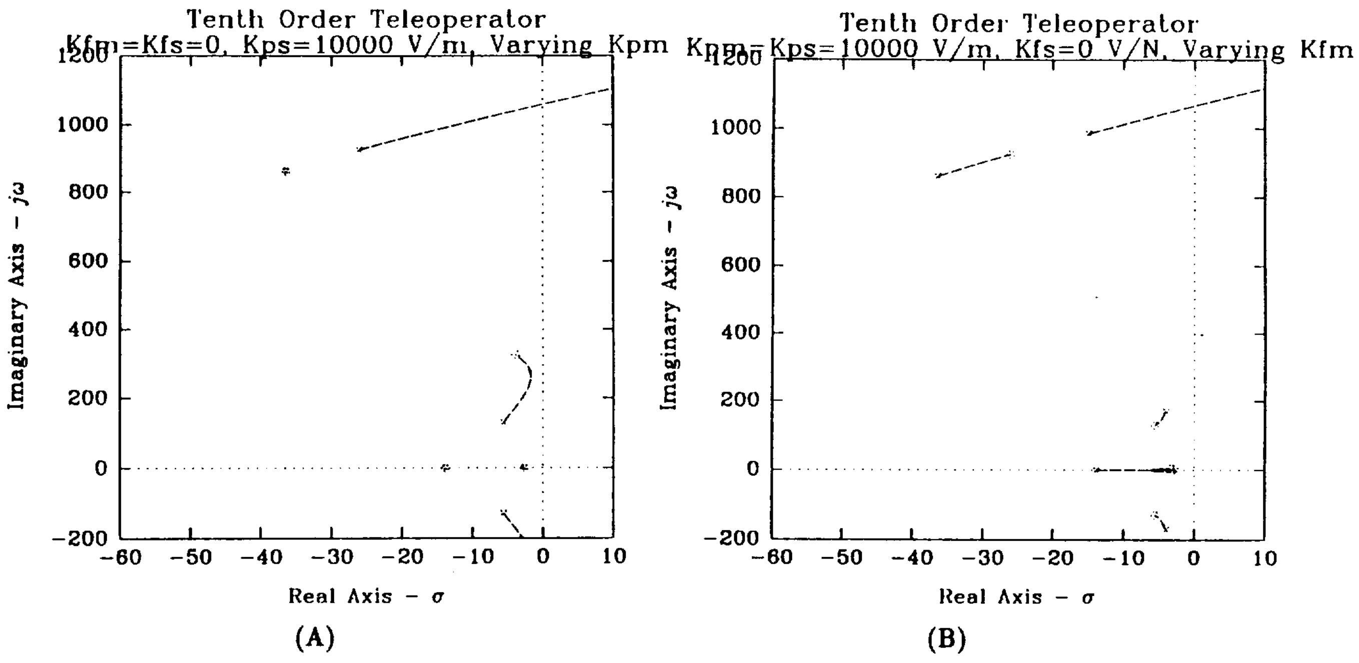


Figure 7: Typical root loci for the tenth order master/slave system derived by connecting two fifth order effector models. (A) shows the root locus for $K_{f,master} = K_{f,slave} = 0$, $K_{p,slave} = 10000$ V/m, varying $K_{p,master}$. This causes the intersystem stiffness to increase but at the same time changes the force reflection ratio (ratio of force applied by the slave to force applied by the operator) since only $K_{p,master}$ varies with $K_{p,slave}$ being help constant. (B) shows the rootlocus plot with $K_{p,master} = K_{p,slave} = 10000$ V/m, $K_{f,slave}$ = 0 V/N, varying $K_{f,master}$

as the gains vary. Note also, that for each pole of the fifth order system varying K_p, there is a pole-zero-pole triad for the tenth order system, and only one of the poles of each triad actually migrates as the gains are varied. This pole-zero-pole coupling can be more easily understood by tracing the signal crossovers in Figure 6. The highlighted signal path shows that there is in actuality a positive feedback loop which starts at the desired position of the master, passes through the master's actuator, is fed to the desired position of the slave, and then through the actuator of the slave and back to the master as the master's desired position. This pathway is never inverted and therefore behaves like a positive feedback loop which causes the zeros to appear on the root locus plots in the same positions on the s-plane as the poles of the effector model. This positive feedback loop also causes some destabilization of the overall force reflective system.

In Figures 4(A) and 7(A), the poles due to the actuator/structure interactions cross the imaginary axis at a gain of 32805 volts/meter for the effector system but this value is reduced to 22806 volts/meter for the force reflective system. The sum of the position gains of the master and slave however is identical to the value for the effector. Thus, in some sense, these position gains add for the teleoperator. This indicates that it is much more difficult to achieve comparable position gains in a force reflective teleoperator than in a simple effector due the inherent nature of the required feedback necessary to connect two effectors into a force reflective teleoperator. Thus, given identical machinery, structures, sensors, and controllers, a force reflective teleoperator configured in a position-position mode will only be half as stiff (for a force reflection ratio of one) as an effector built out of the same components.

It is also interesting that the same mode goes unstable for both the effector and the teleoperator. The interaction between the actuator and structure is the first to cross the imaginary axis. While many teleoperators have well damped and stiff structural elements, it should be remembered that this compliance can also be thought of as a compliant drive element such as a transmission or drive cable/tendon. This high frequency "jitter" is often the limiting mode when the position or force gains are raised to too great a level.

In Figure 7(B), we see that as the force gain is increased while maintaining a constant position gain, the system becomes more backdriveable. A similar "softening" occurs as in the effector of Figure 4(B). In the same way that the effector becomes more free and backdriveable, the teleoperator becomes more free. This indicates a decrease in the amount of force required to move the system and will allow an operator to more easily position the teleoperator by decreasing the impedance of the system in free space. This will allow the operator to work more comfortably and for longer periods of time since the level of exertion required to move the system will be lowered. However, it must be remembered that since an increase in either the position gain or the force gain cause the high frequency poles to migrate toward the imaginary axis, the control engineer is faced with a tradeoff between intersystem stiffness and free space impedance.

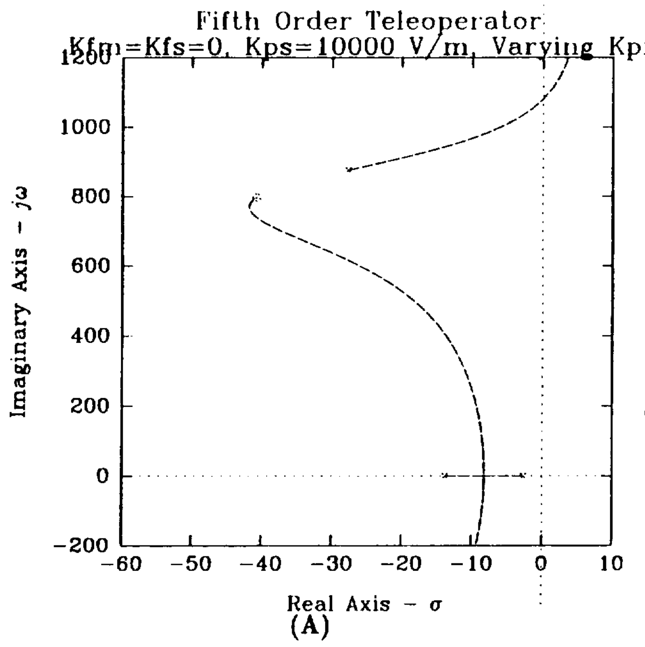
4.2 Reduced Order Model

If we wish to develop an intuitive understanding of the performance of teleoperators, we require a model which exhibits the behavior of the system, yet it must not be so complex as to not allow closed form analysis and the application of principles which are more easily applied to low order systems. However, even this limited complexity model of a teleoperator composed of two fifth order effectors has an order of ten. A model of this high an order does not generally allow closed form solutions for performance criteria and is difficult to understand without rigorous simulation and analysis. Even computer simulations can be painfully slow (and expensive). Therefore, a reduced order model which still embodies the behaviors of interest seems to be indicated.

For specific applications, many of the parameters included in the tenth order model become very small or very large or the eigenvalues associated with those parameters are no longer in a range of interest. Therefore, reduced order models are easily generated for some applications.

Third Order Master and Second Order Slave - If we wish to study the behavior of the system when the slave is in solid contact with a stiff object, we are able to reduce the order of the model to five. This is done by assuming that the master can be treated as the second order effector model and the slave can be thought of as the third order effector model presented in Section 3.2.

The master is attached to a human operator and interacts with his dynamics. In this case, one can assume that the structural compliance is small with respect to the compliance of the operator. We can also assume that the electrical time constant of the actuator is small with respect to the response of the operator which allows us to eliminate the motors inductance. This model is equivalent to the second order effector model described in Section 3.2.



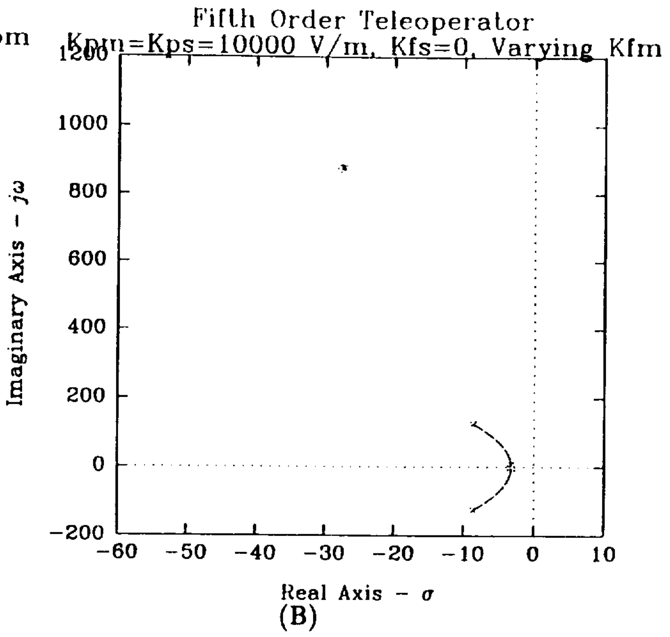


Figure 8: Root locus plots for the reduced order model a of force reflecting teleoperators. (A) shows the root locus for the fifth order master/slave system with $K_{f,master} = K_{f,slave} = 0$, varying $K_{p,master}$, $K_{p,slave} = 10000 V/m$. (B) shows the root locus for the same system with $K_{p,master} = K_{p,slave} = 10000$ V/m, $K_{f,slave} = 0$, varying $K_{f,master}$

Since the slave is in contact with a rigid object, we can assume that the load mass is relatively unmovable and the structural or drive compliance is the dominating dynamic effect. This allows us to similarly use the third order model described in Section 3.2. These simplifications lead to a fifth order model which can be used to derive performance criteria important when the slave is in contact with stiff objects.

Figure 8(A) shows a plot of the root locus for this system as the position gain is varied with no force feedback. Note the similarities and differences between the full order model presented in Section 4.1 and this reduced model. Notice that the pole-zero-pole triads are no longer present since the models for the slave and the master are no longer identical. The agreement between the full order model and the reduced order model is very good for low gains, but as the gain is increased, the poles due to the actuator/load interactions of the master now approach the open loop zeros which occur due to the actuator/structure interactions of the slave rather than the open loop zeros of the master's actuator/structural damping/load interactions. These zeros no longer exist in the reduced order model of the master. The mode which goes unstable however is unchanged between the full order model and the reduced order.

Figure 8(A) shows the same system as we vary the force gain of the master, position gains being held constant. Observe again that the reduced order model depicts the same behavior as the tenth order system. As the force gain is increased, the system becomes more "free" and the dynamics of the master and slave begin to disappear. The system behaves as if the actuator is disconnected from the system at the master, but the forces applied by the slave are still felt by the operator. The system becomes more like the ideal stick model described in Section 2.

5. The Effect of Time Delays Between Master and Slave

Often, in real teleoperation systems, there is a delay between the master and slave. This may be due to transmission delay as in space applications, where the delay may be as great as a few seconds, or it may be due to computational delay if a digital control system is interposed between the two subsystems. In this case the delay may be small, such as a few milliseconds. In our experience at the Center for Engineering Design, we have discovered that even a small delay between the master and slave can cause a serious degradation of overall system performance, especially in the areas of intersystem stiffness and free space operator input impedance. We can study the effects of this delay by interposing a first order lag between the master and

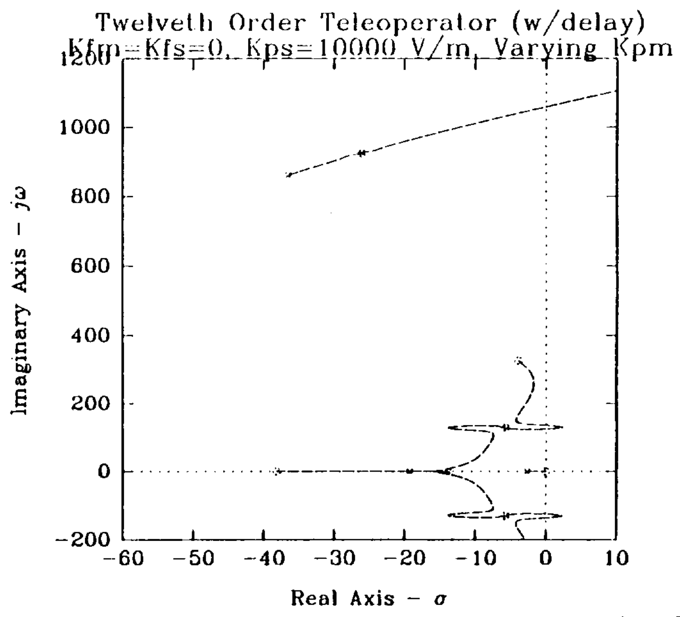


Figure 9: Root locus for the twelveth order system derived from two fifth order effectors with transmission delays between the master and slave and between the slave and master, $K_{f,master} = K_{f,slave} = 0$, varying $K_{p,master}$, $K_{p,slave} = 10000V/m$. (B) show the locus for $K_{p,master} = K_{p,slave} =$??? V/m, varying K_f

slave and between the slave and the master. We can then compare the results of the root locus analysis from the models without the delay to those with the delay. This will allow us to make inferences as to how the delay will effect overall system performance.

If we interpose the delays in the intersystem connections of the tenth order model, each delay adds another order for a total of twelve. Figure 9 shows root a locus plot for this twelveth order system varying K_p with $K_f = 0$.

Observe that the pole-zero-pole triads are no longer present. The delay causes the poles and zeros to separate and therefore the pole migrates to the shifted zero. More importantly, however, notice that in Figure 9(A), some poles due to the actuator/load interactions have been shifted to the left by the delay and move out the negative real axis to zeros arising due to the delays. This implies that the system has a higher natural frequency and is more damped for equivalent gains, in comparison to the teleoperator without the delays. The poles due to the actuator/structure interactions however, remain relatively unchanged. This means that the high frequency "jitter" is unchanged by the delay, but the lower frequency poles are more damped and less "free". Thus, the slew drag is increased by the delay and increasing the appropriate gains to minimize the effect is impossible. Notice also, that the actuator/load interactions cross the imaginary axis at a fairly low frequency (125 radians/sec.) and at a relatively low gain (8084 volts/meter).

6. Conclusions

A general model of a force reflecting teleoperation system was derived in order to examine some basic effects of position and force feedback and the inherent tradeoffs between intersystem stiffness and free space feel which must be made when setting these gains. A reduced order model was generated for specific situations of the master and slave in order to more easily understand observed behaviors. The root locus analysis applied shows that there are intrinsic trade offs which are made as we increase either the position gain or the force gain. The control engineer must balance the intersystem stiffness of the system against the impedance felt by the operator as he moves the system in free space. The effects of delays between the master and slave on the achievable intersystem stiffness and slew drag were examined. This shows that even small delays degrade the performance of the system by causing the actuator/load interactions to become more damped and sluggish. This can either be tolerated or can be minimized by increasing the force gain. If the force gain is increased however, the position gain must simultaneously be decreased, thereby degrading intersystem stiffness.

In the future, additional reduced order models will be derived and used to find closed form and numerical

Quantitaive Performance Criteria (QPC's) [6]. These performance criteria will enable a designer to easily see the impact of design decisions on the overall performance of the system. This should help a designer of a teleoperation system to more easily balance the conflicting constraints to satisfactorily meet the design goals.

References

[1] Pitman corporation specification sheet, 5000 series.

[2] Klaus B. Biggers, Stephen C. Jacobsen, and George E. Gerpheide. Low level control of the utah/mit dextrous hand. In *Proceedings of the IEEE International Conference on Robotics and Automation*, April 1986.

[3] C. R. Flatau. The manipulator as a means of extending our dextrerous capabilities to larger and smaller scales. In *Proceedings of the 21st Conference on Remote Systems Technology*, pages 47–51, 1973.

[4] Stephen C. Jacobsen, Edwin K. Iversen, Clark C. Davis, Dwight M. Potter, and Tim W. McLain. Design of a multiple degree of freedom, force reflective hand master/slave with a high mobility wrist. In *Proceedings of the Third Topical Meeting on Robotics and Remote Systems*, March 1989.

[5] Stephen C. Jacobsen, Edwin K. Iversen, David F. Knutti, R. Todd Johnson, and Klaus B. Biggers. Design of the utah/mit dextrous hand. In *Proceedings of the IEEE International Conference on Robotics and Automation*, April 1986.

[6] Stephen C. Jacobsen, et al. Behavior based design of robot effectors. In *Proceedings of the 1987 ISRR*. MIT Press, 1988.

[7] M. G. Spooner and C. H. Weaver. An analysis and analog-computer study of a force reflecting positional servomanipulator. Technical Report Reprint No. 264, University of Wisconsin Engineering Station, 1955.

[8] J. Vertut. Contribution to define a dexterity factor for manipulation. In *Proceedings of the 21st Conference on Remote Systems Technology*, 1973.

[9] J. Vertut. Control of master-slave manipulators and force feedback. In *Proceedings of the 1977 Joint Automatic Control Conference*, page 172, San Francisco, 1977.

[10] J. Vertut. Advances in computer-aided teleoperation systems (CATS) in the frame of the French ARA project. In *Proceedings of the 1983 International Conference on Advanced Robotics Suppl.*, page 13, Tokyo, September, 1983.

[11] J. Vertut, et al. Short transmission delay in a force reflecting bilataral manipulator. In *Proceedings of the 4th Ro.Man.Sy. Conference*, page 269, Warsaw, 1981.

N90-29857

Real-Time Cartesian Force Feedback Control of a Teleoperated Robot *

Perry Campbell
Lockheed Engineering and Sciences Company
Houston, Texas 77058

Abstract

Active cartesian force control of a teleoperated robot is investigated. An economical micro-computer based control method is tested. Limitations are discussed and methods of performance improvement suggested. To demonstrate the performance of this technique, a preliminary test was performed with success. A general purpose bilateral force reflecting hand controller is currently being constructed based on this control method.

I. Introduction

Most available hand controllers today do not have force reflectance capability, which gives the human operator much needed force information. Such a capability is needed in order for operators to better control their manipulators. The systems which do use force feedback are made of dedicated sub-systems which cannot be inter-changed, i.e. the hand controller from one system is not useable with the robot of another system. For example, the force-reflecting hand controller manufactured by Kraft, Inc. is not compatible with a PUMA or Robotics Research robot. Work at the Teleoperator Systems Branch Laboratory of the Engineering Directorate at the Johnson Space Center is currently addressing this problem as it relates to laboratory and space teleoperators.

The objective of this work is to construct and demonstrate a force reflecting hand controller which is capable of driving several different types of robot slaves. The hand controller is currently under construction, so only preliminary system performance data is available at this time.

In order for such a hand controller to be compatible with different types of slave robots, it must perform its control function in some coordinate frame which is common to most manipulators. The most obvious frame to use is a cartesian (tx,ty,tz,rx,ry,rz) frame which is either fixed in the end link or in the base of the robot. Many robot controllers are factory programmed to accept cartesian position commands with no modifications to their control software. Most non-force reflecting hand controllers today are designed to issue command signals in cartesian space.

---

* This work presents the results of work carried out at the Johnson Space Center, under contract No. NAS9-17900, National Aeronautics and Space Administration.

The "common module" concept has been successfully applied in the Shuttle program. For example, the Shuttle Remote Manipulator System (RMS) has a translational rate hand controller and a rotational rate hand controller. Both of these hand controllers are identical to those used to control the Shuttle's navigation jets, with the exception of an additional switch installed on the rotational hand controller (for rate hold of the RMS). In the Shuttle, the primary advantage is that there was no re-design effort for the RMS hand controllers. Another important factor, which is only now beginning to be appreciated, is that one-of-a-kind items are very susceptible to premature obsolecence. This effect is being observed in the Shuttle program, where original vendors are now beginning to close plants which manufacture parts which are used only by a particular shuttle sub-system. It is very costly to obtain new parts when this happens (a new manufacturer must be contracted to second-source the hardware). When a particular part is used in many places on the Shuttle, a single programmatic effort is all that is required to assure a continuing supply of flight spares throughout the life expectancy of the vehicle. It is expected that commonality will be an important factor in sub-system component design in the Space Station program. There will be several different types of robotic sub-systems on or around the Space Station Freedom, each of which may require a hand controller signal. If these sub-systems utilize force feedback in the teleoperator mode of operation, then some type of force reflecting hand controller will be needed for each teleoperator sub-system. Clearly, an attempt should be made to utilize a single design for all of these applications. For the Space Station, there is the additional factor of the duration of its mission: for extended periods of space activity, the risk is greater that a hardware failure will render a system useless unless spare components are kept on-board (very expensive, but sometimes necessary). If some or all of the teleoperator sub-systems aboard the Space Station used a common hand controller, then a single flight spare could replace any one of them, should it fail.

There is also a need for a single hand controller which can operate several geometrically different robots in our robotics laboratory. We want to have the capability to re-run a test using any of the robots, rather than being required to use only the robot for which a specialized hand controller was built. A single generic hand controller is obviously less expensive than several custom-built hand controllers. There will also be a need for a general purpose hand controller in the MPAC (Multi-Purpose Applications Console), which is a generic operator control station also being developed under the Engineering Directorate.

Several tests have been planned in the Telerobotics Laboratory related to this project. Two are of importance here:

1. Cartesian force control of a single robot.
2. Cartesian force feedback control of a robot and a hand controller.

Of these two tests, only test number 1 has been completed. Test number 2 is pending hardware construction. This report will present an overview of the results of test number 1 and will discuss the present state of test number 2.

II. Hardware Configuration

In test number 1, a PUMA 762 robot was programmed to perform motions in the External Alter control mode. The commands sent to the robot were computed by an 80286 based personal computer (PC-AT). The inputs for the PC-AT computations were a set of forces and torques from a force / torque sensor (F/TS) mounted at the robot tool plate. Behavior characteristics were programmed on the PC so that the robot tip would respond to forces as desired. Several different behaviors were coded and tested and the performance of the system was studied for each case. (Figure 1.)

In test number 2, a PUMA 762 or Robotics Research (RR) 2107 or 1607 will perform as a slave robot under the control of the above-described hand controller. (Figure 2.) The hand controller is to be constructed from a PUMA 250 or 260 robot. This is possible because bilateral force reflectance requires that both master and slave perform similarly. It is sometimes helpful to visualize such a system as two robots being controlled together, rather than one robot and one hand controller. At one robot, the environment it interfaces with is a human operator (a hand). At the other robot, the environment is the actual environment which the operator wishes to interact with.

The present hardware configuration requirement for test 2 is as follows:

1. Large robot and controller (PUMA 762 or RR 1607).
2. Small robot and controller (PUMA 250 or 260).
3. PC-AT with STARGATE serial interface hardware.
4. Cabling as necessary.
5. Two force / torque sensors.

Test number 1 required only one robot and one force / torque sensor.

III. Results of Test Number 1

In test number 1 the function of the control PC was solely to wait for the VAL controller to request another position command, send a command when requested, request data from the Force Torque Sensor every time the VAL asks for a position command, and to read the F/TS data when it arrives on the serial port.

The demonstration given for test 1 utilized several very simple control laws. One law implemented made the end point behave as if it were the center of gravity of a mass in a zero-g field. (a free body). The physical law governing a free body is: F=m*a. Therefore the acceleration of the body should be proportional to the external force acting on it. The external alter function of VAL expects the command signal to be either position or change of position (cummulative or non-cummulative). The latter mode was used in this test. The velocity of a free body is simply the integral of the acceleration. Digitally, it is the sum of all previous force measurements divided the sampling period and also divided by the mass we wish to make the end point behave like.

Another control law implemented was similar to the above law except a certain amount of damping was added, making the "free body" behave as if it were moving through a viscous fluid. This was done by using the same law as above and subtracting a term proportional to the previous velocity command.

A third law was to make the end point behave like a massless damper. This was the simplest law to implement because the velocity command is simply proportional to the F/TS measurement. The end point would move at a rate which was exactly proportional to the force at the end point.

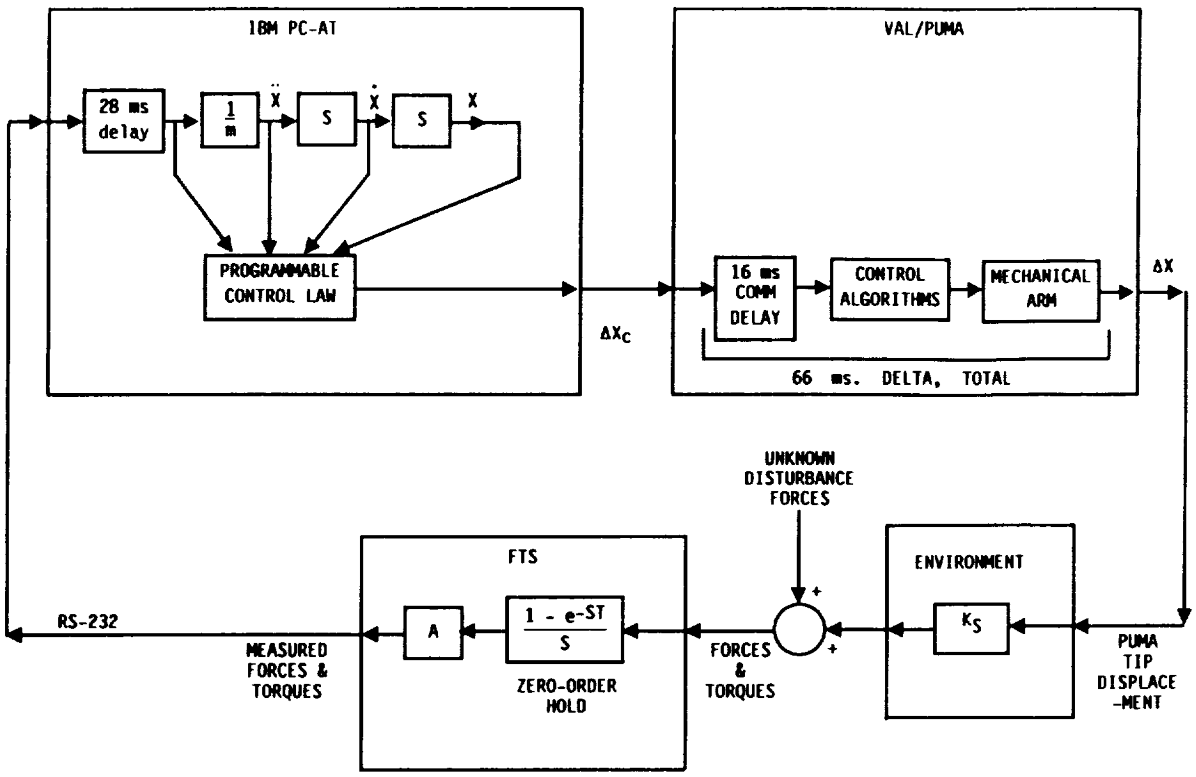


Figure 1: Force Feedback Control Diagram.

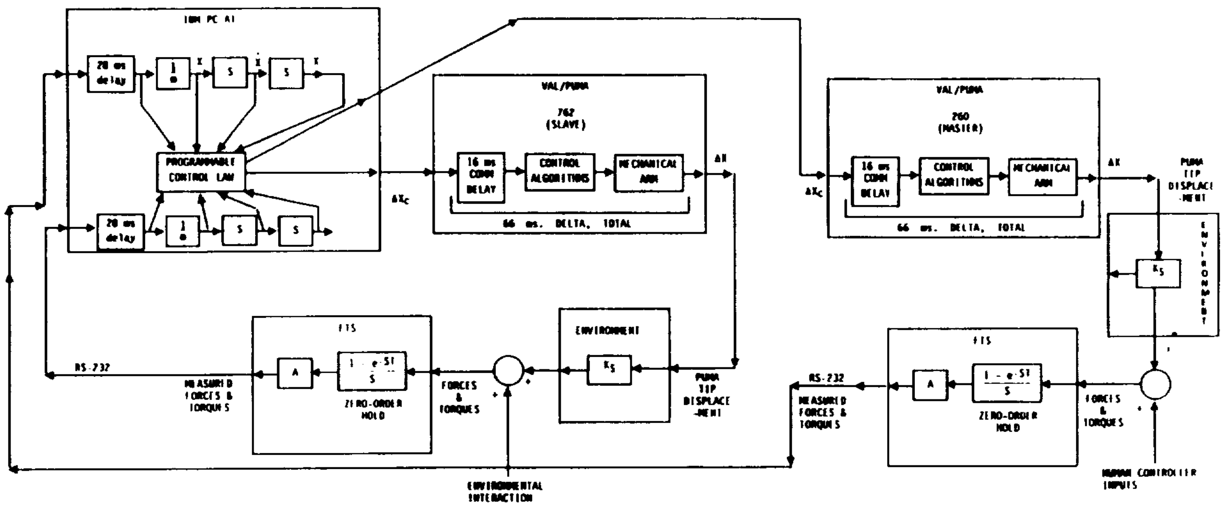


Figure 2: Force Reflecting Hand Controller Block Diagram.

ORIGINAL PAGE IS
OF POOR QUALITY

During the evaluation of the above control law algorithms, it was discovered that momentum was not being conserved in the "free body". This was seen when a bias force was added and a spring mounted under the gripper. The arm tip was allowed to "fall" down to a table-top. If the end point was behaving like a free mass (with the simulated g-field) it should bounce back up to the same height from where it was dropped. This was not the case. The end point bounced up significantly higher on each bounce. Thus, momentum was not being conserved in our simulated environment. The reason for this behavior is speculated to be:

The Puma arm does not reach the commanded position until about 66 ms. after the command is issued. The net effect is that the contact point of the arm is allowed to continue to travel into the table-top for a full 66 ms. longer than it should be allowed to. This delay allows the spring to be compressed a full 66 ms. longer than it would have been if a true mass had been used. Thus, when the tip exits in the opposite direction, the exit velocity is larger than when it entered because the measured forces are larger than they would have been had a true mass been used. This effect has been confirmed by simulation of the cartesian control system.

A simple effort was made to reduce the above described effects. A predictive model was added to the PC software which read the current forces and used them and the previous forces to linearly extrapolate what the forces were going to be when this position command was actually reached by the PUMA arm. This made the system stable but still did not conserve momentum. This is because the predictions are required to be exact for conservation of momentum to hold and our prediction model is not exact. Nevertheless, the predictive model made the end point behave much more like a free body than without it. Other methods (such as correcting the estimate after the fact) were deemed beyond the scope of this test.
Another problem associated with the delays and sampling time was the inability of the arm to maintain a constant force on a rigid surface. This was because the two hard surfaces tended to bounce at rates much faster than could be compensated for. This caused a problem when the two rigid surfaces were to be held together at a constant force. By adding the predictive algorithm and a large amount of damping, the problem was resolved for the case of laying the gripper directly on a table top. Much stiffer surfaces could still cause problems. The modifications made caused the manipulator to behave very sluggishly. The sampling rate used in this test was about 40 Hz.

It is anticipated that sampling rates of about 500 Hz will be sufficient for most space-based force feedback manipulator control systems.

IV. Test Number 2 - Accomplishments to Date

In test 2, the control algorithm in the PC takes 2 force/torque measurements and issues 2 cartesian position commands to the robot controllers during each update period. The system delays are about twice that in test 1 and so performance is expected to be diminished by about a factor of two. This test has not been performed yet. To date, the only data available are expected results, taken from a computer simulation of the proposed hardware/software arrangement.

Preliminary simulation indicates that a reasonable performance is attainable. Figure 3 illustrates the force-tracking capability of the simulated controller in response to a sinusoidal input at one robot end effector while the other robot is constrained in a spring- like environment. Figure 4 shows the step response of

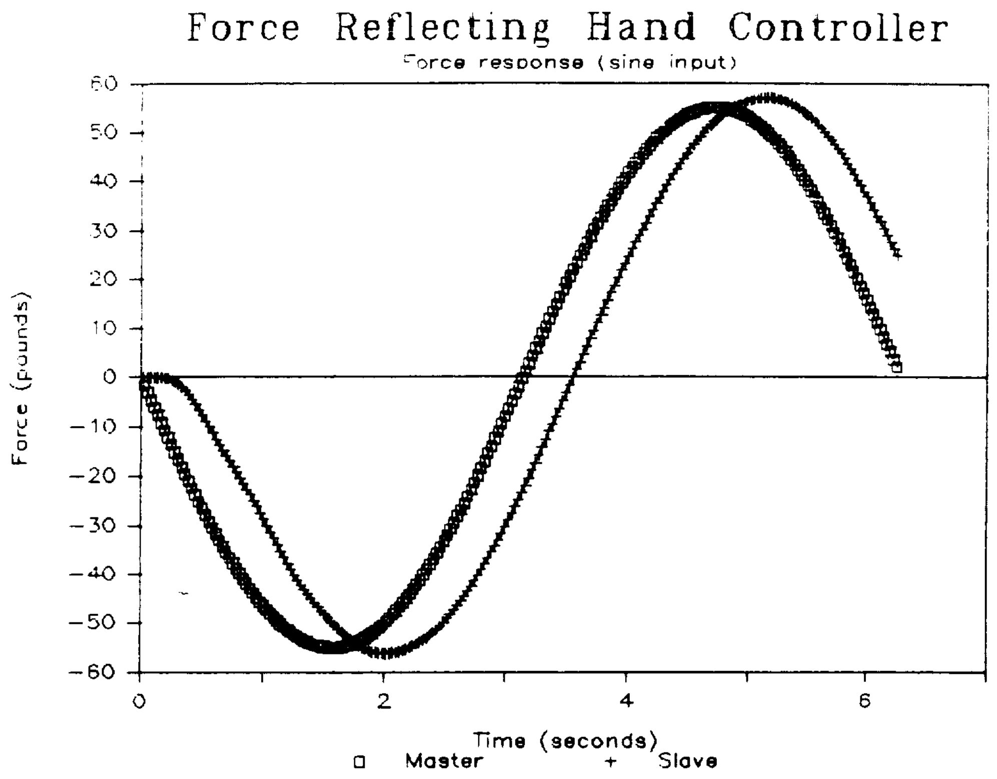


Figure 3: Simulated Force Tracking (Sinusoidal Input).

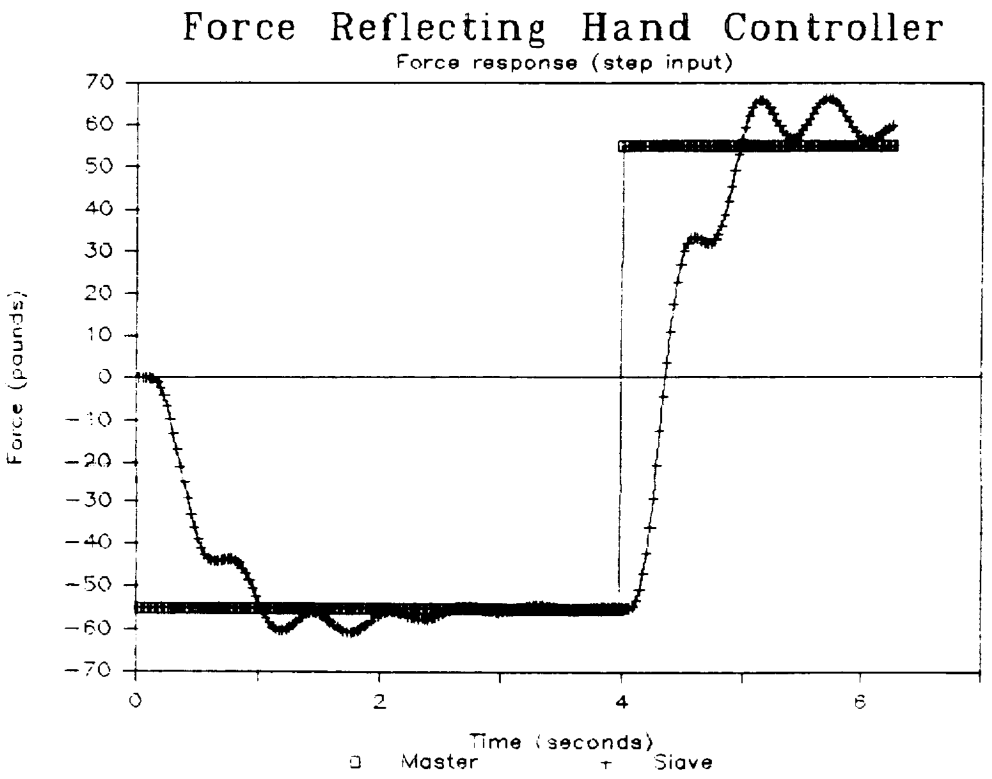


Figure 4: Simulated Force Tracking (Step Input).

the same system. It is assumed that the manipulator performs as a perfect positioning device. Each robot and controller is modelled as a perfect positioner with a mass attached to it by a spring. The mass interacts with the environment, which is also modelled as a spring. The control algorithm simulated is as follows, for each time step:

```
MFm(t)     = MK*(MXcmd(t)-MXee(t))
MFe(t)     = C*u(t)                                  (Step response input)
MAee(t)    = (MFm(t)+MFe(t))/Mm
MVee(t)    = MVee(t-T)+MAee(t)*T
MXee(t)    = MXee(t-T)+MVee(t)*T
MXcmd(t)   = MXcmd(t-T)+(MFm(t)-SFm(t))*T*MG
SFm(t)     = SK*(SXcmd(t)-SXee(t))
SFe(t)     = -MXee(t)*MKe                            (Spring-like environment)
SAee(t)    = (SFm(t)+SFe(t))/Sm
SVee(t)    = SVee(t-T)+SAee(t)*T
SXee(t)    = SXee(t-T)+SVee(t)*T
SXcmd(t)   = SXcmd(t-T)+(SFm(t)-MFm(t))*T*SG
```

where:

A preceding 'M' denotes a Master quantity
A preceding 'S' denotes a Slave quantity
e.g. MFm = Master force (measured)
SFm = Slave force (measured)

Fm	= force (measured)
Fe	= force (at environment interface)
Aee	= acceleration of robot tip
Vee	= velocity of robot tip
Xee	= position of robot tip
Xcmd	= commanded position of robot tip
T	= time step update period (28 ms. for PUMA)
t	= present time
m	= end effector mass
C	= arbitrary magnitude for step response input
Ke	= environmental stiffness at contact point
K	= effective stiffness of robot at tool
G	= cartesian controller gain

This controller essentially sets the velocity of each of the robots to be proportional to the difference between the measured forces. Simplicity is the driver for selecting this algorithm. It is clear how this control architecture is well suited to other control algorithms such as active stiffness control [1], impedance control [2], and hybrid control [3].

V. Conclusions

From these results, it appears feasible and even desirable to construct a bilateral force reflecting hand controller based on cartesian force control algorithms. System performance is mainly limited by computer power. The system under development is limited by the lower level robot controller loop times. This limitation may be extended greatly by modifying the individual robot controllers. Update rates of 40 Hz make the system too sensitive to the environment. It is estimated that 200 to 500 Hz will be more acceptable. A plan has been established to upgrade the Robotics Research controller to obtain a looptime of approximately

2 ms. Such computational demands can be met by a high end dual-processor 80386 based personal computer with math coprocessors or by a reduced instruction set computer such as the IRIS 4D/70G.

REFERENCES

1.Salisbury, J.K. (1980) "Active Stiffness Control of a Manipulator in Cartesian Coordinates". Proceedings of the 19th IEEE Conference on Decision and Control, pp. 95-100.

2. Hogan, N. (1985) "Impedance Control: An approach to Manipulation:Part I - Theory, Part II - Implementation, Part III - Applications".Journal of Dynamic Systems, Measurement, and Control, pp. 1-24.

3. Craig, J.J. & Raibert, M.H. (1981) "Hybrid Position/Force Controlof Manipulators". Transactions of ASME, Journal of Dynamic Systems, Measurement, and Control, Volume 102, pp. 126-133.

N90-20258

"OPTIMAL PAYLOAD RATE LIMIT ALGORITHM FOR ZERO-G MANIPULATORS"

M.L.Ross, D.A.McDermott

Lockheed Engineering & Sciences Company
Houston, Texas 77258

Abstract

An algorithm for continuously computing safe maximum relative velocities for two bodies joined by a manipulator is discussed. The maximum velocities are such that if the brakes are applied at that instant, the ensuing travel between the bodies will be less than or equal to a predetermined amount. This paper deals with an improvement in the way this limit is computed for space manipulators. The new method is explained, test cases are posed, and the results of these tests are displayed and discussed.

I. Introduction

A. What is a payload rate limit algorithm?

The rate limit for a payload is, in effect, a "speed limit" for the rates the payload is allowed to achieve relative to the manipulating body. Observance of this rate limit ensures that the payload's relative motion can be arrested at any time during a maneuver (e.g., in an emergency such as a joint runaway detection) without exceeding a specified amount of overtravel after application of the brakes. Any method employed to compute these rates must consider items like brake torque capability of the manipulator and masses of the payload and manipulator base.

The Remote Manipulator System (RMS) on NASA's Space Shuttle currently utilizes a rate limit algorithm to compute a single (constant) rate limit for each payload to be manipulated during a mission, and these rates are loaded into the on-board computer system prior to the mission. These rates are designed to ensure a stopping distance of the end-effector (not the payload c.g.) in two feet or less when the arm is in an out-stretched position (worst-case).

B. What is wrong with the current rate limit algorithm?

The algorithms developed to determine the rate limit for a payload to be carried by the RMS were developed by SPAR Aerospace, Inc. in August 1979 with revisions in February 1983 [Ref. 1,2]. These algorithms were designed for the original payload mass range of up to 65,000 lbm. (vs. Shuttle mass of 220,000 lbm), and also assumed a worst-case RMS configuration (fully extended). While these algorithms have performed well for the range of payloads seen so far, they produce rate limits approaching zero for payloads whose masses exceed 100,000 lbm. They also produce only one limit, based on the worst-case configuration, when in fact the RMS's ability to arrest relative motion is dependent on arm configuration and the direction of the motion to be arrested. This means the rate limits calculated are lower than necessary (conservative), so the payload is always manipulated at rates lower than actually required for safety.

C. Why should we care?

The RMS manipulator is basically a dexterous crane that has to perform large-excursion travel in many of its tasks. If used, the low rate limits that the current algorithm produces for the massive payloads will cause increases in the time required to perform the job because the arm will move much slower than is necessary for safety. The low rate limits also push all the commands down into the low end of the command scale. This lessens the number of bit-states which can represent the command, thereby decreasing resolution and accuracy of the command actually sent to the joint servos.

The current algorithm must be changed to consider payloads more massive than 65000 lbm., and must consider the fact that the Shuttle also moves during the braking process. Since the algorithm will require at least an update to handle the larger payloads, this is a good time to look into deriving a better method. This method should be dependent on the arm configuration and commanded velocity.

Some analysis on improving the algorithm has been done before, but the algorithm arising from that analysis was still a single limit for a payload [Ref. 3]. That method did, however, take into account the fact that the Shuttle moves during manipulation, and was the basis of this current effort.

II. Proposed New Payload Rate Limit Algorithm

A. What should it do?

Any new method should include better models of system dynamics, and yet be computationally simple. Simplicity will enable calculation of the limit in real-time as a function of the current arm configuration and the requested direction of motion. The new method should also consider relative rotational motion, and allow a maximum rotation angle as an additional criterion.

The end goal of this algorithm should be to produce a rate limit which is higher than the currently-used conservative value yet is still safe.

B. How can it be done?

1. System Dynamics

By considering the system dynamics of the payload and the Shuttle over the process of:

Phase 1. beginning with relative rates (possibly zero) between Shuttle and payload,
Phase 2. accelerating the payload and Shuttle to some new commanded relative rate, and finally

Phase 3. applying the brakes until all relative rates have been arrested,

one can see that, since we assume no external forces, system momenta is conserved throughout this process. The dynamics from the time of brake actuation to arrest of relative motion (Phase 3) can be viewed as an inelastic collision, in which both bodies have initial relative motion, collide and stick together, and then proceed on as one body.

Realizing that the final system velocity V_f is constant, we can define a reference frame F, translating at velocity V_f. The final rotational rates of the system are expected to be very small (less than 0.2 degrees per second) This means we can allow the frame F to rotate with the system at its final rotational velocity ω_f and still consider it inertial (pseudo-inertial). In this frame, an observer would see that each body would have an initial velocity, but would come to rest at impact. The relative velocities between these two bodies would be the commanded velocity. The momentum equation can be written for each body in this pseudo-inertial frame:

$$m_1\underset{\rightarrow}{V_1} + \int_0^t \underset{\rightarrow}{F}(t)\ dt = 0 \quad \text{and} \quad m_2\underset{\rightarrow}{V_2} - \int_0^t \underset{\rightarrow}{F}(t)\ dt = 0$$

so

$$\underset{\rightarrow}{V_1} = -\frac{m_2}{m_1}\underset{\rightarrow}{V_2}$$

which says that the initial velocities of the two bodies in this frame will always be opposed and parallel. Because the common direction of these velocities is also the direction of the relative velocity, and we assume the force between the bodies to be opposed to the relative velocity so everything falls on this line, leaving us a scalar equation.

Writing the equation of linear momentum for the system in frame F, we get the following equations, which we can treat as scalar since all vectors are parallel.

For the entire system (no external impulse) :

$$m_1V_1 + m_2V_2 = 0$$

For bodies 1 and 2 individually (external impulse from arm) :

$$m_1V_1 + \int_0^t F(t)\ dt = 0 \qquad m_2V_2 - \int_0^t F(t)\ dt = 0$$

where $(V_2 - V_1)$ is the commanded velocity.

2. <u>Estimation of Impulse</u>

If we had some idea about what kind of impulse we could expect to see, we could extrapolate what the travel motion of the system would be. The impulse does not have to be exactly known, but the estimate must be less than the actual impulse, and still produce a rate faster than the old value. A way to estimate the impulse is to estimate the initial force or torque, and then estimate the manner in which it approaches zero.

Any force applied by the RMS to the bodies it connects will result from brake slippage and storage of strain energy in the booms and gearboxes. Since only brake slippage is non-conservative, this mechanism alone will be considered. The potential brake torque at each joint is assumed to be known. In this case, the initial resistive force encountered by the bodies can be estimated. By assuming all joints required to contribute to the commanded motion will slip if the brakes are applied, a vector of brake torques can be created which is an estimate of the torques which would be seen at each joint:

$$(\underrightarrow{B}) = -\begin{pmatrix} \tau_1 * \mathrm{sign}(\dot{\gamma}_1) \\ \vdots \\ \tau_6 * \mathrm{sign}(\dot{\gamma}_6) \end{pmatrix}$$

where sign() is -1 if the commanded joint rate is negative, zero if zero, and +1 if positive.

Here we assume that the inverse of the Jacobian matrix relating end-effector states to joint states is known or easily obtained. Multiplication of the transposed inverse of the Jacobian by the brake torque vector just constructed results in the static moments and forces at the end-effector to counteract those brake torques.

$$\begin{pmatrix} \underrightarrow{T} \\ \underrightarrow{F} \end{pmatrix} = [J]^{-T}(\underrightarrow{B})$$

These loads at the end-effector will not generally be parallel to their counterpart velocities. The components of these loads in the direction of those velocities are an estimate of initial loads the payload and shuttle will encounter, while the components normal to those velocities are assumed to arise from the errors in the assumed joint torque vector.

$$T_{max} = \underrightarrow{T} \cdot \hat{\omega}_c$$
$$F_{max} = \underrightarrow{F} \cdot \hat{V}_c$$

where

$\hat{V}_c$, $\hat{\omega}_c$ are the unit vectors of the commanded velocities, and T_{max}, F_{max} are the available torque and force in those directions (scalar). These loads are the estimates of the actual initial loads seen upon applying the manipulator brakes.

So far this method predicts the initial loads seen at the beginning of the braking, but its behavior of these loads over time is unknown. Since we are only having to compute a conservative limit, we only need assume a conservative load profile, i.e. the assumed profile's integral must always be less than or equal to the actual impulse. Nominally, loads due to brake slippage could be thought of as constant over time, but simulation has shown that these loads tend to drop off over the braking maneuver. To be conservative, we assume a profile in which the loads begin at the predicted value (F_{max}, T_{max}), but then linearly ramp down to zero over the time of the maneuver.

3. Calculation of Maximum Safe Speed

a. Linear Velocity

Once we know or assume a force-time profile (impulse), we can determine the distance travelled over that time and make sure all distances travelled are less than 2 feet (or some other criterion).

To do this, we must look at the equations of motion of these bodies along the line of action in frame F:

$$F(t) = A - Bt \qquad \{ 0 < t < t_{end} \}$$

where $F(t)$ is the force exerted on each body,

t_{end} is the time at which motion stops

A is the maximum force F_{max} computed above

B is the slope at which F(t) ramps down, i.e., F_{max}/t_{end}

so the acceleration of each mass is

$$a(t) = (A - Bt)/m$$

and the velocity of each mass is

$$v(t) = \int_0^{t_{end}} a(t)\ dt + V(t=0)$$

$$= (At - Bt^2/2)/m \quad (0 < t < t_{end})$$

and the distance travelled by each mass is

$$x(t) = \int_0^{t_{end}} V(t)\ dt + X(t=0)$$

$$= (At^2/2 - Bt^3/6)/m \ (0 < t < t_{end})$$

We can also see that since

$$x_1(t=t_{end}) + x_2(t=t_{end}) < 2 \text{ feet}$$

we can back out a relationship between the F_{max} and the amount of time required to arrest the motion, t_{end}.

$$2 \text{ feet} = (At_{end}^2/2 - Bt_{end}^3/6)(1/m_1 + 1/m_2)$$

or, substituting for $A = F_{max}$, and $B = F_{max}/t_{end}$, we get

$$t_{end} = \sqrt{[3*d*m_1*m_2/((m_1+m_2)*F_{max})]}$$

where d is the stopping distance allowed, which is 2 feet for the RMS.

Once we know the amount of time required to arrest the motion, we can use the momentum equations for each body:

$$m_1V_1 + \text{impulse} = 0, \quad m_2V_2 + \text{impulse} = 0,$$

and since the impulse is the area under the assumed force-time curve,

$$\text{impulse} = 1/2\ F_{max}\ T_{end}$$

and the final allowable command velocity is $V_2 + V_1$,

$$V_{max} = (F_{max} * t_{end}/2) * [1/m_1 + 1/m_2]$$

b. Rotational Velocity

The computation of the maximum safe rotational velocity closely parallels that of the translational. An assumption of the torque-time profile (linearly approaching zero) is made, and the equation of angular momentum is written for the system about the end-effector tip. This requires that all body inertias be computed about that point.

The assumed torque-time profile is

$$T(t) = A - Bt \quad (0 < t < t_{end})$$

where A is T_{max} (initial torque)

B is the slope at which it ramps down (T_{max}/t_{end}).

For each body,

$$T(t) = I\alpha$$

where

I is the scalar moment of inertia at the end-effector about the rotation axis, and

α is the rotational acceleration.

Solving for α, $\Delta\omega$, and $\Delta\theta$

$$\alpha = I^{-1}[A - Bt]$$

and

$$\Delta\omega = I^{-1}[At - Bt^2/2]$$

and

$$\Delta\theta = I^{-1}[At^2/2 - Bt^3/6]$$

As before, we set the travel constraint

$$10 \text{ degrees} = \Delta\theta 1 + \Delta\theta 2 \quad (\text{at } t = t_{end})$$

or

$$\theta_{max} = 10 \text{ degrees} = (I_1^{-1} + I_2^{-1})\,[T_{max}\, t_{end}^2/3]$$

which yields the expression for tend

$$t_{end} = \sqrt{((\theta_{max}/57.3 * I_1 I_2 * 3)/((I_1 + I_2) * T_{max}))}$$

which yields the expression of commanded velocity

$$\omega_{max} = (T_{max}\, t_{end}/2) * [I_1^{-1} + I_2^{-1}]$$

much as the translational equation.

This V_{max} and ω_{max} are the magnitudes of the allowable velocity in the commanded direction. This command is what would be sent on to the robot controller, if the operator desired to go the maximum safe speed.

4. Application of Limit to Command

Obviously, an operator would not want to go the maximum safe speed in all situations, so what to do with the knowledge of this instantaneous speed limit raises several possibilities. One way to apply the limit would be to use some constant conservative limits under normal operation, and use the maximum limits whenever a hand controller has been fully deflected in some axis. Another method would be to use the maximum limit as the upper end of the hand-controller's range, i.e. if the hand-controller is deflected 50% in some direction, then the commanded velocity would be 50% of the maximum safe velocity. This latter appears to be possibly unwieldy for the operator, since a constant deflection of the controller would produce varying velocity commands as the arm's configuration changes. Either of these methods could be implemented as an operator-requested mode.

III. Tests, Results and Conclusions

A. How can we evaluate this new algorithm?

In order to evaluate the performance of the algorithm, dynamic simulation of the Shuttle/RMS/Payload system was required. This was done with a dynamic batch simulation program developed at the Johnson Space Center which uses an extensive model of the Shuttle and the RMS. This program (called MIRRORS, for Model for Integrated Robotics Research and Operational Requirements Synthesis) is a spin-off of the PDRSS (Payload Deployment and Retrieval System Simulator, which is a spin-off of the SVDS (Space Vehicle Dynamic Simulator), which was written during the Apollo Program. The program has been checked against actual flight data from Shuttle/RMS missions as well as other simulators, and is used routinely for RMS maneuver simulation.

The underlying idea behind the evaluation was to get a prediction of safe velocity from the algorithm and then start the simulation with those velocities and the brakes on, watching the ensuing travel. This was done while varying command direction, payload mass and arm configuration. The commands given were all single-axis commands:

X -	translate in positive Orbiter X-axis (toward nose)
Y -	translate in positive Y-axis (toward starboard wing)
Z -	translate in positive Z-axis (down toward bay)
R -	rotate in positive direction about X-axis (roll)
P -	rotate in positive direction about Y-axis (pitch)
Y -	rotate in positive direction about Z-axis (yaw).

The algorithm was coded into the flight software module of the MIRRORS program. Tests were run in the following manner for commands in each rotational and translational axis:

1. For given command direction, determine from the algorithm the maximum safe speed.
2. Initialize simulation with payload moving at that speed in the commanded direction, and with the brakes just applied.
3. Let simulation run until motion arrested, compare amount of travel with specified maximum (2 feet or 10 degrees in all cases).

The test runs were conducted for three different payloads, all 15 foot diameter homogeneous cylinders, grappled on the side at the midpoint, having masses of 32000, 100000, and 250000 lbm. The test also used two different initial arm configurations, for a total of 6 * 3 * 2 = 36 test runs.

To evaluate the amount of travel, the code was altered to compute the Euclidean distance from the current position to the point where the brakes were applied. For the rotational cases, the angular displacement about the relative rotational eigen-axis (component of the quaternion relating payload attitude relative to the Orbiter) was computed. These are included in the test results below.

B. What were the test results?

The test results for all 36 runs were in general agreement with the desired end goals, in that 33 of 36 runs resulted in payloads travelling 2 feet/10 degrees or less while at speeds faster than the old method would allow. Three runs, however, did result in up to 2.3 feet of travel, all in the Z direction. The results of all the runs are tabled below, for each arm configuration (joint angles from shoulder yaw to wrist roll shown in parentheses). The three runs that exceeded the travel limits are also noted. The format for the data for each run (three numbers) is as follows:

[1] rate limit as predicted by old method (feet/sec or deg/sec)
[2] rate limit as predicted by new method
[3] stopping distance or angle for new rate (feet or degrees)

ARM CONFIGURATION #1 (-90,90,-71,0,0,0):

	Command Directions						
	X	Y	Z	R	P	Y	
	feet /sec and feet			degrees/sec and degrees			
Pay-load							
32K	0.153	0.153	0.153	0.498	0.498	0.498	old rate limit
	0.250	0.252	0.324	1.372	0.687	0.718	new rate limit
	0.990	0.354	2.365*	2.027	3.417	3.177	stopping distance or angle
100K	0.100	0.100	0.100	0.284	0.284	0.284	
	0.160	0.161	0.207	0.779	0.400	0.411	
	0.749	0.223	2.362*	1.907	3.497	3.000	
250K	0.071	0.071	0.071	0.180	0.180	0.180	
	0.123	0.124	0.159	0.497	0.269	0.267	
	0.624	0.228	2.225*	2.037	3.690	2.770	* Exceeded 2' criterion

ARM CONFIGURATION #2 (-48,118,-118,-26,-39,3):

	Command Directions					
	X	Y	Z	R	P	Y
	feet /sec and feet			degrees/sec and degrees		
Pay-Load						
32K	0.153	0.153	0.153	0.498	0.498	0.498
	0.361	0.349	0.287	0.195	0.502	0.935
	1.445	1.364	0.975	0.101	0.229	0.445
100K	0.100	0.100	0.100	0.284	0.284	0.284
	0.231	0.223	0.183	0.112	0.291	0.542
	1.283	1.215	0.874	0.101	0.229	0.435
250K	0.071	0.071	0.071	0.180	0.180	0.180
	0.177	0.171	0.141	0.730	0.193	0.361
	1.161	1.101	0.831	1.206	0.232	0.444

The amount of additional computer time required to compute this rate limit was negligible for the simulation, which is already numerically intensive. This is not an indication of its impact on some other manipulator, although the scheme doesn't require much numerical work provided the inverse of the Jacobian matrix is available.

C. Any conclusions?

Since the tests did produce three runs which exceeded the 2 feet limit, one conclusion is that further work is needed in better estimating the impulse imparted between the bodies. However enough runs (33 out of 36, or 92%) not only stopped well within the limit, but at speeds faster than the present method would allow, which indicates the potential worth of this form of electronic safety monitoring. The conclusion of the study conducted to date is that further investigation is warranted to increase accuracy of the impulse estimation. If this can be improved and remain numerically simple, the algorithm will be a useful tool in speeding-up tasks for space robots.

IV. REFERENCES

[1] "SRMS Manipulator Arm Payload Handling Capability", SPAR Aerospace memo SPAR-RMS-TM.211, Geoff Marks, August 1979, SPAR Aerospace Limited, 1700 Ormont Drive, Ontario, Canada M9L 2W7

[2] "Report on an SRMS Stopping Distance Study Including Transient Arm Effects", SPAR Aerospace memo SPAR-RMS-TM.601, J. Dueckman, February 1983, SPAR Aerospace Limited, 1700 Ormont Drive, Ontario, Canada M9L 2W7

[3] "Proposed Joint Rate Limit Algorithm for Heavy Payloads", Lockheed Engineering & Sciences Co. memo LEMSCO-24097, D. McDermott, Oct. 1987, Lockheed Engineering & Sciences Co., 2400 Nasa Rd. 1, Houston, TX 77258

N90-29859

ASSEMBLY OF OBJECTS WITH NOT FULLY PREDEFINED SHAPES

M.A. Arlotti, V. Di Martino

IBM Rome Scientific Center
via Giorgione 159
Roma, Italy, 00159

Abstract

An assembly problem in a non-deterministic environment, i.e. where parts to be assembled have unknown shape, size and location, is described. The only knowledge used by the robot to perform the assembly operation is given by a connectivity rule and geometrical constraints concerning parts. Once a set of geometrical features of parts has been extracted by a vision system, applying such rule lets to determine the composition sequence. A suitable sensory apparatus allows to control the whole operation.

KEYWORDS: artificial intelligence / machine vision / robot planning / robotics / sensor integration

1. Introduction

In this paper we present an experimental work realized to investigate some robot capabilities when dealing with unstructured operational environments. Generally, different degrees of uncertainty are present in the robot operation world. In this context we could consider the following situations.

a) The shape and dimension of parts as well as the final assembly are known, while their location on the workplane is unknown. In this case the robot vision system must recognize parts and determine their location and orientation. Then the robot has to plan the composition sequence to reach the final assembly. This implies to define a grasp approach trajectory for the arm, a grasp position for the end-effector, a "collision-free" trajectory and a suitable positioning of the moved part into the assembly to be built. Use of endpoint sensing should be made by the robot in an interactive fashion in order to recover unpredictable error situations.

b) The shape and dimension of parts as well as their locations are not known while the final assembly is. In such a case the vision system must locate the various parts while checking, starting from a general knowledge of the problem, if they are admittable parts to be assembled or not, however without identifying them, since they are not completely known "a priori". In addition to the items examined in the previous case, the planning effort implies to determine an assembly sequence of the given parts that matches the goal. Some constraints are to be considered in order to make possible a solution strategy.

c) As an extension of the previous case, not only the shape, dimension and location of parts are unknown but also the assembly goal, meaning that the only "a priori" knowledge consists in a set of possible goals. Besides, some assembly constraints and rules should be considered. Once

parts have been located and a number of features of them have been extracted by means of vision, the planning system, using the given constraints and rules, must try to match the various possible goals with the given parts.
The described situations correspond to different philosophies for the use of a robot in manufacturing environments. The former is the most usual in industry: parts are known together with the way they are to be assembled. A large amount of *a priori* knowledge mitigates the problem complexity, both for visual recognition and for planning. Conversely, the latter correspond to a case where a number of parts are present and the robot ignores what assembly they belong to. This could be useful in flexible environments (FMS-FAS) where a mix of products can be handled at one time. Obviously, the vision apparatus should give more detailed and accurate information and the planner has to solve a more complicate problem. At the moment there are no industrial applications realized to operate in such a way, primarily because they are not cost effective.
Our experiment has been carried out considering a very simple assembly case. The purpose has been to validate some robot reasoning capabilities in a practical problem. The problem configuration has been such to neglet other essential planning issues, in particular collision avoidance and grasp planning, considered in previous experiments [1][2] .

2. Experiment description

The task the robot must perform is to compose a plane figure starting from some pieces placed on its operation plane. Such pieces are made by a white thin millboard and are placed on a black background. This choice semplifies the vision effort while complicating the planning complexity, as will be explained later. The pieces have been obtained by cutting a millboard polygonal figure, choosen among a set of some similar ones prestored into the robot memory, by means of random straight cuts. Thus the pieces are intrinsically unknown "a priori" to the robot system. Then, such pieces are randomly located onto the operation plane by the oper-

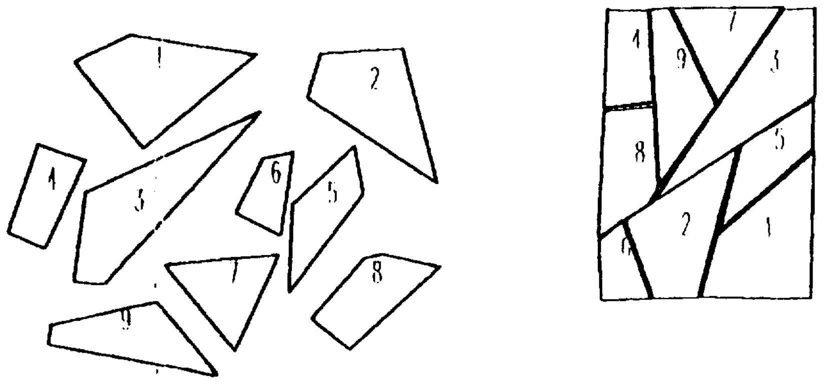


Figure 1. Puzzle assembly example

ator. Obviously, in order for the robot system to find a solution requires that all pieces are consistent with one of stored figures, i.e. all of them should have been generated cutting one such figure. In other words, the robot task consists in manipulating pieces ignoring their number, shape, dimensions, location and what assembly they belong to. It could be considered a *generalized puzzle problem* (fig. 1).

In addition to the previous uncertainties, it should be considered that some pieces can be placed onto the robot plane *overturned* with respect to the upper face of the figure. It is important to note that the millboard has both faces white, so that the *up* and *down* faces of each piece are indistinguishable by means of visual information, tipically by colour. Then, it is only by means of reasoning that the robot should be able to detect a similar situation identifying the initial figure and assembling all pieces in the correct way.

As a preliminary step, the robot must learn the figures it has to reconstruct and store them into the robot memory. Each figure, after beeing placed onto the robot plane, is acquired by the robot camera, then coded into numeric information to be stored in a suitable database. Once some figures have been stored in this way, the robot is able to compose any of them if some pieces are placed, by the right side or overturned, onto the operating plane. As explained above, the only constraint in order for the system to reach a solution, is that the various pieces must be consistent with one figure. In case of inconsistency (the pieces do not belong to anyone of the stored figures or they are less than needed), the system tries to compose a default stored rectangle. If this planning attempt is also unsuccessfull the system fails notifying the event to the operator.

Two further constraints must be considered, regarding the generation and location of the pieces involved in this experience. In particular: a) The pieces must be generated by straight side-to-side cuts: this implies that all pieces are convex polygons and is required by the geometric reasoning procedure. b) The pieces cannot be placed *overlapped*. This should be immediately clear observing that no piece is known "a priori" by the system and then only a complete visual information allows a complete knowledge of its required characteristics. In short, it should be taken into account that the experiment was designed to validate an AI approach to a concrete assembly problem, leaving unsolved some practical issues.

3. Adopted approach

The solution of the described problem is based on a geometrical reasoning approach, since this perfectly matches the problem characteristics. In order to implement the reasoning process it is necessary to traduce the various pieces to assemble into a set of geometric elements. This is accomplished during the vision process, when the image of the various assembly elements is traduced into a sequence of *vectors*, by a process called *vectorization*. Each vector is the representation of an edge side of a polygonal piece. The geometric reasoning operates on such vector sequences applying some geometrical connectivity rule in order to find the assembly sequence (solution). Once this has been found, it is necessary to traduce it into phisical displacement/rotation pairs, in order to perform a correct manipulation at assembly time. So, three classical activity steps can be identified: vision, planning and manipulation. Let us examine separately each one of them.

3.1 Vision

The vision task consists essentially in a low-level phase, by which the polygonal pieces boundaries must be located and traduced into vector sequences. No recognition is made since pieces are unknown. It operates according to the following steps:

- image acquisition and binarization using a suitable threshold
- edge detection by means of a raster-to-vector conversion
- vector postprocessing, to eliminate vectorization artifacts

The raster-to-vector algorithm [3] consists essentially of three phases. (i) A *pre-processing* step, consisting in filling gaps and removing noise from the raw image. (ii) A *boundary tracing* step, which consists in determining boundary points between binary regions. (iii) A *line following* phase, where segment-like regions are transformed into couples of points coordinates (extremes). The raster-to-vector conversion, because of the discrete nature of the image, can be affected by some errors, such as unexisting short sides or adjacent sides with very similar orientation (see fig. 2). Very often such cases are conversion artifacts corresponding to the following phisical situations. (a) A boundary corner is not "seen" as a real tip but as a confused edge region, so that it is converted to a short vector, instead of the cross point between two adjacent vectors. (b) A side, because of its bending, due, for instance, to lens distorsion, is broken into two sides with very similar orientation. In order to filter such artifacts a postprocessing phase has become necessary.

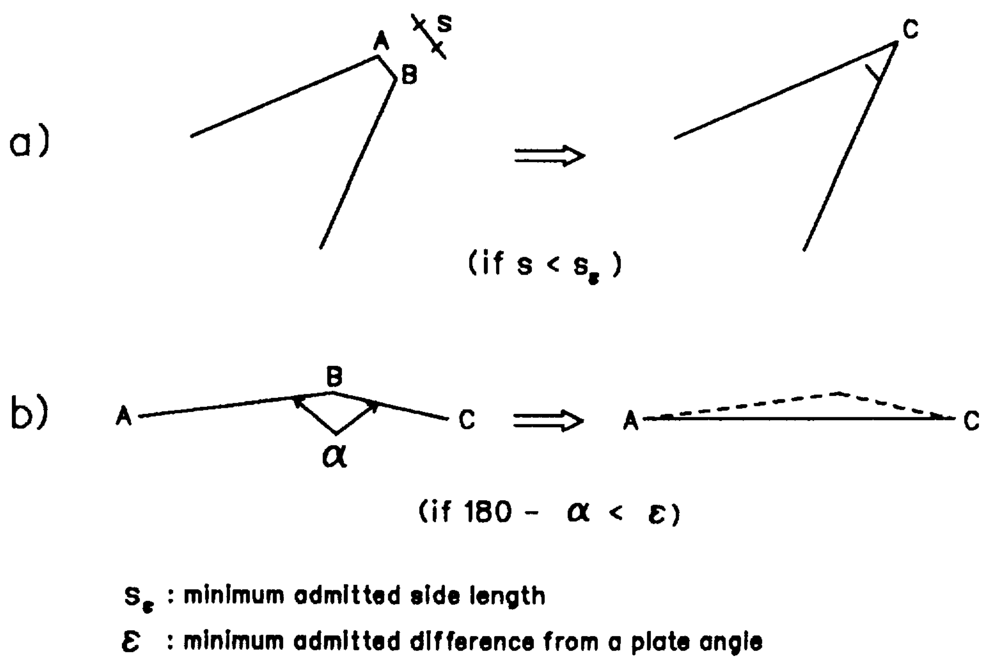

a)
A
s
B
C
(if s < s_ε)
b)
A
B
C
α
A
C
(if 180 - α < ε)
s_ε : minimum admitted side length
ε : minimum admitted difference from a plate angle

Figure 2. Contour extraction: vector postprocessing

Essentially, this operates as follows (see fig. 2):

- vectors too short are eliminated lengthening the two adjacent ones in the sequence until they intersect;
- angles between vectors very near to 180 degrees are eliminated rectifying its sides to obtain a unique vector.

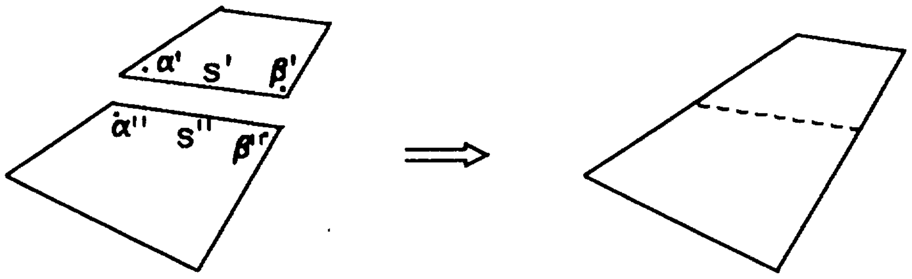
In addition to the previous artifacts, very small white regions can be detected and converted; usually they correspond to spots due to manipulator oil leakage. They produce closed vector sequences with a very small enclosed areas: for each one the postprocessor tests the area value and, in case this is less than a given threshold, erases the whole vector sequence from the world state.

Hence, this phase solves the most vision problems. Anyway, the final vision data (initial world state) are affected by some amount of precision error, due to lens distortion, camera calibration, image resolution, conversion quantization.

3.2 Process planning

As stated before, the "world"representation built by the vision is not completely accurate. This fact should be taken into account by the assembly solution method. For such a reason, an error insensitive connectivity rule is adopted. The global solution strategy is based on the recursive application of such a rule and operates on reduced search spaces. [4] This means that intermediate world states are created during the solution search.

In the following by "polygon" will be denoted a generic piece. The adopted rule allows to determine when two polygons are adjacent along a side in the recomposed figure. In particular, it states what follows: "two polygons are produced by one cut if they have a side of equal length and the angles at the extremes of this side supplementary two by two" (fig. 3). In this way the solution method consists in comparing three couples of values for each couple of sides (one



$$\alpha' + \alpha'' = 180$$
$$\beta' + \beta'' = 180$$
$$s' = s''$$

Figure 3. Connectivity rule and world state update

couple of segments and two couples of angles). Each comparison is made with a prefixed tolerance to take into account errors introduced by the vision system. This makes the process quite error insensitive. The rule is applied to all couples of sides by an exhaustive search among all polygons of the actual world state. When a couple of sides, belonging to different polygons, satisfies the connectivity rule, the planner "adds" the polygons along the common side, building a new abstract polygon and deleting the previous two from the world (fig. 3). This corresponds to update the world state at each recursion. When a unique polygon remains in the world (final world state of the planning process), this is compared with each of the initially stored figures, which are the goal of the planning process. When such a polygon matches, with some prefixed tolerances, one figure in the database, the process is successfully ended and the figure is identified.
Anyway, the process could "fail" at any step. If this happens before a unique polygon has been assembled, it implies that the rule fails for all the couples of polygons actually in the world. This means that an inconsistent set of pieces has been submitted to the robot. Conversely, a fail could also occur when a unique polygon remains in the world state. Generally, this sholud be ascribed to wrong adjacencies, i.e. to couples of sides satisfying the connectivity rule but not arising from physical cuts. Wrong adjacencies are then marked not to repeat, during following searches, wrong branches of the research tree (this mechanism is commonly called *backtracking*). Finally, when all the research tree has been visited without any success, the system identifies an inconsistent set of pieces.
The described problem solver corresponds to an initial implementation. Next, in order to detect overturned polygons on the scene, an enhanced problem solver has been implemented, where the connectivity rule is applied to all polygons in right and overturned configuration (the sides-angles sequences are inverted), tracking for each successful operation the initial condition (right or overturned) of each elementary polygon. This method increases the number of wrong adjacencies and so, requires smaller tolerances. As expected, it results more time consuming than the previous one.

Hence, the result of the entire problem solving process can be:

1. Solution found with right pieces: list of polygon adjacencies
2. Solution found with some pieces overturned: list of polygons to be overturned
3. Solution not found: pieces inconsistent with all the initial figures

As mentioned before, the problem solving process is implemented by two different software modules. The first one is capable to find a solution only when polygons are in right position and can be immediately assembled; it is characterized by a quite fast execution time. The second one, started only when the first fails, can recognize a more complex situation, discriminating between the case of overturned polygons and that of pieces inconsistent with the initial figure database. The latter is of course, much more slower than the former. This software architecture, based on two distinct problem solvers with different capabilities, has been mantained in order to optimize the performance of the whole planning process.
Both modules are written in Prolog language because of the "built-in" backtracking mechanism of such a language.

The planning output, i.e. the list of polygon adjacencies, in order to be used during the physical manipulations, is traduced into a sequence of couples translation+rotation, needed to the manipulator to displace the pieces. This is made by a suitable sequential program, starting from the knowledge of initial location (from vision) and final position (outside the camera field, then predefined) for the whole assembly. An analogous conversion is made for the pieces to be overturned, taking into account the effects of the upsetting operation at manipulation time.

3.3 Manipulation

Starting from the sequence of the displacements and rotations, the actuation module controls the physical handling of the pieces. The puzzle pieces are millboard plates, randomly distributed on the work plane inside to the visual field of the tv camera, while the reassembled figure is constructed by the manipulator on an area outside such a field. The whole pick-and-place of a single piece is made by a particular lifting actuator, a suction cup, which is grasped and held by the manipulator gripper (fig. 4). Such particular actuator is driven in *on-off* mode by the robot controller in order to grasp/release the piece itself. Its operation principle is based on a Venturi tube which generates, when air flows through it, the necessary vacuum for it to operate.
Two critical phases during the described operation are identified. First the picking/release of a piece, the complementary steps where the actuator approaches the workplane surface. In both cases it is necessary to control the motion using the tip force sensors of the gripper to detect the impact with the plane. These are continuously monitored: when the sensed impact reaction force overcomes a given threshold, the motion is stopped and the actuator is switched, on/off, depending on the operation to be performed (picking or releasing). The second critical operation is the upsetting of an overturned piece. This is obtained by means of an experimental fixture realized *ad hoc*, consisting in a kind of vice with two couples of elastic jaws, devoted to

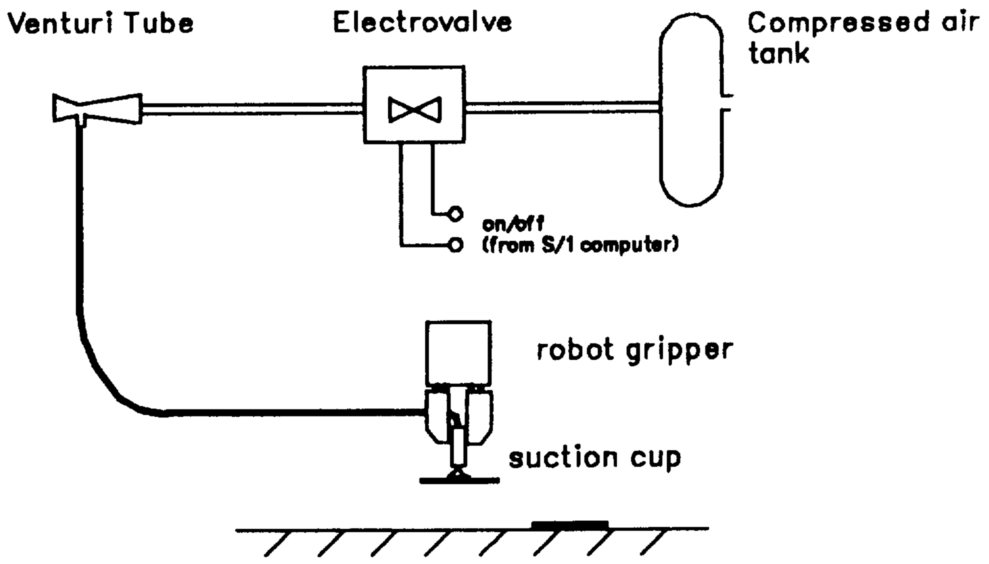


Figure 4. Pickup arrangement

hold the piece to upset while the actuator approaches it by the opposite side. Such structure of the fixture allows the actuator to release the piece, invert its orientation with respect the piece and get it back. The last step is critical. In fact, in order to have a reliable hold of the piece, it is necessary to approach it with a sufficient pressure without deforming it. To obtain this, the actuator must "search" the piece using the tip force sensors.

It should be remarked that sensors are also used to control the actuator grasping. Normally the actuator is fixed on the workplane, in a known position. The gripper approaches it, verifies its real presence by the presence sensor and grasps it controlling the tightening force by the pinch force sensors.

4. Hardware configuration: the robot workstation

The described experiment has been carried out on general purpose robot workstation set up at IBM Rome Scientific Center. This is based on a IBM 7565 robot, which is controlled by a special version of the IBM S/1 minicomputer, integrated with some other machines and computing facilities in order to achieve an adequate power so as flexibility, to develop similar experiments. [2]. Figure 5 shows the overall workstation architecture. The whole station supervision is performed by a personal computer AT. Furthermore, it implements the user

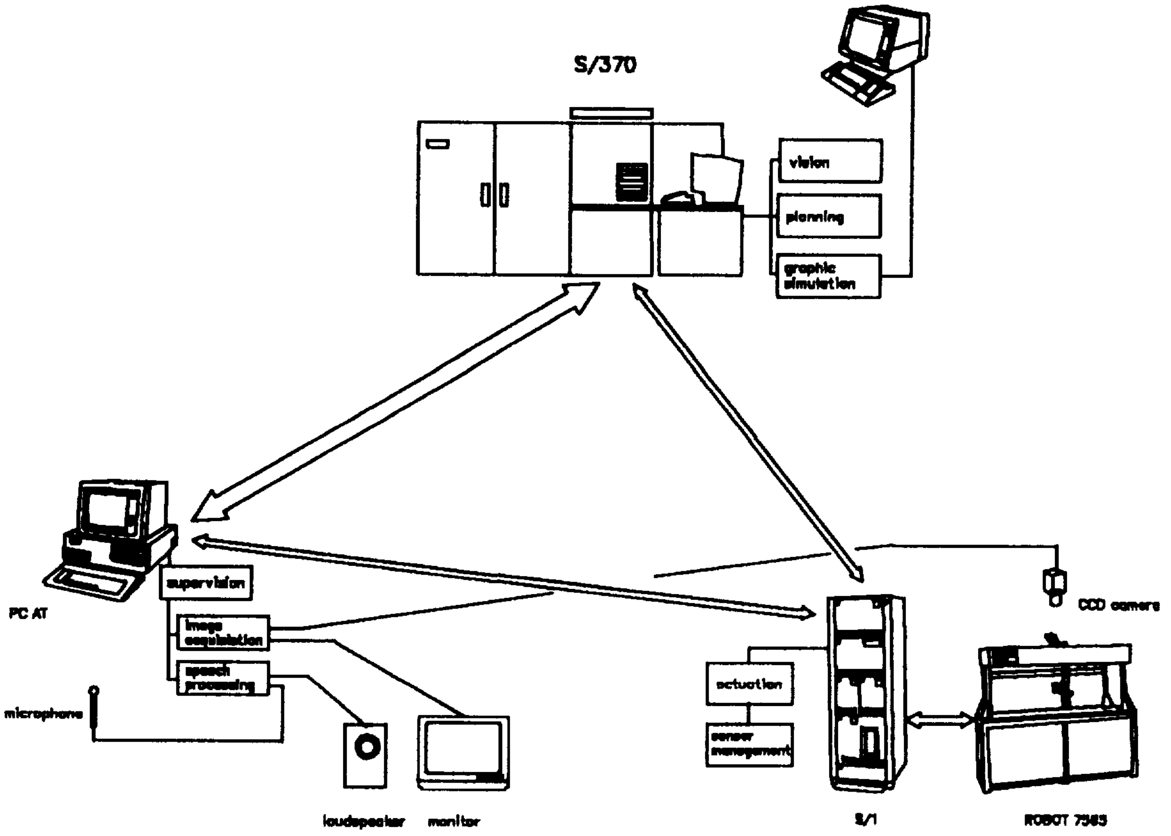


Figure 5. Workstation architecture

interface acting as a system console. Such interface makes use of a speech recognizer and a speech synthesizer. In addition, the PC is devoted to image acquisition and preprocessing for robot vision tasks. The station includes also a S/370 mainframe which is used to perform hard computations such as machine vision, planning tasks and graphic simulations. The three mentioned computer systems are connected together in a network with triangular topology and bidirectional links. In particular, the PC and the mainframe are connected through a S/370 channel attachment to have a fast transfer of large image data sets. The other network links are serial lines, being devoted to more concise data set transfers.

4.1 The manipulator

The IBM 7565 [5] is a cartesian hydraulically powered manipulator, consisting of 6 d.o.f. arm supported by a parallelepiped box frame. Its joints, three prismatic (arm joints, X,Y,Z) and three revolute (wrist joints, *roll, pitch and yaw*), are controlled by analog position servos driven by the robot controller. The gripper is mechanically configured so that the finger surfaces move toward each other remaining parallel. A set of endpoint sensors are mounted in connection with them: three couples of force sensors and a presence sensor. The former are *strain gauges* connected, for each finger, along the three spatial directions. The latter consists in a led-phototransistor pair (*led-beam*) which, once broken by an opaque object located between fingers, lets the manipulator to detect its presence.
The described manipulator is programmed by a special purpose language called A.M.L. ("A Manufacturing Language") [6]. This provides an interactive environment to perform robot motion control, sensor management, data processing and data communication. In the AML environment two different modes are available to process sensors signals. The first one is under program control: sensors are polled and tested by the application program. The second one is an asynchronous, interrupt-like mode; this means that it is possible for the system to detect sensory events (force threshold overcoming, led-beam interrupt ...) in an asynchronous way, interrupt the running AML program at any instant and run a proper user-written AML service routine.

4.2 Image acquisition subsystem

The image acquisition process involves many different hardware and software components. The image is acquired using a CCD camera fixed over the robotic scene and looking downward with the optical axis perpendicular to the robot plane. The camera is attached to the PC via a frame grabber with a resolution of 512 x 512 pels. The acquired image is monochromatic with 256 gray levels: such features have appeared to be adequate in the most 2-D vision experiments carried out until now. In the actual experiment the chromatic resolution is not a critical point, beeing the image thresholded and reduced to a bitmap.
A critical feature of the acquisition is *camera calibration* i.e. the knowledge of the correct correspondence between the world (robotic plane) and the camera coordinate system. First, due to CCD sensor geometry (rectangular) and to the grabbing process (producing a square image), the real scene and its image are not isomorphic. In other words, the spatial passes corresponding to

pixels in the horizontal and vertical directions are not the same. To overcome this problem, the image is stretched in the horizontal direction, by an experimental *stretching factor*.
Besides, the two mentioned coordinate systems have not the same origin, orientation and scale. Thus, to transform a coordinate pair to another it is necessary to determine the proper transformation parameters: this is the goal of the calibration process. Normally, this is made by a linear process, by sensing two different reference points (*calibration posts*) in robot coordinate (mm) and in image coordinate (pels). Such values are used in a linear equation system, whose solution are the reference system change parameters (x_o, y_o (mm) of image origin and k_x and k_y, ratios between pels and mm along x and y). Such process does not take into account the non-linear behavior of lens near edges. This gives an acceptable accuracy in applications not requiring a high precision, while in other applications, like the described one, this is not acceptable.
More accurate results have been obtained applying the same procedure to various couples of posts, placed simultaneously in different points of the scene. For all the couples the required parameters are computed; in this way for each parameter a sample of values is obtained. For each sample mean and mean square error are computed; the final value of each parameter is determined discarding those values outside m.s.e. and computing the mean of remaining. This procedure gives a more "robust" calibration mitigating the effects of lens distorsion. A very accurate calibration procedure is described in [7].

Bibliography.

[1] R. Golini e M. Arlotti **An Intelligent Robot Workstation,** Artificial Intelligence - Implications for CIM, IFS / Springer-Verlag, 1988.

[2] M.A. Arlotti, A.De Castro,V. Di Martino,C.Raspollini,M. Vascotto, **Applicazioni di robotica e visione: esperimenti di manipolazione,** *Robots between Science & Technology SIRI fourth National Conference Proceedings, 211-226.* Milan, March, 21-23, 1988.

[3] IBM Japan Science Institute **Graphical Image Formatting and Translating System User's Guide** Tokio, 1985.

[4] V. Di Martino **Algoritmi di pianificazione per robot** *Atti delle giornate AIRO* Pisa, october 1988.

[5] IBM 7565 Manufacturing System, **Maintenance Information, Circuit Diagrams and Schematics, and Installation,** 1983.

[6] IBM 7565 Manufacturing System, **A Manufacturing Language Reference,** 1983.

[7] R.Y. Tsai, **An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision,** IEEE Conference on Computer Vision and Pattern Recognition, Miami, 1986.

ROBOT KINEMATICS, DYNAMICS AND CONTROL

Recursive Multibody Dynamics and Discrete-Time Optimal Control

G. M. T. D'Eleuterio and C. J. Damaren
Institute for Aerospace Studies
University of Toronto
Downsview, Ontario, Canada M3H 5T6

Abstract

A recursive algorithm is developed for the solution of the simulation dynamics problem for a chain of rigid bodies. Arbitrary joint constraints are permitted, that is, joints may allow translational and/or rotational degrees of freedom. The recursive procedure is shown to be identical to that encountered in a discrete-time optimal control problem. For each relevant quantity in the multibody dynamics problem, there exists an analog in the context of optimal control. The performance index that is minimized in the control problem is identified as Gibbs' function for the chain of bodies.

1 Introduction

The need to predict the motion of robotic systems in terrestial and and space applications has focused attention on the area of multibody dynamics. In this paper, we treat the simulation dynamics of a chain of rigid bodies. Given the external force distribution and control influences acting on the chain, we show how its subsequent motion, namely, the joint accelerations, can be determined using a recursive procedure. The equations of motion and kinematical constraints constitute a two-point boundary value problem. The key to its solution is the elimination of the constraint forces which exist at each joint. Our method in this regard is a generalization of that used by FEATHERSTONE [1983] for single degree of freedom joints, although FEATHERSTONE [1987] has explained how the extension to general constraints can be effected.

Recently, RODRIGUEZ [1987] has pointed out the similarity between the equations describing a chain of hinged bodies and those that arise in discrete-time optimal estimation and smoothing problems. His approach has utilized the correspondence with optimal filtering (the Kalman filter) and smoothing (the Bryson-Frazier smoother). Here, we show that the equations are identical in form to an optimal control problem. In fact, there is a one-to-one correspondence between the elements of the multibody dynamics problem and the control problem. The feedback solution for the control in terms of the state is precisely that which yields the joint accelerations in terms of the body accelerations. The analogy is further uncovered by identifying the performance index (written in terms of the chain dynamics) as GIBBS' [1879] function.

The major benefit of a recursive solution of the simulation problem is its computational efficiency. One avoids dealing with the system of equations describing the system in its entirety. This would involve Gaussian elimination of the global mass matrix at each time step. The computational consequences of this can be quite substantial since the number of calculations involved in a recursive solution grows linearly with the number of bodies whereas the Gaussian elimination obeys a cubic relationship.

2 Equations of Motion

Let us consider a chain of contiguous bodies $\mathcal{B}_0$, $\mathcal{B}_1, \ldots, \mathcal{B}_N$ as shown in Figure 1. Interbody joints may permit arbitrary relative (rotational and/or translational) motion. Each joint therefore possesses at least one degree of freedom and at most six. For convenience, we shall assume interbody translations to be small; however, the extension to large translations can be incorporated into the present formulation. For additional details on the derivation of the equations of motion, the reader should consult SINCARSIN & HUGHES [1989].

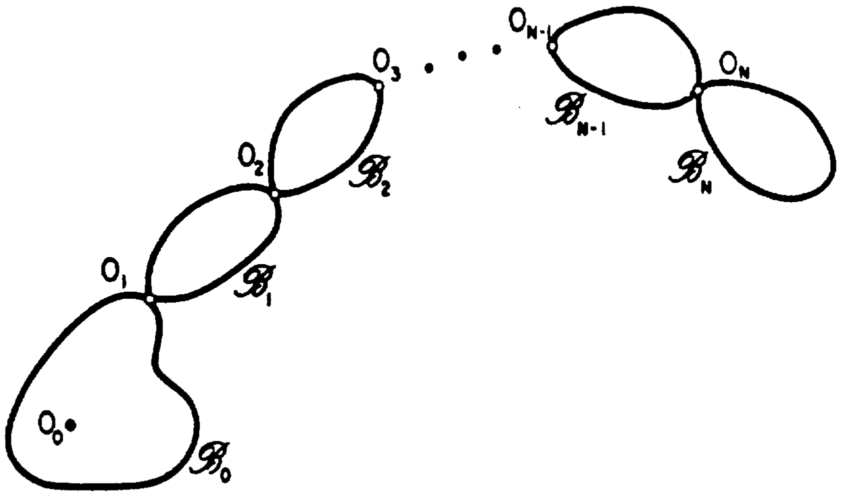


Figure 1: A Chain of Rigid Bodies

The motion of $\mathcal{B}_n$ is defined by the velocity $\mathbf{v}_n$ of O_n and the angular velocity ω_n of $\mathcal{B}_n$. (See Figure 2.) Both $\mathbf{v}_n$ and ω_n are measured with respect to inertial space but are expressed in $\mathcal{F}_n$, a reference frame attached to $\mathcal{B}_n$. We shall define

$$\boldsymbol{v}_n \triangleq \begin{bmatrix} \mathbf{v}_n \\ \omega_n \end{bmatrix} \tag{1}$$

as the *generalized velocity* (*cf.* twist velocity) of $\mathcal{B}_n$ at O_n. We furthermore introduce the accompanying definition for a *generalized force* (*cf.* wrench) acting at O_n:

$$\boldsymbol{f}_n^{n-1} \triangleq \begin{bmatrix} \mathbf{f}_n^{n-1} \\ \mathbf{g}_n^{n-1} \end{bmatrix} \tag{2}$$

where $\mathbf{f}_n^{n-1}$ and $\mathbf{g}_n^{n-1}$ are the reaction forces and torques on $\mathcal{B}_n$ due to $\mathcal{B}_{n-1}$ as expressed in $\mathcal{F}_n$.

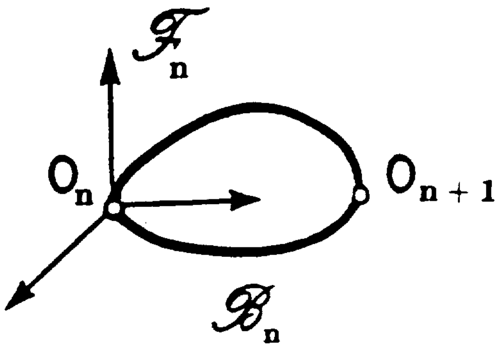


Figure 2: Reference Frame

The resulting equation of motion for $\mathcal{B}_n$ can be written as

$$\mathcal{M}_n \dot{\boldsymbol{v}}_n = \boldsymbol{f}_{nT} + \boldsymbol{f}_{nI} \tag{3}$$

where

$$\mathcal{M}_n \triangleq \begin{bmatrix} m_n\mathbf{1} & -\mathbf{c}_n^\times \\ \mathbf{c}_n^\times & \mathbf{J}_n \end{bmatrix}$$

is the (constant) mass matrix corresponding to $\mathcal{B}_n$, that is, m_n, $\mathbf{c}_n$ and $\mathbf{J}_n$ are the zeroeth (mass), first and second moments of inertia (about O_n) of $\mathcal{B}_n$. Also, $\boldsymbol{f}_{nT}$ is the total external (generalized) force acting on $\mathcal{B}_n$, including interbody forces, and $\boldsymbol{f}_{nI}$, which accounts for the nonlinear inertial terms, can be neatly written as

$$\boldsymbol{f}_{nI} = (\boldsymbol{v}_n^\times)^T \mathcal{M}_n \boldsymbol{v}_n \tag{4}$$

where

$$\boldsymbol{v}_n^\times \triangleq \begin{bmatrix} \boldsymbol{\omega}_n^\times & \mathbf{v}_n^\times \\ \cdot & \boldsymbol{\omega}_n^\times \end{bmatrix}$$

and $(\cdot)^\times$ operating on a Cartesian (3×1) column matrix, such as $\mathbf{v}_n$, $\boldsymbol{\omega}_n$ or $\mathbf{c}_n$, is the matrix equivalent of the vector cross product. In a rate-linear model, one would set $\boldsymbol{f}_{nI} \equiv 0$.

Interbody Constraints

The set of equations (3) does not yet describe a chain of bodies since it does not take into consideration the interbody constraints imposed by the joints. To do so, we begin by observing that

$$\boldsymbol{v}_n = \mathcal{T}_{n,n-1}\boldsymbol{v}_{n-1} + \boldsymbol{v}_{n,\text{int}} \tag{5}$$

which introduces the *relative* interbody generalized velocity $\boldsymbol{v}_{n,\text{int}}$ of $\mathcal{B}_n$ with respect to $\mathcal{B}_{n-1}$. In addition,

$$\mathcal{T}_{n,n-1} = \begin{bmatrix} \mathbf{C}_{n,n-1} & -\mathbf{C}_{n,n-1}\mathbf{r}_{n-1}^{n\,\times} \\ \cdot & \mathbf{C}_{n,n-1} \end{bmatrix}$$

is the *generalized tranformation matrix* between $\mathcal{B}_{n-1}$ and $\mathcal{B}_n$; $\mathbf{C}_{n,n-1}$ is the rotation matrix from $\mathcal{F}_{n-1}$ to $\mathcal{F}_n$ and $\mathbf{r}_{n-1}^n$ is the position of O_n with respect to O_{n-1}. The geometric constraints imposed by the joints can thus be expressed formally as

$$\boldsymbol{v}_{n,\text{int}} = \mathcal{P}_n \boldsymbol{v}_{n\gamma} \tag{6}$$

where $\mathcal{P}_n$ is a projection matrix and $\boldsymbol{v}_{n\gamma}$ is the column of free joint (rate) variables. The absolute velocities $\boldsymbol{v}_n$ can be obtained *recursively* from $\boldsymbol{v}_{n-1}$ and $\boldsymbol{v}_{n\gamma}$.

We also note that

$$\boldsymbol{f}_{nT} = \mathcal{T}_{n+1,n}^T \boldsymbol{f}_{n+1}^n - \boldsymbol{f}_n^{n-1} + \boldsymbol{f}_{n,\text{ext}} \tag{7}$$

where $\boldsymbol{f}_{n,\text{ext}}$ is due to solely to external influences. Furthermore, the generalized interbody forces $\boldsymbol{f}_n^{n-1}$ can be expressed as a sum of control forces $\boldsymbol{f}_{n,\text{c}}$ and constraint forces $\boldsymbol{f}_{n,\square}$, *i.e.*,

$$\boldsymbol{f}_n^{n-1} = -\mathcal{P}_n \boldsymbol{f}_{n,\text{c}} - \mathcal{Q}_n \boldsymbol{f}_{n,\square} \tag{8}$$

The projection matrix $\mathcal{Q}_n$ is the complement of $\mathcal{P}_n$.

Projection Matrices

A few words are perhaps in order regarding the projection matrices. First, as a simple yet very important example, consider a joint with a single rotational degree of freedom about, say, the third axis of an appropriately chosen reference frame. The corresponding projection matrix $\mathcal{P}_n$ is

$$\mathcal{P}_n = [\ 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1\ ]^T$$

We may also add that $\boldsymbol{v}_{n\gamma} = \dot{\gamma}_3$, where γ_3 is the angle of rotation.

In general, $\mathcal{P}_n$ is not constant, as above, but rather is dependent on configuration. Contemplation of a universal joint will quickly reveal this fact. The columns of $\mathcal{P}_n$ are in general not orthonormal but

$$\mathcal{P}_n^T \mathcal{P}_n = \mathcal{I}_n \tag{9}$$

where is $\mathcal{I}_n$ is nonsingular. The complementary projection matrix $\mathcal{Q}_n$ satisfies

$$\mathcal{P}_n^T \mathcal{Q}_n = \mathbf{O} \tag{10}$$

Without loss in generality, the columns of $\mathcal{Q}_n$ can be taken as orthonormal.

Kinematical Equations

The kinematical equations accompanying the dynamical equations (3) can be summarized in terms of $\mathcal{T}_{n,n-1}$:

$$\dot{\mathcal{T}}_{n,n-1} = -\boldsymbol{v}_{n,\text{int}}^{\times} \mathcal{T}_{n,n-1} \tag{11}$$

If we express $\boldsymbol{v}_{n,\text{int}}$ as

$$\boldsymbol{v}_{n,\text{int}} \triangleq \begin{bmatrix} \mathbf{v}_{n,\text{int}} \\ \boldsymbol{\omega}_{n,\text{int}} \end{bmatrix} \tag{12}$$

we can extract from (11),

$$\dot{\mathbf{C}}_{n,n-1} = -\boldsymbol{\omega}_{n,\text{int}}^{\times} \mathbf{C}_{n,n-1} \tag{13}$$

For physical reasons, Euler angles make for the most convenient and expedient representation of rotational joint degrees of freedom. Interbody translation is given by the integration of $\mathbf{v}_{n,\text{int}}$ and would be reflected in $\mathbf{r}_{n-1}^n$.

3 Rate-Linear Simulation Dynamics

The recursive method presented here is a generalization of Featherstone's method applicable to rigid multibody chains with arbitrary interbody constraints. The development, in fact, runs parallel to a similar generalization of Armstrong's recursive method [D'ELEUTERIO 1989]. The essential difference is that the former is based on an affine relationship of the total interbody force to the absolute (generalized) body acceleration while the latter relates explicitly only the interbody constraint force. The generalized Featherstone approach is particularly appealing because of its direct analogy to the discrete-time optimal problem. As shall be demonstrated, however, a simple equivalence exists between the two schemes.

Let us begin, for explanatory purposes, by considering the rate-linear model, that is, we shall set $\boldsymbol{f}_{nI} \equiv 0$ in (3) leaving

$$\mathcal{M}_n \dot{\boldsymbol{v}}_n = \boldsymbol{f}_{nT} \tag{14}$$

The extension to the nonlinear case (and, in fact, to elastic multibody trees) will be straightforward from here, although not totally without some algebraic effort.

Recursion for $\boldsymbol{f}_n^{n-1}$

We conjecture that the interbody forces $\boldsymbol{f}_n^{n-1}$ can be written as

$$-\boldsymbol{f}_n^{n-1} = \boldsymbol{\Psi}_n \dot{\boldsymbol{v}}_n + \boldsymbol{\psi}_n \tag{15}$$

which is a generalization of Featherstone's hypothesis. Note that $\boldsymbol{\Psi}_n$ is, in effect, a mass matrix and $\boldsymbol{\psi}_n$ is a generalized force quantity. The recursive algorithm is based on this result and the fact that $\boldsymbol{\Psi}_n$ and $\boldsymbol{\psi}_n$ can be determined recursively from $\mathcal{B}_N$ to $\mathcal{B}_0$.

The proof of (15) is by induction:

Step I. For $\mathcal{B}_N$, (14) becomes

$$\mathcal{M}_N \dot{\boldsymbol{v}}_N = -\boldsymbol{f}_N^{N-1} + \boldsymbol{f}_{N,\mathrm{ext}} \tag{16}$$

wherein it has been observed that

$$\boldsymbol{f}_{N+1}^{N} \equiv \mathbf{0}$$

since $\mathcal{B}_N$ is the (free) terminal body. It is immediately obvious that if we set

$$\boldsymbol{\Psi}_N = \mathcal{M}_N, \qquad \boldsymbol{\psi}_N = -\boldsymbol{f}_{N,\mathrm{ext}} \tag{17}$$

(15) is satisfied for $n = N$.

Step II. We assume that

$$-\boldsymbol{f}_{n+1}^{n} = \boldsymbol{\Psi}_{n+1}\dot{\boldsymbol{v}}_{n+1} + \boldsymbol{\psi}_{n+1} \tag{18}$$

Step III. Given (18), we shall show that (14) follows. Substituting (7) into (14) yields

$$\mathcal{M}_n \dot{\boldsymbol{v}}_n = \mathcal{T}_{n+1,n}^T \boldsymbol{f}_{n+1}^n - \boldsymbol{f}_n^{n-1} + \boldsymbol{f}_{n,\mathrm{ext}} \tag{19}$$

Now,

$$\boldsymbol{v}_{n+1} = \mathcal{T}_{n+1,n}\boldsymbol{v}_n + \mathcal{P}_{n+1}\boldsymbol{v}_{n+1,\gamma} \tag{20}$$

and

$$\dot{\boldsymbol{v}}_{n+1} = \mathcal{T}_{n+1,n}\dot{\boldsymbol{v}}_n + \mathcal{P}_{n+1}\dot{\boldsymbol{v}}_{n+1,\gamma} \tag{21}$$

(Note that the terms involving the time derivatives of $\mathcal{T}_{n+1,n}$ and $\mathcal{P}_{n+1}$ are omitted since they are nonlinear rate terms.) Substituting (21) and (8) in (19) and premultiplying by $\mathcal{P}_{n+1}^T$ gives

$$\mathcal{I}_{n+1}\boldsymbol{f}_{n+1,\mathrm{c}} = \boldsymbol{\Psi}_{n+1,PP}\dot{\boldsymbol{v}}_{n+1,\gamma} + \mathcal{P}_{n+1}^T\boldsymbol{\Psi}_{n+1}\mathcal{T}_{n+1,n}\dot{\boldsymbol{v}}_n + \boldsymbol{\psi}_{n+1,P} \tag{22}$$

where, in general,

$$\boldsymbol{\Psi}_{nPP} \triangleq \mathcal{P}_n^T\boldsymbol{\Psi}_n\mathcal{P}_n, \qquad \boldsymbol{\psi}_{nP} \triangleq \mathcal{P}_n^T\boldsymbol{\psi}_n$$

Solving for $\dot{\boldsymbol{v}}_{n+1,\gamma}$ from (22), inserting back into (21) and using the result with (18) in (19) eventually leads to

$$\begin{aligned}-\boldsymbol{f}_n^{n-1} &= \{\mathcal{M}_n + \mathcal{T}_{n+1,n}^T(\boldsymbol{\Psi}_{n+1} - \boldsymbol{\Psi}_{n+1}\mathcal{P}_{n+1}\boldsymbol{\Psi}_{n+1,PP}^{-1}\mathcal{P}_{n+1}^T\boldsymbol{\Psi}_{n+1})\mathcal{T}_{n+1,n}\}\dot{\boldsymbol{v}}_n \\ &\quad + \{\mathcal{T}_{n+1,n}^T[\boldsymbol{\Psi}_{n+1}\mathcal{P}_{n+1}\boldsymbol{\Psi}_{n+1,PP}^{-1}(\mathcal{I}_{n+1}\boldsymbol{f}_{n+1,\mathrm{c}} - \boldsymbol{\psi}_{n+1,P}) + \boldsymbol{\psi}_{n+1}] - \boldsymbol{f}_{n,\mathrm{ext}}\}\end{aligned} \tag{23}$$

Hence, we can identify

$$\begin{aligned}\boldsymbol{\Psi}_n &= \mathcal{M}_n + \mathcal{T}_{n+1,n}^T(\boldsymbol{\Psi}_{n+1} - \boldsymbol{\Psi}_{n+1}\mathcal{P}_{n+1}\boldsymbol{\Psi}_{n+1,PP}^{-1}\mathcal{P}_{n+1}^T\boldsymbol{\Psi}_{n+1})\mathcal{T}_{n+1,n} \\ \boldsymbol{\psi}_n &= \mathcal{T}_{n+1,n}^T[\boldsymbol{\Psi}_{n+1}\mathcal{P}_{n+1}\boldsymbol{\Psi}_{n+1,PP}^{-1}(\mathcal{I}_{n+1}\boldsymbol{f}_{n+1,\mathrm{c}} - \boldsymbol{\psi}_{n+1,P}) + \boldsymbol{\psi}_{n+1}] - \boldsymbol{f}_{n,\mathrm{ext}}\end{aligned} \tag{24}$$

Step IV. By induction, then, (15) is proven. □

The matrix $\boldsymbol{\Psi}_n$ has an attractive physical interpretation. It is the mass matrix (about O_n) of the part of the chain from $\mathcal{B}_n$ to $\mathcal{B}_N$ associated with the constrained degrees of freeedom. FEATHERSTONE [1983] would refer to $\boldsymbol{\Psi}_n$ as the *articulated-body inertia*. It should also be pointed out that $\boldsymbol{\Psi}_n$, which is positive-definite, and $\boldsymbol{\psi}_n$ are configuration-dependent.

Recursion for $\dot{\boldsymbol{v}}_{n\gamma}$

By the inductive nature of the proof for (15), it has been shown that the matrices $\boldsymbol{\Psi}_n$ and $\boldsymbol{\psi}_n$ can be evaluated recursively inward, *i.e.*, from $\mathcal{B}_N$ to $\mathcal{B}_0$. Having done so, one can then perform outward recursion, from $\mathcal{B}_0$ to $\mathcal{B}_N$, to solve for $\dot{\boldsymbol{v}}_{n\gamma}$. This is evident from (22).

Rewriting (22) for $\mathcal{B}_n$ instead of $\mathcal{B}_{n+1}$ and solving explicitly for $\dot{\boldsymbol{v}}_{n\gamma}$ yields

$$\dot{\boldsymbol{v}}_{n\gamma} = \boldsymbol{\Psi}_{nPP}^{-1}(\mathcal{I}_n\boldsymbol{f}_{n,\mathrm{c}} - \mathcal{P}_n^T\boldsymbol{\Psi}_n\mathcal{T}_{n,n-1}\dot{\boldsymbol{v}}_{n-1} - \boldsymbol{\psi}_{nP}) \tag{25}$$

Examining this result, we see that at $\mathcal{B}_n$ all the quantities on the right-hand side are known since $\dot{\boldsymbol{v}}_{n-1}$ can be computed recursively from its inboard neighbor according to (21).

4 Nonlinear Simulation Dynamics

The extension to the nonlinear case can be had by simply persevering with the nonlinear rate terms in the preceding development. However, there is a much more palatable approach which is also not without significance in computational considerations.

Let [GOLLA 1988]

$$\dot{\boldsymbol{v}}_n = \boldsymbol{a}_n + \boldsymbol{a}_{n,\text{non}} \tag{26}$$

such that

$$\boldsymbol{a}_n = \boldsymbol{\mathcal{T}}_{n,n-1}\boldsymbol{a}_{n-1} + \boldsymbol{\mathcal{P}}_n\dot{\boldsymbol{v}}_{n\gamma} \tag{27}$$

Inserting (27) into (5) and differentiating reveals that we must have

$$\boldsymbol{a}_{n,\text{non}} = \boldsymbol{\mathcal{T}}_{n,n-1}\boldsymbol{a}_{n-1,\text{non}} + \dot{\boldsymbol{\mathcal{T}}}_{n,n-1}\boldsymbol{v}_{n-1} + \dot{\boldsymbol{\mathcal{P}}}_n\boldsymbol{v}_{n\gamma} \tag{28}$$

for (28) to hold. In essence, the acceleration quantities $\boldsymbol{a}_n$ account for the rate-linear effects and $\boldsymbol{a}_{n,\text{non}}$ for the nonlinear effects. Moreover, not only is $\boldsymbol{a}_n$ found recursively (outward) but $\boldsymbol{a}_{n,\text{non}}$ as well.

Upon substitution of (27) into the motion equation (3), we have

$$\boldsymbol{\mathcal{M}}_n\boldsymbol{a}_n = \boldsymbol{f}_{nT} + \boldsymbol{f}_{nI} + \boldsymbol{f}_{n,\text{non}} \tag{29}$$

where

$$\boldsymbol{f}_{n,\text{non}} \triangleq -\boldsymbol{\mathcal{M}}_n\boldsymbol{a}_{n,\text{non}}$$

In fact, we could write (30) as

$$\boldsymbol{\mathcal{M}}_n\boldsymbol{a}_n = \boldsymbol{\mathcal{T}}^T_{n+1,n}\boldsymbol{f}^n_{n+1} - \boldsymbol{f}^{n-1}_n + \boldsymbol{f}_{n,\text{net}} \tag{30}$$

where

$$\boldsymbol{f}_{n,\text{net}} \triangleq \boldsymbol{f}_{n,\text{ext}} + \boldsymbol{f}_{nI} + \boldsymbol{f}_{n,\text{non}}$$

Comparing (31) to (19), we learn that the nonlinear dynamics model is of the identical form as the rate-linear model with $\dot{\boldsymbol{v}}_n$ replaced by $\boldsymbol{a}_n$ and $\boldsymbol{f}_{n,\text{ext}}$ replaced by $\boldsymbol{f}_{n,\text{net}}$. We can therefore apply the results obtained above directly to the nonlinear case.

Recursion for $\boldsymbol{f}^{n-1}_n$

In general, then, for rigid multibody chains

$$-\boldsymbol{f}^{n-1}_n = \boldsymbol{\Psi}_n\boldsymbol{a}_n + \wp_n \tag{31}$$

Note that $\boldsymbol{\Psi}_n$ is the same as before; however,

$$\wp_n = \boldsymbol{\mathcal{T}}^T_{n+1,n}[\boldsymbol{\Psi}_{n+1}\boldsymbol{\mathcal{P}}_{n+1}\boldsymbol{\Psi}^{-1}_{n+1,PP}(\boldsymbol{\mathcal{I}}_{n+1}\boldsymbol{f}_{n+1,c} - \wp_{n+1,P})\wp_{n+1}] - \boldsymbol{f}_{n,\text{net}} \tag{32}$$

with

$$\wp_{nP} \triangleq \boldsymbol{\mathcal{P}}^T_n\wp_n$$

and $\wp_N = -\boldsymbol{f}_{N,\text{net}}$.

Recursion for $\dot{\boldsymbol{v}}_{n\gamma}$

The recursive relation for $\dot{\boldsymbol{v}}_{n\gamma}$ can be expressed as

$$\dot{\boldsymbol{v}}_{n\gamma} = \boldsymbol{\Psi}^{-1}_{nPP}(\boldsymbol{\mathcal{I}}_n\boldsymbol{f}_{n,c} - \boldsymbol{\mathcal{P}}^T_n\boldsymbol{\Psi}_n\boldsymbol{\mathcal{T}}_{n,n-1}\boldsymbol{a}_{n-1} - \wp_{nP}) \tag{33}$$

which reflects (26). It bears mentioning that the kinematical equations remain unchanged.

Relationship to Armstrong's Work

Before proceeding onward, it is worth pointing out that

$$\boldsymbol{\Psi}_n - \boldsymbol{\Psi}_n \mathcal{P}_n \boldsymbol{\Psi}_{nPP}^{-1} \mathcal{P}_n^T \boldsymbol{\Psi}_n = \mathcal{Q}_n \boldsymbol{\Phi}_n \mathcal{Q}^T \tag{34}$$

where

$$\boldsymbol{\Phi}_n = \boldsymbol{\Psi}_{nQQ} - \boldsymbol{\Psi}_{nPQ}^T \boldsymbol{\Psi}_{nPP}^{-1} \boldsymbol{\Psi}_{nPQ}$$

and

$$\boldsymbol{\Psi}_{nPQ} \triangleq \mathcal{P}_n^T \boldsymbol{\Psi}_n \mathcal{Q}_n, \qquad \boldsymbol{\Psi}_{nQQ} \triangleq \mathcal{Q}_n^T \boldsymbol{\Psi}_n \mathcal{Q}_n$$

Showing (34) requires invoking the identity

$$\mathcal{P}_n \mathcal{I}_n^{-1} \mathcal{P}_n^T + \mathcal{Q}_n \mathcal{Q}_n^T = \mathbf{1}$$

By virtue of (34), we can rewrite the first of (24) as

$$\boldsymbol{\Psi}_n = \mathcal{M}_n + \mathcal{T}_{n+1,n}^T \mathcal{Q}_{n+1} \boldsymbol{\Phi}_{n+1} \mathcal{Q}_{n+1}^T \mathcal{T}_{n+1,n} \tag{35}$$

which is a more streamlined expression.

The significance of $\boldsymbol{\Phi}_n$, however, lies in the fact that

$$\boldsymbol{f}_{n,\square} = \boldsymbol{\Phi}_n \mathcal{Q}_n^T \boldsymbol{a}_n + \boldsymbol{\phi}_n \tag{36}$$

where

$$\boldsymbol{\phi}_n = \mathcal{Q}_n^T \boldsymbol{\wp}_n + \boldsymbol{\Psi}_{nPQ}^T \boldsymbol{\Psi}_{nPP}^{-1} (\mathcal{I}_n \boldsymbol{f}_{n,c} - \boldsymbol{\wp}_{nP})$$

This result is equivalent to Armstrong's method for rigid multibody chains with arbitrary joint constraints.

5 A Discrete-Time Optimal Control Problem

Diverting our attention from multibody dynamics momentarily, consider the following optimal control problem: minimize

$$\mathcal{J} = \sum_{k=0}^{N} \frac{1}{2} \mathbf{x}_k^T \mathbf{M}_k \mathbf{x}_k + \mathbf{x}_k^T \mathbf{h}_k - \mathbf{u}_{k-1}^T \mathbf{t}_{k-1} \tag{37}$$

subject to the linear state equation

$$\mathbf{x}_{k+1} = \mathbf{A}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k, \quad \mathbf{x}_{-1} = \mathbf{0} \tag{38}$$

Here, $\mathbf{M}_k$ is a sequence of positive-definite weighting matrices, and $\mathbf{h}_k$ and $\mathbf{t}_k$ are vector weighting sequences. Since $\mathbf{u}_N$ does not influence $\mathbf{x}_k$, $k \le N$, we shall assume that $\mathbf{t}_N = \mathbf{0}$. This problem is slightly different than the standard "linear-quadratic" version that one typically encounters. The cost functional in the present case is linear in the control variable.

Minimizing $\mathcal{J}$ subject to the state equation is a straightforward optimization problem. Introducing the lagrange multiplier or adjoint variable $\boldsymbol{\lambda}_k$, we define the augmented performance index as follows:

$$\mathcal{J}' \triangleq \sum_{k=0}^{N} \frac{1}{2} \mathbf{x}_k^T \mathbf{M}_k \mathbf{x}_k + \mathbf{x}_k^T \mathbf{h}_k - \mathbf{u}_{k-1}^T \mathbf{t}_{k-1} + \boldsymbol{\lambda}_k^T (\mathbf{x}_k - \mathbf{A}_{k-1} \mathbf{x}_{k-1} - \mathbf{B}_{k-1} \mathbf{u}_{k-1}) \tag{39}$$

The necessary conditions for optimality,

$$\frac{\partial \mathcal{J}'}{\partial \boldsymbol{\lambda}_{k+1}} = \frac{\partial \mathcal{J}'}{\partial \mathbf{x}_k} = \frac{\partial \mathcal{J}'}{\partial \mathbf{u}_k} = 0$$

produce the two-point boundary value problem (TPBVP):

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}_k\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k \, , \quad \mathbf{x}_{-1} = \mathbf{0} & (40)\\ \boldsymbol{\lambda}_k &= \mathbf{A}_k^T\boldsymbol{\lambda}_{k+1} - \mathbf{M}_k\mathbf{x}_k - \mathbf{h}_k \, , \quad \boldsymbol{\lambda}_{N+1} = \mathbf{0} & (41)\\ & \mathbf{t}_k = -\mathbf{B}_k^T\boldsymbol{\lambda}_{k+1} & (42)\end{aligned}$$

We have taken $\boldsymbol{\lambda}_{N+1} = \mathbf{0}$, without loss in generality, since $\mathbf{t}_N = \mathbf{0}$. Hence, from (41), $\boldsymbol{\lambda}_N = -\mathbf{M}_N\mathbf{x}_N - \mathbf{h}_N$ which supplies the basis for the inhomogeneous Riccati transformation, sometimes called the sweep method:

$$\boldsymbol{\lambda}_k = -\mathbf{S}_k\mathbf{x}_k - \mathbf{r}_k \tag{43}$$

with $\mathbf{S}_N = \mathbf{M}_N$ and $\mathbf{r}_N = \mathbf{h}_N$. Substituting (43) into the equation for $\mathbf{t}_k$ (42) and replacing $\mathbf{x}_{k+1}$ with the right side of (40), produces the feedback law

$$\mathbf{u}_k = -\mathbf{K}_k\mathbf{x}_k + \mathbf{R}_k^{-1}[\mathbf{t}_k - \mathbf{B}_k^T\mathbf{r}_{k+1}] \tag{44}$$

where

$$\mathbf{R}_k \triangleq \mathbf{B}_k^T\mathbf{S}_{k+1}\mathbf{B}_k \, , \quad \mathbf{K}_k \triangleq \mathbf{R}_k^{-1}\mathbf{B}_k^T\mathbf{S}_{k+1}\mathbf{A}_k$$

The matrix $\mathbf{R}_k$ will be invertible if $\mathbf{B}_k$ is monic and $\mathbf{S}_{k+1}$ is positive-definite. Substituting the sweep solution (43) for $\boldsymbol{\lambda}_k$ and $\boldsymbol{\lambda}_{k+1}$ and using (40) for $\mathbf{x}_{k+1}$ and (44) for $\mathbf{u}_k$ gives

$$\begin{aligned} & [\mathbf{S}_k - \mathbf{A}_k^T(\mathbf{S}_{k+1} - \mathbf{S}_{k+1}\mathbf{B}_k\mathbf{R}_k^{-1}\mathbf{B}_k^T\mathbf{S}_{k+1})\mathbf{A}_k - \mathbf{M}_k]\mathbf{x}_k \\ = \; & -\mathbf{r}_k + (\mathbf{A}_k - \mathbf{B}_k\mathbf{K}_k)^T\mathbf{r}_{k+1} + \mathbf{K}_k^T\mathbf{t}_k + \mathbf{h}_k \end{aligned}$$

Since this must hold for general $\mathbf{x}_k$, the coefficient of $\mathbf{x}_k$ must vanish as well as the right hand side. Hence,

$$\mathbf{S}_k = \mathbf{A}_k^T(\mathbf{S}_{k+1} - \mathbf{S}_{k+1}\mathbf{B}_k\mathbf{R}_k^{-1}\mathbf{B}_k^T\mathbf{S}_{k+1})\mathbf{A}_k + \mathbf{M}_k \tag{45}$$

which is the discrete-time matrix Riccati equation and

$$\mathbf{r}_k = (\mathbf{A}_k - \mathbf{B}_k\mathbf{K}_k)^T\mathbf{r}_{k+1} + \mathbf{K}_k^T\mathbf{t}_k + \mathbf{h}_k \tag{46}$$

We now return to the question of the invertibility of $\mathbf{R}_k$. The definitions of $\mathbf{K}_k$ and $\mathbf{R}_k$ reveal that $(\mathbf{A}_k - \mathbf{B}_k\mathbf{K}_k)^T\mathbf{S}_{k+1}\mathbf{B}_k = \mathbf{0}$ which allows us to write the Riccati equation as

$$\mathbf{S}_k = (\mathbf{A}_k - \mathbf{B}_k\mathbf{K}_k)^T\mathbf{S}_{k+1}(\mathbf{A}_k - \mathbf{B}_k\mathbf{K}_k) + \mathbf{M}_k \tag{47}$$

Since $\mathbf{S}_N = \mathbf{M}_N$ is symmetic and positive-definite, $\mathbf{S}_k$ is symmetric and positive-definite (using backwards induction). Hence, $\mathbf{R}_k$ defined previously is positive-definite and is always invertible.

The optimal control policy can now be summarized as follows: one solves the Riccati equation (45) (or (47)) and the vector equation (46) backwards from $k = N$ to $k = 0$ using the boundary conditions $\mathbf{S}_N = \mathbf{M}_N$ and $\mathbf{r}_N = \mathbf{h}_N$. The optimal control can then be calculated using (44) while propagating the state forward using the state equation (40).

6 Relationship Between Optimal Control and Recursive Dynamics

The TPBVP generated by the previous optimal control problem (40-42) is identical in form to that of the multibody dynamics problem (30), (33), and

$$\mathcal{I}_n \boldsymbol{f}_{n,c} = -\mathcal{P}_n^T \boldsymbol{f}_n^{n-1} \tag{48}$$

which follows from premultiplying (8) by $\mathcal{P}_n^T$ while recognizing (9) and (10). Therefore, we make the following identifications:

$\mathbf{x}_k$	$\longleftrightarrow$	$\boldsymbol{a}_n$	$\boldsymbol{\lambda}_k$	$\longleftrightarrow$	$\boldsymbol{f}_n^{n-1}$
$\mathbf{u}_k$	$\longleftrightarrow$	$\dot{\boldsymbol{v}}_{n+1,\gamma}$	$\mathbf{h}_k$	$\longleftrightarrow$	$-\boldsymbol{f}_{n,\text{net}}$
$\mathbf{A}_k$	$\longleftrightarrow$	$\mathcal{T}_{n+1,n}$	$\mathbf{M}_k$	$\longleftrightarrow$	$\mathcal{M}_n$
$\mathbf{B}_k$	$\longleftrightarrow$	$\mathcal{P}_{n+1}$	$\mathbf{t}_k$	$\longleftrightarrow$	$\mathcal{I}_{n+1}\boldsymbol{f}_{n+1,c}$

Hence, the accelerations $\boldsymbol{a}_n$ are analogous to the states, the interbody forces $\boldsymbol{f}_n^{n-1}$ are analogous to the adjoint states, the joint accelerations $\dot{\boldsymbol{v}}_{n\gamma}$ play the role of the control inputs, and the projection matrices $\mathcal{P}_{n+1}$ take the place of the input matrix $\mathbf{B}_k$. It can be shown that the interbody tranformation matrices $\mathcal{T}_{n+1,n}$ possess the properties of the state transition matrix thus completing the analogy. Comparing the transformation (43) with the generalization of Featherstone's solution (32) allows us to identify

$$\mathbf{S}_k \longleftrightarrow \boldsymbol{\Psi}_n \ , \quad \mathbf{r}_k \longleftrightarrow \wp_n \ , \quad \mathbf{R}_k \longleftrightarrow \boldsymbol{\Psi}_{n+1,PP}$$

We also emphasize that recursion in time (k) has been replaced by spatial recursion (n) at a given instant in time.

Using the above identifications, the performance index $\mathcal{J}$ can be written as

$$\mathcal{J} = \sum_{n=0}^{N} \frac{1}{2} \boldsymbol{a}_n^T \mathcal{M}_n \boldsymbol{a}_n - \boldsymbol{f}_{n,\text{net}}^T \boldsymbol{a}_n - \boldsymbol{f}_{n,c}^T \mathcal{I}_n \dot{\boldsymbol{v}}_{n\gamma}$$

Hence, in the multibody dynamics problem one can minimize $\mathcal{J}$ subject to the kinematical constraint equation (30) to arrive at the defining equations. Compare this with GIBBS' [1879] formulation of the dynamics of a system of N particles with masses m_n, coordinates x_n, y_n, z_n, and subjected to forces X_n, Y_n, Z_n: minimize

$$\sum_{n=1}^{N} \frac{1}{2} m_n (\ddot{x}_n^2 + \ddot{y}_n^2 + \ddot{z}_n^2) - X_n \ddot{x}_n - Y_n \ddot{y}_n - Z_n \ddot{z}_n$$

subject to the kinematical constraints.

In the work of RODRIGUEZ [1987], he points out the similarity between the equations describing a chain of hinged bodies and the TPBVP that arises in discrete-time, optimal estimation and smoothing problems. In his formulation, the bodies in the chain are numbered inwardly (*i.e.*, the tip body is $\mathcal{B}_0$ and the root body is $\mathcal{B}_N$). Here, the numbering of the bodies is outward (the root body is $\mathcal{B}_0$ and the tip body is $\mathcal{B}_N$). With this convention, the equations are rendered ***dual*** to those of Rodriguez. As such, the corresponding discrete-time problem is not one of estimation and smoothing but one of control. It is interesting to note the dual relationships inherent in Rodriguez's work. The role of the state is played by the interbody forces and the adjoint states are the link accelerations, which are a juxtaposition of the results given above. The control torque at each joint plays the role of a measurement of the states whereas we have the joint accelerations acting as 'control inputs'.

7 Summary of the Recursive Algorithm

We now summarize the procedures for determining the motion of the chain of bodies. The control forces, $\boldsymbol{f}_{n,c}(t)$, and external force distribution, $\boldsymbol{f}_{n,\text{ext}}(t)$, are prescribed on the time interval of interest. Beginning with $t = 0$, we proceed as follows:

Step 1. At time t, the relative velocities $\boldsymbol{v}_{n\gamma}(t)$ and the rotation matrices $\mathrm{C}_{n,n-1}(t)$ are known.

Step 2. Outward recursion for the velocities $\boldsymbol{v}_n$ and determination of $\boldsymbol{f}_{n,\text{net}}$:

```
Do $n = 0$ to $N$;
        Generate $\mathcal{T}_{n,n-1}$ using $\mathrm{C}_{n,n-1}$.
        $\boldsymbol{v}_{n,\text{int}} = \mathcal{P}_n \boldsymbol{v}_{n\gamma}$.
        $\dot{\mathcal{T}}_{n,n-1} = -\boldsymbol{v}_{n,\text{int}}^{\times} \mathcal{T}_{n,n-1}$.
        $\boldsymbol{v}_n = \mathcal{T}_{n,n-1} \boldsymbol{v}_{n-1} + \boldsymbol{v}_{n,\text{int}}$.
        $\boldsymbol{a}_{n,\text{non}} = \mathcal{T}_{n,n-1} \boldsymbol{a}_{n-1,\text{non}} + \dot{\mathcal{T}}_{n,n-1} \boldsymbol{v}_{n-1} + \dot{\mathcal{P}}_n \boldsymbol{v}_{n\gamma}$.
        $\boldsymbol{f}_{nI} = (\boldsymbol{v}_n^{\times})^T \mathcal{M}_n \boldsymbol{v}_n$, $\boldsymbol{f}_{n,\text{non}} = -\mathcal{M}_n \boldsymbol{a}_{n,\text{non}}$.
        $\boldsymbol{f}_{n,\text{net}} = \boldsymbol{f}_{n,\text{ext}} + \boldsymbol{f}_{nI} + \boldsymbol{f}_{n,\text{non}}$.
        Next n.
```

```
Step 3. Inward Recursion for $\boldsymbol{\Psi}_n$ and $\boldsymbol{\wp}_n$:
        Set $\boldsymbol{\Psi}_N = \boldsymbol{\mathcal{M}}_N$ and $\boldsymbol{\wp}_N = -\boldsymbol{f}_{N,\text{non}}$.
        Do $n = N - 1$ to 0;
            $\boldsymbol{\Psi}_{n+1,PP} = \boldsymbol{\mathcal{P}}_{n+1}^T \boldsymbol{\Psi}_{n+1} \boldsymbol{\mathcal{P}}_{n+1}$, $\boldsymbol{\wp}_{n+1,P} = \boldsymbol{\mathcal{P}}_{n+1}^T \boldsymbol{\wp}_{n+1}$
            $\boldsymbol{\mathcal{K}}_n = \boldsymbol{\Psi}_{n+1,PP}^{-1} \boldsymbol{\mathcal{P}}_{n+1} \boldsymbol{\Psi}_{n+1} \boldsymbol{\mathcal{T}}_{n+1,n}$, $\boldsymbol{\Gamma}_{n+1,n} = \boldsymbol{\mathcal{T}}_{n+1,n} - \boldsymbol{\mathcal{K}}_n \boldsymbol{\mathcal{P}}_{n+1}$.
            $\boldsymbol{\Psi}_n = \boldsymbol{\Gamma}_{n+1,n}^T \boldsymbol{\Psi}_{n+1} \boldsymbol{\Gamma}_{n+1,n} + \boldsymbol{\mathcal{M}}_n$
            $\boldsymbol{\wp}_n = \boldsymbol{\Gamma}_{n+1,n}^T \boldsymbol{\wp}_{n+1} + \boldsymbol{\mathcal{K}}_n^T \boldsymbol{f}_{n+1,\text{c}} - \boldsymbol{f}_{n,\text{net}}$
            Next n.
        If $\boldsymbol{\mathcal{P}}_0 \neq \mathbf{O}$, $\boldsymbol{\Psi}_{0PP} = \boldsymbol{\mathcal{P}}_0^T \boldsymbol{\Psi}_0 \boldsymbol{\mathcal{P}}_0$, $\boldsymbol{\wp}_{0P} = \boldsymbol{\mathcal{P}}_0^T \boldsymbol{\wp}_0$

Step 4. Outward Recursion for $\dot{\boldsymbol{v}}_{n\gamma}$:
        If $\boldsymbol{\mathcal{P}}_0 = \mathbf{O}$ ($\mathcal{B}_0$ is constrained), then $\dot{\boldsymbol{v}}_{0\gamma} = \boldsymbol{a}_0 = \mathbf{0}$
        Otherwise, $\dot{\boldsymbol{v}}_{0\gamma} = \boldsymbol{\Psi}_{0PP}^{-1}[\boldsymbol{f}_{0,\text{c}} - \boldsymbol{\psi}_{0P}]$, $\boldsymbol{a}_0 = \boldsymbol{\mathcal{P}}_0 \boldsymbol{v}_{0\gamma}$.
        Do $n = 1$ to $N$;
            $\dot{\boldsymbol{v}}_{n\gamma} = -\boldsymbol{\mathcal{K}}_{n-1} \boldsymbol{a}_{n-1} + \boldsymbol{\Psi}_{n,PP}^{-1}(\boldsymbol{f}_{n,\text{c}} - \boldsymbol{\wp}_{nP})$
            $\dot{\mathbf{C}}_{n,n-1} = -\boldsymbol{\omega}_{n,\text{int}}^{\times} \mathbf{C}_{n,n-1}$
            $\boldsymbol{a}_n = \boldsymbol{\mathcal{T}}_{n,n-1} \boldsymbol{a}_{n-1} + \boldsymbol{\mathcal{P}}_n \dot{\boldsymbol{v}}_{n\gamma}$
            Next n.

Step 5. Estimate $\boldsymbol{v}_{n\gamma}(t + \Delta t)$, $\mathbf{C}_{n,n-1}(t + \Delta t)$ using some quadrature scheme.
        Go back to Step 1 and replace $t$ with $t + \Delta t$.
```

This completes the summary of the recursive simulation procedure. Note that in a rate-linear simulation, one ignores the contributions of $\boldsymbol{f}_{nI}$ and $\boldsymbol{f}_{n,\text{non}}$ to $\boldsymbol{f}_{n,\text{net}}$ in Step 2. We have written the recursion for $\boldsymbol{\Psi}_n$ and $\boldsymbol{\wp}_n$, in Step 3, in terms of the quantities $\boldsymbol{\mathcal{K}}_n$ and $\boldsymbol{\Gamma}_{n+1,n}$ since this leads to the most compact and efficient expressions. The fourth step produces the joint acclerations $\dot{\boldsymbol{v}}_{n\gamma}$ which can be integrated in conjunction with the kinematical relationships for the rotation matrices to produce the joint orientations/positions and velocities.

8 Concluding Remarks

Given the forces on a chain of rigid bodies, we have shown that the accelerations of the bodies can be determined using the recursive procedures of discrete-time optimal control. The underlying analogy that makes this possible yields great insight into the structure of the multibody dynamics problem.

There are many extensions of the present results, a few of which we shall briefly mention here. The analysis presented was limited to topological chains of rigid bodies. It is easily extended to topological tree configurations. The problem of flexible multibody dynamics has been considered by D'ELEUTERIO [1989] who shows that the structure of the equations is unaltered by flexibility. Indeed there is a one-to-one correspondence between the rigid and flexible problems. With this duality in hand, one can readily extend the present analysis to the problem of elastic multibody chains. Such an extension has been performed by DAMAREN & D'ELEUTERIO [1989].

References

ARMSTRONG, W. W.,, "Recursive Solution to the Equations of an n-Link Manipulator", Proc. 5th World Congress on Theory of Machines and Mechanisms, Montreal, 1979, 1343-1346.

DAMAREN, C. J. & D'ELEUTERIO, G. M. T., "On the Relationship Between Discrete-Time Optimal Control and Recursive Dynamics for Elastic Multibody Chains", Proceedings of AMS-SIAM-IMS Summer 1988 Research Conference on Control Theory & Multibody Systems, to be published, 1989.

D'ELEUTERIO, G. M. T., "Dynamics of an Elastic Multibody Chain: Part C – Recursive Dynamics", *Dynamics and Stability of Systems*, to appear, 1989.

FEATHERSTONE, R., "The Calculation of Robot Dynamics Using Articulated-Body Inertias", *Int. J. Robotics Research,* **2**, 1, 13-30, 1983.

FEATHERSTONE, R., *Robot Dynamics Algorithms,* Kluwer Academic Publishers, Boston, 1987.

GIBBS, J. W., "On the Fundamental Formulae of Dynamics", *American Journal of Mathematics, Pure and Applied,* **II**, 1879, 49-64.

GOLLA, D. F., *private communication,* 1988.

HUGHES, P. C. & SINCARSIN, G. B., "Dynamics of an Elastic Multibody Chain: Part B – Global Dynamics", accepted for publication in *Dynamics and Stability of Systems,* 1989.

RODRIGUEZ, G., "Kalman Filtering, Smoothing and Recursive Robot Arm Forward and Inverse Dynamics," *IEEE J. Robot. Autom.,* **RA-3**, 6, 1987, pp. 624-639.

SINCARSIN, G. B. & HUGHES, P. C., "Dynamics of an Elastic Multibody Chain: Part A – Body Motion Equations", accepted for publication in *Dynamics and Stability of Systems,* 1989.

N90-29861

THE EFFECTS OF GEAR REDUCTION ON ROBOT DYNAMICS

J. Chen
Department of Mechanical Engineering
University of Maryland
College Park, MD 20742

Abstract

The effect of the joint drive system with gear reduction for a generic two-link system is studied. It is done by comparing the kinetic energy of such a system with that of a direct drive two-link system. The only difference are two terms involving the inertia of the motor rotor and gear ratio. Modifications of the equations of motion from a direct drive system are then developed and generalized to various cases encountered in robot manipulators.

Introduction

Formulating the equations of motion for a robot is an important part of robot analysis that will provide necessary information for the design of control laws and mechanical components. Before the process of formulation can begin, idealization of the robot system into a model amenable to analysis has to be performed. Assumptions such as rigid bodies, perfect revolute joints, complete isolation between electrical phenomena and mechanical motions and idealized torque transmission in the gear trains are commonly made [1-5]. By removing one or several of these assumptions, one can come up with models with different levels of fidelity. The price to pay is the increased complexity in the resulted equations and possible numerical difficulties. But sometimes, the price has to be paid in order to obtain equations that correspond better with the important characteristics of the actual robot dynamics. In this paper, the effect of some idealizations of joint drive systems in the commonly used model will be investigated.

Although many robot joints are driven by motors through the use of gears with reasonably high (on the order of hundred) reduction ratio, the commonly used model does not include any detail of the drive system. Strictly speaking, the model of the generally used multi-body system is only directly related to a direct drive robot. For this model to be applied to a robot with torque transmission and amplification, certain rules are usually implied. It is generally believed that the torque at the joints are equal to the product of the corresponding motor torques and reduction ratios. It is also known to some that the rotational inertias of the motor rotors when amplified by the corresponding reduction ratios squared should be included in the link inertia to resist motion. These implications can be found in textbooks on automatic control, such as [6], but they are rarely spelled out in robot literature and their validity is not established.

In this paper, a simple model will be used to study the effect of gear reduction on system dynamics. Specifically, the necessary modifications to the equations of motion for the commonly used simply jointed robot model will be presented. An example will also be used to demonstrate the effects of gear reduction to the equations of motion.

System Model

In the present study, it is assumed that the motor rotor is the only massive element in a joint drive system that will contribute to the modifications of the equations of motion from that of a multi-body direct drive system. To study this problem, one might be tempted to just study a particular system by including the rotors in the model and deriving equations for the system. Simulation can then be performed to hopefully reveal the effects caused by the inclusion of rotors in the model. This investigation, however, will be only specific for the particular system simulated and will not shed too much light for a system with very different geometry and mass distribution. The methodology adopted here is to understand the general effects of the rotors and thus to enumerate the suitable modifications of the equations due to them. In place of a specific model, a generic system model should be used and the resulted equations studied to identify and generalize the effects of rotors.

The generic model chosen is shown in Fig. 1. Body C is the carrier of the motor whose rotor together with the connecting shaft and attached gear forms a rigid body R. The driven link D has an attached gear which meshes with the gear in R. This is an idealized system of two links with simple drive system. Since it is assumed that the rotor is the only massive element in the drive system, this model is sufficiently comprehensive for the study. Instead of letting C be a link jointed to the base, it is allowed a general motion relative to the base. This is an intentional choice so that C can be any link of a multi-link system. The system can therefore be considered as a subsystem of an overall system. It can be seen that the linkage in Fig. 1 involves a closed-loop topology and formulations for the commonly studied open-loop multi-link system cannot be applied here. In particular, Newton-Euler formulation is not very convenient for this system and is especially difficult to generalize. Lagrange's or Kane's formulation is more suitable for the present study because generalized active and inertia forces can be considered as being contributed from individual elements of the system. The overall system can be thought of as having n generalized coordinates, $q_1, \ldots, q_n$, and q, the joint angle between C and D, can be one of them.

Formulation of Dynamic Equations

Vector-dyadic formalism will be used in all the following formulations. Here, a vector and a dyadic are defined as abstract entities which are invariant with respect to unit vector bases [7-9]. They can be expressed in terms of any unit vector basis or bases but some operations among them can be reduced without being expressed in any basis. Vectors and dyadics as used in the following are therefore different from column matrices and square matrices, which are just congregates of numbers. However, when the vectors and the dyadics are represented in the same vector basis, their operations can be facilitated by matrix operations.

The first step in Lagrange's formulation is to derive the kinetic energy of the system. For the subsystem in Fig. 1, the contribution to the kinetic energy is

$$K = \frac{1}{2} \sum_{C+R+D} m({}^N v^P)^2 \tag{1}$$

where N is an inertia reference frame, m and ${}^N v^P$ are, respectively, the mass and the velocity in N of a generic particle P in the system, and $\sum_{C+R+D}$ denotes summation over all particles in bodies C, R and D. In the notation for a velocity, an acceleration, an angular velocity or an angular acceleration, a left superscript is used to denote the reference frame the quantity is referred to. In the sequel, when the left superscript is omitted in any of these notations, reference frame N is implied. Applying the theorem called one point moving on a rigid body [9], one can express the velocity of a generic particle P of R or D by

$$v^P = v^{\bar{C}} + {}^C v^P \tag{2}$$

where ${}^C v^P$ is the velocity of P in C, and $v^{\bar{C}}$ is the velocity of $\bar{C}$, which is a point of C that coincides with P at the instant under consideration. Substitution of equation (2) in equation (1) yields

$$K = \frac{1}{2} \{ \sum_{E} m(v^P)^2 + \sum_{R+D} m[({}^C v^P)^2 + 2\, v^{\bar{C}} \cdot {}^C v^P]\} \tag{3}$$

where E is a fictitious rigid body that moves exactly like C but has exactly the same mass distribution as that of C, R and D altogether at the instant under consideration.

With the application of a kinematic theorem for two points fixed on a rigid body [9], one can express the velocity of a generic particle P of E as

$$v^P = v^Q + \omega^C \times r^{QP} \tag{4}$$

where ω^C is the angular velocity of C in N, Q is an arbitrary reference point on C and r^{QP} is the position vector from Q to P. The use of equation (4) brings the first term in the right hand side of equation (3) to the form of

$$\sum_{E} m(v^P)^2 = m_E(v^Q)^2 + \omega^c \cdot I^{E/Q} \cdot \omega^c + 2m_E\, v^Q \cdot (\omega^c \times r^{QE^*}) \tag{5}$$

where

$$I^{E/Q} = \sum_{E} m\, [(r^{QP})^2\, U - r^{QP}\, r^{QP}] \tag{6}$$

and, m_E and E^* are the mass and the mass center of E, respectively, while r^{QE^*} is the position vector from Q to E^*, and U is a unit dyadic. It should be noticed that E^* and $I^{E/Q}$ are not fixed in C. As for the second term in equation (3), the following equations can be similarly derived by suitably choosing a reference point. Firstly,

$$\sum_R m({}^C v^P)^2 = m_R({}^C v^{R^*})^2 + {}^C \omega^R \cdot I^{R/R^*} \cdot {}^C \omega^R \tag{7}$$

where

$$I^{R/R^*} = \sum_R m[(r^{R^*P})^2 U - r^{R^*P} r^{R^*P}] \tag{8}$$

and, m_R and R^* are the mass and the mass center of R, while ${}^C v^{R^*}$ and ${}^C \omega^R$ are the velocity of R^* in C and the angular velocity of R in C, respectively, and r^{R^*P} is the position vector from R^* to P. Next,

$$\sum_D m({}^C v^P)^2 = {}^C \omega^D \cdot I^{D/Q'} \cdot {}^C \omega^D \tag{9}$$

where

$$I^{D/Q'} = \sum_D m[(r^{Q'P})^2 U - r^{Q'P} r^{Q'P}] \tag{10}$$

and, r^{QP} and ${}^C \omega^D$ are the position vector from Q', a point on the joint axis, to P and the angular velocity of D in C, respectively. Then, making use of equation (4) for v^P, one can write

$$\sum_R m\, v^{\bar{C}} \cdot {}^C v^P = v^Q \cdot m_R\, {}^C v^{R^*} + \omega^C \cdot {}^C H^{R/Q} \tag{11}$$

where

$${}^C H^{R/Q} = \sum_R m\, r^{QP} \times {}^C v^P \tag{12}$$

and ${}^C H^{R/Q}$ is the angular momentum of R about Q in C. Similarly,

$$\sum_D m\, v^{\bar{C}} \cdot {}^C v^P = v^Q \cdot m_D\, {}^C v^{D^*} + \omega^C \cdot {}^C H^{D/Q} \tag{13}$$

where

$${}^C H^{D/Q} = \sum_D m\, r^{QP} \times {}^C v^P \tag{14}$$

Since R^* is fixed in C which implies

$${}^C v^{R^*} = 0 \tag{15}$$

and

$$^{C}\omega^{R} = \mu \dot{q} a \tag{16}$$

where μ is the gear ratio, it follows that

$$^{C}H^{R/Q} = \mu J \dot{q} a \tag{17}$$

and

$$^{C}\omega^{R} \cdot I^{R/R^*} \cdot {}^{C}\omega^{R} = \mu^2 J \dot{q}^2 \tag{18}$$

where J is the axial moment of inertia of R. Also since

$$^{C}\omega^{D} = \dot{q} c \tag{19}$$

equation (9) can be rewritten as

$$\sum_{D} m(^{C}v^{P})^2 = \dot{q}^2 (c \cdot I^{D/Q'} \cdot c) \tag{20}$$

Substitution of equations (4)-(18) in equation (3) yields

$$K = \frac{1}{2} \{m_E(v^Q)^2 + \omega^C \cdot I^{E/Q} \cdot \omega^C + \dot{q}^2(\mu^2 J + c \cdot I^{D/Q'} \cdot c)\}$$
$$+ \omega^C \cdot (r^{QE^*} \times m_E v^Q + \mu J \dot{q} a + {}^{C}H^{D/Q}) + m_D v^Q \cdot {}^{C}v^{D*} \tag{21}$$

It can be seen in equation (21) that the kinetic energy of the system involves only two terms, $\frac{1}{2} \mu^2 J\dot{q}^2$ and $\mu J\dot{q}\, \omega^C \cdot a$ that depend exclusively on the motor rotor R. Other contributions to K due to R are lumped in those terms involving the fictitious body E.

Consider now another system S' consisting of only two links C' and D' jointed together as shown in Fig. 2. The kinetic energy of S' will be the same as that in equation (21) less the two terms mentioned above if C, D and E are replaced by C', D' and E', respectively, and if E' is a fictitious body having the mass distribution of C' and D' at the instant under consideration. With perfect rotation of axisymmetric rotor R, the inertia dyadic of C and R together for Q is fixed C. Therefore a real rigid body C' can have the same mass distribution as C and R at all times and have the same motion as C. If this choice is made, and in addition, D' is chosen to have the mass distribution and the motion of D, then E' has the same mass distribution as that of E at all times. The kinetic energies K' and K , respectively, of S' and the original system S consisting of C, D and R can thus be related by

$$K = K' + \frac{1}{2}\mu^2 J \dot{q}^2 + \mu J \dot{q}\, \omega^C \cdot a \qquad (22)$$

It is worth noticing that the common wisdom of simply adding $\mu^2 J$ to the "inertia of the driven link" to compensate for the drive system dynamics is only true when either ω^C, the angular velocity of the carrier of the drive system, or $\omega^C \cdot a$ is zero. For many robots there are some drive systems that do not satisfy either condition.

The differences between generalized inertia force contributions due to S and S' can be worked out based on equation (22). Since generalized inertia force F_r^* is related to kinetic energy K by

$$F_r^* = -\left(\frac{d}{dt}\frac{\partial K}{\partial \dot{q}_r} - \frac{\partial K}{\partial q_r}\right) \quad r = 1, \ldots, n \qquad (23)$$

it follows that

$$(F_r^*)_S = (F_r^*)_{S'} + G_r^* \qquad r = 1, \ldots, n \qquad (24)$$

where

$$G_r^* = \begin{cases} -\mu J(\mu\ddot{q} + \alpha^C \cdot a) & (q_r = q) \\ -\mu J\left\{\ddot{q}\,\dfrac{\partial(\omega^C \cdot a)}{\partial \dot{q}_r} + \dot{q}\left[\dfrac{d}{dt}\dfrac{\partial(\omega^C \cdot a)}{\partial \dot{q}_r} - \dfrac{\partial(\omega^C \cdot a)}{\partial q_r}\right]\right\} & (q_r \neq q) \end{cases} \qquad (25)$$

The partial differentiations in equation (25) are most advantageously performed with C being the reference frame because a is fixed in C. Furthermore, if C is the ith link in an n-link articulated robot as shown in Fig. 3 and D is the subsequent link, then it can be shown that

$$\frac{\partial(\omega^C \cdot a)}{\partial \dot{q}_r} = \begin{cases} z_r \cdot a & (r \le i) \\ 0 & (r > i) \end{cases} \qquad (26)$$

$$\frac{\partial(\omega^C \cdot a)}{\partial q_r} = \frac{{}^C\partial \omega^C}{\partial q_r} \cdot a = \begin{cases} (\omega^{B_r} \times z_r) \cdot a & (r \le i) \\ 0 & (r > i) \end{cases} \qquad (27)$$

and

$$\frac{{}^{C}d}{dt}(z_r \cdot \underline{a}) = ({}^{C}\omega^{B_r} \times z_r) \cdot a = ({}^{B_i}\omega^{B_r} \times z_r) \cdot a \tag{28}$$

With equations (26)-(28) in equation (25), one can rewrite G_r^* as

$$G_r^* = \begin{cases} -\mu J\,(\mu\,\ddot{q}_{i+1} + {}^{N}\alpha^{B_i} \cdot a) & (r = i + 1) \\ -\mu J[(z_r \cdot a)\,\ddot{q}_{i+1} - \dot{q}_{i+1}\,({}^{N}\omega^{B_i} \times z_r) \cdot a] & (r \le i) \\ 0 & (r > i + 1) \end{cases} \tag{29}$$

If the generalized inertia forces of system S' have been worked out separately, then that of S can be derived using equations (24) and (29).

As to the generalized active forces, consider that R is acted upon through electromagnetic or viscous damping interaction by C and the net result of this interaction is a couple of torque Ta. The laws of dynamics dictate that a couple of torque -Ta is also acted on C by R. The contribution of this pair of couples to the generalized active forces F_r are simply

$$F_r = \begin{cases} \mu T & (r = i + 1) \\ 0 & (r \ne i + 1) \end{cases} \tag{30}$$

No other interaction forces between the bodies in S contribute to the generalized active forces. For system S', it is easily seen that if a couple of torque μTc is assumed to act on D' by C', then the contributions of the interaction forces to the generalized active forces are the same as that in equation (30).

Generalization

In some situations, the joint drive system of a particular joint is mounted on the outward link rather than the inward link of the joint, such as that of the second joint of Unimation PUMA robots. The equations derived for S can still be applied with D being the inward link and C being the outward link. Consider that C is still the ith link, and D is the (i-1)th link. The difference terms G_r^* in equation (24) become

$$G_r^* = \begin{cases} -\mu J\,[\mu\, q_i + {}^{N}\alpha^{B_i} \cdot a + \ddot{q}_i\,(z_i \cdot a) - \dot{q}_i({}^{N}\omega^{B_i} \times z_i) \cdot a] & (r = i) \\ -\mu J\,[(z_r \cdot a)\,\ddot{q}_i - \dot{q}_i({}^{N}\omega^{B_i} \times z_r) \cdot a] & (r < i) \\ 0 & (r > i) \end{cases} \tag{31}$$

where unit vector a should be in the direction that is associated in the right hand sense with the rotation of R when the joint experiences a positive rotation. The above generalization is true because all the derivations for the generic system in Fig. 1 do not depend on whether C or D is the preceding link, with the exception of equations (16) and (19). When D is the link that precedes C, unit vectors a and c can be properly changed to maintain the validity of these equations. With this in mind, equation (22) remain valid for the new case. The difference from the original case is that ω^c in equation (22) is a function of q for the present case, and it therefore gives rise to the differences between the expressions in equations (29) and (31).

It is also observed that the above development applies to any axisymmetric body designated R that is carried on C and performs fixed axis rotation in C. Any gear in a gear train connecting a motor to the link it drives can be the rotor R and its contribution to the change of the generalized inertia forces can be identified. It should be noticed that there will be a different gear ratio and a different unit vector a for each gear.

Furthermore, one can see that the only role D plays in equations (29) or (31) is related to the definition of q which is used in equation (16). If the generalized coordinates can be properly introduced for the system and the angular velocity of R in C can be expressed as a function of these generalized coordinates, then the role of D can be eliminated. The above results can thus be extended to the drive system for linear joint. They can also be extended to complicated gear systems that make up many robot wrist mechanisms such as that discussed in [10]. For such a system with three degrees of freedom,

$$^{C}\omega^{R} = (\mu_1\dot{q}_1 + \mu_2\dot{q}_2 + \mu_3\dot{q}_3)\ a \tag{32}$$

where q_1, q_2 and q_3 are the generalized coordinates associated with the system and μ_1, μ_2 and μ_3 are the corresponding ratios for R, should be used instead of equation (16). With this, equation (22) is replaced by

$$K^S = K^{S'} + \frac{1}{2} J(\mu_1\dot{q}_1 + \mu_2\dot{q}_2 + \mu_3\dot{q}_3)^2 + J(\mu_1\dot{q}_1 + \mu_2\dot{q}_2 + \mu_3\dot{q}_3)\ \omega^c \cdot a \tag{33}$$

where S is the system consisting of C and R while S' consists of C' only.

For a complete n degree of freedom motor-driven robot with speed reductions involved, the above procedure can be applied in the following manner:

1. Model the system as made up of n rigid bodies connected by the appropriate linear or revolute joints with each of these bodies having the mass distribution of the actual link and any rotational elements that it carries. Formulate the equations of motion for such a model.

2. For each of the rotational elements, figure out its additional contributions to the kinetic energy and in turn the contributions to the generalized inertia forces as developed above. Add these contributions to the equations of motion.

3. Use equation (30) to work out the generalized active forces.

Significance of Reduction Effects

Dynamic equations for PUMA 560 robot has been explicitly derived with parameter values measured or estimated in [11]. Based on the parameter values listed, the effects of the drive systems can be estimated. For PUMA 560 robots, the 2nd and 3rd joint drive systems are mounted on the 2nd link while the 4th to 6th drive systems are mounted on the 3rd link. The motors are mounted in such a way that their axes of rotation are always perpendicular to the 2nd and the 3rd joint axes. If only the inertia matrix, which is the congregate of the coefficients of $\ddot{q}_i$, i= 1,..,6, and the motor rotor's contributions are considered, the coefficients to have additional terms include the diagonal elements as well as those with indices (1,i), i=2,..., 6, and the off-diagonal elements with indices (4,5), (4,6) and (5,6) due to the coupling of the drive systems of the wrist joints. It is understood that the inertia matrix is a symmetric matrix, and only the elements in the upper triangular part of the matrix are addressed.

Listed in [11] are inertia contributions of joint drive systems to the diagonal elements of the inertia matrix. Here, these are assumed to be mainly contributed by the motor rotors in the form of $\mu^2 J$ pursuant to equations (29) and (30). With this assumption, the constant coefficients of the dominant terms, involving sine and cosine functions of joint angles, of the matrix elements effected can be compared with those additional contributions proportional to μJ as computed based on equations (29) and (30). Table 1 shows the list.

Table 1 Effects of Drive System on Inertia Matrix

Element	1,1	2,2	3,3	4,4	5,5	6,6	1,2	1,3	1,4	1,5	1,6
Dom. Coef. as in [2]	2.57	6.79	1.16	.20	.18	.19	.69	.13	1.64 E-3	1.25 E-3	4. E-5
Coef. of add. term	1.14	4.71	.83	.2	.18	.19	.04	.015	2.60 E-3	2.5 E-3	2.5 E-3

It can be seen from Table 1 that $\mu^2 J$ terms are dominant in the diagonal elements of the inertia matrix. They are included in the equations in [11] while the additional contributions to the off-diagonal elements that are proportional to μJ are not included. Due to the fact that their contributions remain constant when the robot posture is changed, the percentage variations of the diagonal elements is smaller than what it would be if the robot is a direct drive one. This makes fixed gain control more likely to succeed. Although some of the latter ones are dominant, they are still very small compared to the (1,1), or even the (4,4), (5,5) or (6,6) element. As to the elements (4,5), (4,6) and (5,6), since the coupling relationship between the drive systems are not discussed in [11], the additional contributions cannot be estimated. It is reasonable to predict that they will be more significant than those for other off-diagonal elements judging from equation (33).

Contributions due to the motor rotors to those terms second order in $\dot{q}_i$, i=1,.., 6, can also be estimated. Again, some of them may be dominant in the coefficient of a particular term, but their effect to the complete equations may not be significant unless $\dot{q}_1$, i=1,.., 6, assume significantly high values.

Conclusions

Although it has been known to joint drive system designers that the inertia properties of the motor rotor and other elements connected to it are important factors in determining the system dynamic response, it has not been elaborated in so many articles on robot dynamics. The contributions in the form of $\mu^2 J$ in the inertia matrix can dominate some of the matrix elements. Other contributions proportional to μJ are less significant, but they may not be negligible in all cases. With more and very different robots to be developed in the future, it is important to know what need to be included in the dynamic model for it to have sufficient fidelity. The methodology set forth in this paper provides a means to gain the necessary information for a sound judgment.

Acknowledgement

Funding support from NASA GSFC (Grant No. NAG5-1019) for the work reported is greatly appreciated. The technical officer monitoring the grant is John Celmer.

References

1. Paul, R. P., *Robot Manipulator*, The MIT Press, Cambridge, MA, 1981.

2. Vukobratovic, M. and Kircanski, N., *Scientific Fundamentals of Robotics 4*, Springer-Verlag, Berlin, Heidelberg, 1985.

3. Luh, J. Y. S., Walker, M. W., and Paul, R. P. C., "On-Line Computational Scheme for Mechanical Manipulators", ASME Journal of Dynamic Systems, Measurement and Control, Vol. 102, June 1980, pp. 69-76.

4. Walker, M. W. and Orin, D. E., "Efficient Dynamic Computer Simulation of Robotic Mechanisms",Journal of Dynamic Systems, Measurement, and Control, vol. 104, September, 1982.

5. Hollerbach, J. M., "A Recursive Formulation of Lagrangian Manipulator Dynamics", IEEE Transactions of Systems, Man, Cybernetics SMC-10, No. 11, 1980.

6. Ogata, K., *Modern Control Engineering*, Prentice-Hall, Englewood Cliffs, NJ, 1970, pp. 129-132.

7. Wilson, E. B., *Vector Analysis*, Yale University Press, New Haven, 1931.

8. Weatherburn, C. E., *Advanced Vector Analysis*, Open Court Publishing Company, LaSalle, Illinois, 1948.

9. Kane, T. R. and Levinson, D. A., *Dynamics: Theory and Applications*, McGraw-Hill Book Company, 1985.

10. Freudenstein, F., "Kinematic Analysis of Robotic Bevel-Gear Trains," Journal of Mechanisms, Transmissions and Automation in Design, Vol. 106, Sept.

1984, pp. 371-375.

11. Armstrong, B., Khatib, O. and Burdick, J., "The Explicit Dynamic Model and Inertial Parameters of the PUMA 560 Arm," 1986 IEEE International Conference on Robotics and Automation, San Francisco, CA, April, 1986, pp. 510-518.

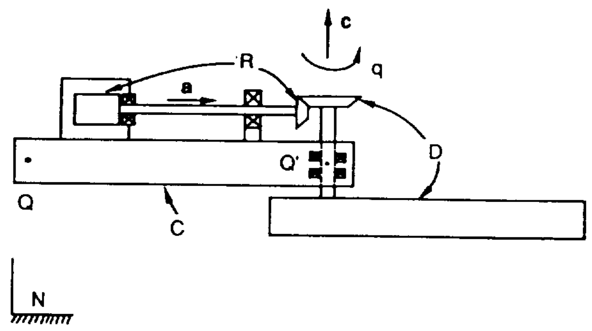
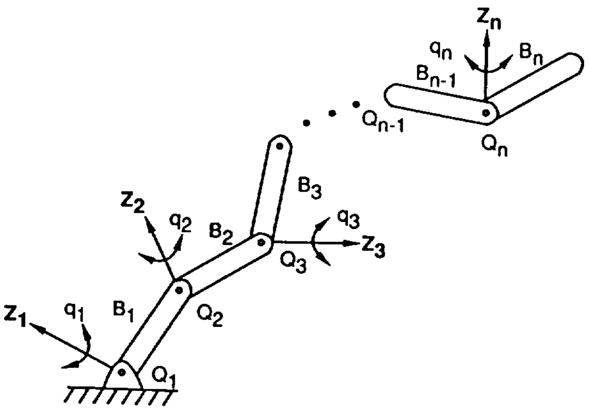


Fig. 1 Two Link System With Gear Reduction



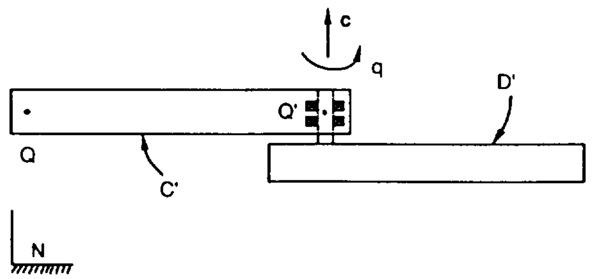


Fig. 3 Multi-Link System

Fig. 2 Two Link Direct Drive System

N90-29862

RECURSIVE NEWTON-EULER FORMULATION OF MANIPULATOR DYNAMICS

M. G. Nasser
Lockheed Engineering & Sciences Company
P. O. Box 58561
Houston, Texas 77258

1. INTRODUCTION

This paper presents a new recursive Newton-Euler procedure for the formulation and solution of manipulator dynamical equations. The procedure includes rotational and translational joints and a topological tree. This model was verified analytically using a planar two-link manipulator. Also, the model was tested numerically against the Walker-Orin (ref. 1) model using the Shuttle Remote Manipulator System data. The hinge accelerations obtained from both models were identical. The computational requirements of the model vary linearly with the number of joints. The computational efficiency of this method exceeds that of Walker-Orin methods.

This procedure may be viewed as a considerable generalization of Armstrong's method (ref. 2). A six-by-six formulation is adopted which enhances both the computational efficiency and simplicity of the model.

In section 2.1, we begin with assuming an open chain, rotational joints, and prescribed base motion. In section 2.2, the procedure is extended to translational joints. Section 2.3 extends the formulation to a topological tree. Section 3 includes the algorithm summary and computational efficiency. The appendix contains descriptions of coordinate frames and notations and a summary of the standard kinematic relations used in the algorithm.

2. DYNAMICS FORMULATION

Let's begin with a quick look at the procedure. The first step is to set up the equations of motion for a generic link i (rotational) in the $i - 1$ frame in a 6×6 form; namely, $S_i U_i = F_i^*$. U_i is a 6×1 vector consisting of the reaction loads from link $i - 1$ on link i and $\ddot{\theta}_i$, the hinge acceleration of link i. S_i is a coefficient matrix, and F_i^* consists of the mass and inertia of link i (inertial parameters) acting on the inertial motion of the $i - 1$ frame, nonlinear terms, body forces and torques, control torques, and reaction loads between link i and link $i + 1$.

The procedure consists essentially of two phases, the inbound and the outbound. In the inbound phase, one begins at the free end, $i = N$. Since there is no outbound link, the reaction loads from link N on link $N + 1$ are zero. Therefore, F^*_N is given by $F^*_N = A_{N,N-1}\, q_{N-1,N-1} + B_{N,N-1}$ where $A_{N,N-1}$ involves only link N inertial parameters.

Now $U^R{}_{N-1,N}$ [equation (2.1.7.1)] may be solved for in terms of $S_N{}^{-1}, A_{N,N-1}, q_{N-1,N-1}$, and $B_{N,N-1}$ but not $\ddot{\theta}_N$. Now we are ready to proceed to link $N - 1$ and substitute $U^R{}_{N-1,N}$. However, $U^R{}_{N-1,N}$ must be transformed to the $N - 2$ frame first. This transformation results in decomposing $(U^R{}_{N-1,N})_{N-2}$ into three terms: the first involving $\ddot{\theta}_{N-1}$; the second, $q_{N-2,N-2}$; and the third, a collection of nonlinear and forcing terms. This decomposition enables one to group these terms with their counterparts from link $N - 1$. The resulting equation of motion is

$$L_{N-1} U_{N-1} = F_{N-1}$$

Note that in this equation of motion for link $N - 1$, $\ddot{\theta}_N$ does not appear, only $\ddot{\theta}_{N-1}$, $\ddot{\theta}_{N-2}$, etc. Repeating the procedure by solving for $U^R_{i-1,i}$ for $i = N - 2$, $N - 3$, ..., we finally obtain the equation containing the hinge acceleration of the base link only.

For the outbound pass, beginning at the base link, link 2, we compute $\ddot{\theta}_2$, $(\dot{\vec{v}}_2)_2$, and $(\dot{\vec{\omega}}_2)_2$, then proceed to link 3 to compute $\ddot{\theta}_3$, $(\dot{\vec{v}}_3)_3$, and $(\dot{\vec{\omega}}_3)_3$, and so on to obtain all hinge accelerations.

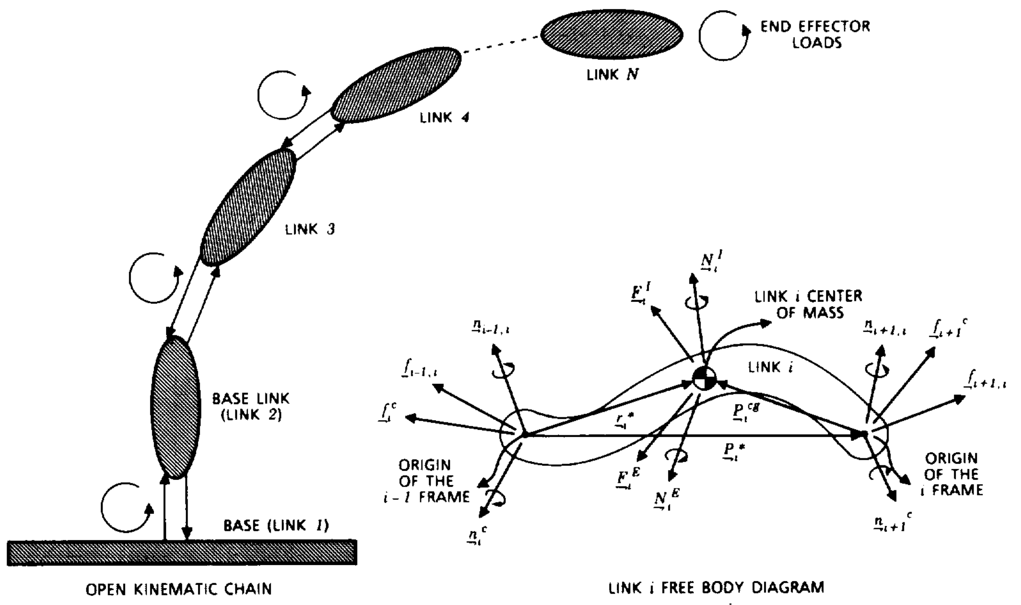
Now we proceed with a detailed description of the model.

2.1 MANIPULATOR WITH ROTATIONAL JOINTS

2.1.1 INBOUND PASS

The translational equation of motion for the center of mass of link i in the $i - 1$ frame is (see figure 2-1 and the appendix)

$$\sum \vec{F}_i = \left(\frac{d\vec{p}_i}{dt}\right)_{i-1} = m_i\left(\dot{\vec{v}}_{i-1} + \vec{\omega}_i \times \left(\vec{\omega}_i \times \vec{r}_i^*\right) + \left(\dot{\vec{\omega}}_{i-1} + \vec{\omega}_{i-1} \times \vec{z}_{i-1}\dot{\theta}_i + \vec{z}_{i-1}\ddot{\theta}_i\right) \times \vec{r}_i^*\right) \qquad (2.1.1)$$



Definitions:

$(\vec{F}_i^I)_{i-1}$	=	the inertia forces developed in link i in the $i - 1$ frame
$(\vec{N}_i^I)_{i-1}$	=	the inertia torques developed in link i in the $i - 1$ frame
$(\vec{f}_i^c)_{i-1}$	=	the control forces applied at the proximal joint of link i in the $i - 1$ frame
$(\vec{f}_{i+1}^c)_{i-1}$	=	the control forces applied at the distal joint of link i in the $i - 1$ frame
$(\vec{f}_{i+1,i})_{i-1}$	=	the reaction force exerted on link i by link $i + 1$ expressed in the $i - 1$ frame
$(\vec{n}_{i+1,i})_{i-1}$	=	the reaction moment exerted on link i by link $i + 1$ expressed in the $i - 1$ frame
$(\vec{f}_{i-1,i})_{i-1}$	=	the reaction force exerted on link i by link $i - 1$ expressed in the $i - 1$ frame
$(\vec{n}_{i-1,i})_{i-1}$	=	the reaction moment exerted on link i by link $i - 1$ expressed in the $i - 1$ frame
$(\vec{P}_i^*)_{i-1}$	=	the position vector of the i frame relative to the $i - 1$ frame and expressed in the $i - 1$ frame
$(\vec{n}_{i+1}^c)_{i-1}$	=	the control torques applied at the distal joint of link i in the $i - 1$ frame
$(\vec{n}_i^c)_{i-1}$	=	the control torques applied at the proximal joint of link i in the $i - 1$ frame
$(\vec{F}_i^E)_{i-1}$	=	the external forces applied at the center of mass of link i in the $i - 1$ frame
$(\vec{N}_i^E)_{i-1}$	=	the external torques applied at the center of mass of link i in the $i - 1$ frame

Figure 2-1.

ORIGINAL PAGE IS OF POOR QUALITY

$\Sigma \underset{\to}{F}_i$ is the total force exerted on the center of mass of link i in the $i - 1$ frame. $\underset{\to}{p}_i$ is the linear momentum of the center of mass of link i in the $i - 1$ frame.

$$\sum \underset{\to}{F}_i = \underset{\to}{f}_{i+1,i} + \underset{\to}{f}_{i-1,i} + \underset{\to}{F}_i^E - \underset{\to}{f}_{i+1}^c + \underset{\to}{f}_i^c \tag{2.1.2}$$

Substituting equation (2.1.2) into equation (2.1.1) yields the following translational equation of motion for any link i in the $i - 1$ frame:

$$\underset{\to}{f}_{i-1,i} + m_i\left(\underset{\to}{r}_i^* \times \underset{\to}{z}_{i-1}\ddot{\theta}_i\right) = \underset{\to}{\alpha}_i + \underset{\to}{\beta}_i + \underset{\to}{f}_{i,i+1} - \underset{\to}{f}_i^c - \underset{\to}{f}_{i+1}^c + \underset{\to}{F}_i^E \tag{2.1.3}$$

$$\underset{\to}{\alpha}_i = m_i\left(\underset{\to}{\dot{v}}_{i-1} + \underset{\to}{\dot{\omega}}_{i-1} \times \underset{\to}{r}_i^*\right) \tag{2.1.3.1}$$

$$\underset{\to}{\beta}_i = m_i\left(\underset{\to}{\omega}_i \times \left(\underset{\to}{\omega}_i \times \underset{\to}{r}_i^*\right) + \left(\underset{\to}{\omega}_{i-1} \times \underset{\to}{z}_{i-1}\dot{\theta}_i\right) \times \underset{\to}{r}_i^*\right) \tag{2.1.3.2}$$

The rotational equations of motion for link i in the $i - 1$ frame (torque balance about the proximal joint of link i) are

$$\left(\sum \underset{\to}{N}_i\right)_{i-1} = \frac{d}{dt}\left(\underset{\to}{r}_i^* \times \underset{\to}{p}_i + I_i\underset{\to}{\omega}_i\right)_{i-1} + \left(\underset{\to}{v}_{i-1} \times \underset{\to}{p}_i\right)_{i-1} \tag{2.1.4}$$

or

$$\left(\sum \underset{\to}{N}_i\right)_{i-1} = \underset{\to}{r}_i^* \times m_i\underset{\to}{\dot{v}}_i^{cg} + \left(I_i\right)_{i-1}\underset{\to}{\dot{\omega}}_i + \underset{\to}{\omega}_i \times \left(I_i\right)_{i-1}\underset{\to}{\omega}_i \tag{2.1.5}$$

$$\left(I_i\right)_{i-1} = R_{i-1,i}I_iR_{i,i-1} = J_i \tag{2.1.5.1}$$

$$\left(\sum \underset{\to}{N}_i\right)_{i-1} = \underset{\to}{n}_{i+1,i} + \underset{\to}{n}_{i-1,i} + \underset{\to}{N}_i^t + \underset{\to}{P}_i^* \times \underset{\to}{f}_{i+1,i} \tag{2.1.5.2}$$

$$\underset{\to}{N}_i^t = \underset{\to}{N}_i^E + \underset{\to}{r}_i^* \times \underset{\to}{F}_i^E - \underset{\to}{P}_i^* \times \underset{\to}{f}_{i+1}^c - \underset{\to}{n}_{i+1}^c + \underset{\to}{n}_i^c \tag{2.1.5.3}$$

The rotational equation of motion for arbitrary link i is:

$$\begin{aligned}\underset{\to}{n}_{i-1,i} + m_i\underset{\to}{r}_i^* \times \left(\underset{\to}{r}_i^* \times \underset{\to}{z}_{i-1}\ddot{\theta}_i\right) - J_i\left(\underset{\to}{z}_{i-1}\ddot{\theta}_i\right) &= \underset{\to}{\alpha}_i^* + \underset{\to}{\beta}_i^* + \underset{\to}{r}_i^* \times \underset{\to}{\beta}_i - \underset{\to}{N}_i^t + \underset{\to}{r}_i^* \times \underset{\to}{\alpha}_i \\ &\quad + \underset{\to}{n}_{i,i+1} - \underset{\to}{P}_i^* \times \underset{\to}{f}_{i+1,i}\end{aligned} \tag{2.1.6}$$

$$\underset{\to}{\alpha}_i^* = J_i\underset{\to}{\dot{\omega}}_{i-1} \tag{2.1.6.1}$$

$$\underset{\to}{\beta}_i^* = J_i\left[\underset{\to}{\omega}_{i-1} \times \underset{\to}{z}_{i-1}\dot{\theta}_i\right] + \underset{\to}{\omega}_i \times J_i\underset{\to}{\omega}_i \tag{2.1.6.2}$$

Equations (2.1.3) and (2.1.6) may be combined and written in the following matrix form:

$$S_iU_i = F_i^* \tag{2.1.7}$$

$$U_i = \left[U_{i-1,i}^R \;\; \ddot{\theta}_i\right] = \left[\underset{\to}{f}_{i-1,i}(1) \;\; \underset{\to}{f}_{i-1,i}(2) \;\; \underset{\to}{f}_{i-1,i}(3) \;\; \underset{\to}{n}_{i-1,i}(1) \;\; \underset{\to}{n}_{i-1,i}(2) \;\; \ddot{\theta}_i\right]^T \tag{2.1.7.1}$$

There is no reaction torque in the drive direction.

$$S_i = I - Z_iZ_i^T - A_{i,i-1}Z_iZ_i^T - Z_iZ_i^TJ_i^a \tag{2.1.7.2}$$

where I is a 6×6 identity matrix, Z_i is its last column, and J_i^a is the actuator inertia associated with hinge i.

$$F_i^* = A_{i,i-1}q_{i-1,i-1} + B_{i,i-1} + \begin{bmatrix} U_{i,i+1}^R \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \underset{\to}{P}_i^* \times \underset{\to}{f}_{i,i+1} \end{bmatrix} \tag{2.1.7.3}$$

$$A_{i,i-1} \triangleq \begin{bmatrix} m_i[I] & -m_i[\tilde{\underline{r}}_i^*] \\ m_i[\tilde{\underline{r}}_i^*] & J_i - m_i[\tilde{\underline{r}}_i^*]^2 \end{bmatrix} \tag{2.1.7.4}$$

$$q_{i-1,i-1} \triangleq \begin{bmatrix} (\dot{\underline{v}}_{i-1})_{i-1} \\ (\dot{\underline{\omega}}_{i-1})_{i-1} \end{bmatrix} \tag{2.1.7.5}$$

$$B_{i,i-1} \triangleq \begin{bmatrix} \underline{\beta}_i - \underline{f}_i^c + \underline{f}_{i+1}^c - \underline{F}_i^E \\ \underline{\beta}_i^* + \underline{r}_i^* \times \underline{\beta}_i - \underline{N}_i^t \end{bmatrix} \tag{2.1.7.6}$$

$[I]$ is a 3×3 identity matrix, and $[\tilde{\underline{r}}_i^*]$ is a skew symmetric matrix associated with $\underline{r}_i^*$.

Since the formal structure of equation (2.1.7) has been defined, consider link N (the link at the free end) and make use of the following boundary conditions:

$$\begin{bmatrix} \underline{f}_{N,N+1} \\ \underline{n}_{N,N+1} \end{bmatrix} = \phi = 0 \tag{2.1.8}$$

$$L_N \triangleq S_N \tag{2.1.9}$$

Therefore, equation (2.1.7) applied to link N is

$$F_N = A_{N,N-1} q_{N-1} + B_{N,N-1} \tag{2.1.10}$$

$$G_i \triangleq L_i^{-1} \quad , \quad \forall i = 1,2,\ldots,N \tag{2.1.11}$$

$$U_N = G_N F_N \tag{2.1.12}$$

Although the expression for $\ddot{\theta}_N$ was obtained in equation (2.1.12), $\ddot{\theta}_N$ cannot be computed until $\dot{\underline{v}}_{N-1}$ and $\dot{\underline{\omega}}_{N-1}$ are. Therefore, proceed to link $N-1$ and set up equation (2.1.7) for $i = N-1$.

When transforming $(U^R_{i-1,i})$ into the $i-2$ frame, the following recursive relation is used:

$$\left(q_{i,i}\right)_{i-1} = P_i^T \left(q_{i-1,i-1} + \sigma_{i,i-1} + \sigma^*_{i,i-1}\right) \quad , \quad \forall i = 1,2,\ldots,N \tag{2.1.13}$$

$$\sigma_{i,i-1} = \ddot{\theta}_i Z_i \tag{2.1.13.1}$$

$$\sigma^*_{i,i-1} = \begin{bmatrix} \underline{\omega}_i \times \left(\underline{\omega}_i \times \underline{P}_i^*\right) \\ -\underline{\omega}_{i-1} \times \underline{z}_{i-1} \dot{\theta}_i \end{bmatrix} \tag{2.1.13.2}$$

$$P_i^T = \begin{bmatrix} I & -[\tilde{\underline{P}}_i^*] \\ \phi & I \end{bmatrix} \tag{2.1.13.3}$$

I is a 3×3 identity matrix, and $[\tilde{\underline{P}}_i^*]$ is a skew symmetric matrix associated with $\underline{P}_i^*$.

$$\begin{bmatrix} U^R_{N-1,N} \\ 0 \end{bmatrix}_{N-2} = \left(A_{N,N-1}\right)_{N-2} P^T_{N-1} \left(q_{N-2} + \sigma_{N-1,N-2} + \sigma^*_{N-1,N-2}\right)$$

$$F^*_{N-1} = A_{N-1,N-2}q_{N-2,N-2} + B_{N-1,N-2} + \begin{bmatrix} U^R_{N-1,N} \\ 0 \end{bmatrix}_{N-2} + \begin{bmatrix} \phi \\ \underset{\sim}{P}^*_{N-1} \times \underset{\sim}{f}_{N-1,N} \end{bmatrix}_{N-2}$$

$$\begin{bmatrix} U^R_{N-1,N} \\ 0 \end{bmatrix}_{N-2} = \left[R^*_{N-2,N-1}\gamma G_N A_{N,N-1} R^{*T}_{N-2,N-1}\right]\left(q_{N-1,N-1}\right)_{N-2} + R^*_{N-2,N-1}\gamma G_N B_{N,N-1}$$

$$A_{N,N-2} = R^*_{N-2,N-1}\gamma G_N A_{N,N-1} R^{*T}_{N-2,N-1} \tag{2.1.14}$$

$$A^*_{N,N-2} = P_{N-1} A_{N,N-2} P^T_{N-1} \tag{2.1.15}$$

$$B_{N,N-2} = R^*_{N-2,N-1}\gamma G_N B_{N,N-1} \tag{2.1.16}$$

$$B^*_{N,N-2} = P_{N-1} B_{N,N-2} + A^*_{N,N-2}\sigma^*_{N-1,N-2} \tag{2.1.17}$$

The superscript T denotes the transpose operator.

$$R^*_{i-1,i} = \begin{bmatrix} R_{i-1,i} & \phi \\ \phi & R_{i-1,i} \end{bmatrix} \tag{2.1.18}$$

Obviously, upon substituting for $U^R_{N-1,N}$ into equation (2.1.18) for $i = N-1$, we get

$$F^*_{N-1} = \left(A_{N-1,N-2} + A^*_{N,N-2}\right)q_{N-2,N-2} + B_{N-1,N-2} + B^*_{N,N-2} + A^*_{N,N-2}\sigma_{N-1,N-2}$$

Since $A^*_{N,N-2}\,\sigma_{N-1,N-2}$ is a function of $\ddot{\theta}_{N-1}$ only, it can be moved to the left-hand side to combine with its counterpart from link $N-1$.

Thus, in general, the equation of motion for any link i takes the following form:

$$L_i U_i = F_i \tag{2.1.19}$$

$$L_i = S_i - A^*_{i+1,i-1} Z_i Z_i^T \tag{2.1.19.1}$$

$$F_i = A^*_{i,i-1} q_{i-1,i-1} + B^*_{i,i-1} \tag{2.1.19.2}$$

$$A^*_{i,i-1} = A_{i,i-1} + A^*_{i+1,i-1} \tag{2.1.19.3}$$

$$A^*_{i+1,i-1} = P_i R^*_{i-1,i}\gamma_{i+1} G_{i+1} A^*_{i+1,i} R^{*T}_{i-1,i} P_i^T \tag{2.1.19.4}$$

$$B^*_{i,i-1} = B_{i,i-1} + B^*_{i+1,i-1} \tag{2.1.19.5}$$

$$B^*_{i+1,i-1} = P_i\, B_{i+1,i-1} + A^*_{i+1,i-1}\sigma^*_{i,i-1} \tag{2.1.19.6}$$

$$B_{i+1,i-1} = R^*_{i-1,i}\gamma_{i+1} G_{i+1} B^*_{i+1,i} \tag{2.1.19.7}$$

2.1.2 OUTBOUND PASS

Assume a prescribed base motion. In this case, $\dot{\underset{\sim}{v}}_1$, $\dot{\underset{\sim}{\omega}}_1$, and $(\underset{\sim}{v}_1, \underset{\sim}{\omega}_1)$ are given. First compute F_2 and then solve for $\ddot{\theta}_2$ from the following equation.

$$\ddot{\theta}_2 = Z_2^T G_2 F_2 \tag{2.1.20}$$

Once $\ddot{\theta}_2$ is obtained, $(\dot{\underset{\to}{v}}_2)_2$ and $(\dot{\underset{\to}{\omega}}_2)_2$ can be computed. This completes the outbound computational cycle for the base link. Next we can move on to link 3 and repeat the same sequence - namely, compute F_3, $\ddot{\theta}_3$, $(\dot{\underset{\to}{v}}_3)_3$, $(\dot{\underset{\to}{\omega}}_3)_3$, $\ddot{\theta}_4$, etc., until all hinge accelerations are determined. Then we proceed to the integration phase.

2.2 MANIPULATOR WITH TRANSLATIONAL JOINTS

Some manipulators contain a mixture of translational and rotational joints. The procedure developed in the previous section for rotational joints is still applicable with slight modifications of the expressions involved (using the kinematics for translational link). These expressions include U_i, Z_i, $\sigma_{i,i-1}$, $\sigma_{i,i-1}{}^*$, $\underset{\to}{\beta}_i$, and $\underset{\to}{\beta}_i{}^*$. If we denote these variables by a prime to distinguish them from their rotational counterparts, we get

$$\underset{\to}{\beta}_i' = m_i\left(\underset{\to}{\omega}_{i-1} \times \left(\underset{\to}{\omega}_{i-1} \times \underset{\to}{v}_i^*\right) + 2\,\underset{\to}{\omega}_{i-1} \times \underset{\to}{z}_{i-1}\dot{\theta}_i\right) \tag{2.2.1}$$

$$\underset{\to}{\beta}_i^{*'} = \underset{\to}{\omega}_{i-1} \times J_i\, \underset{\to}{\omega}_{i-1} \tag{2.2.2}$$

$$U_i' = \left[\underset{\to}{f}_{i-1,i}(1) \quad \underset{\to}{f}_{i-1,i}(2) \quad \ddot{\theta}_i \quad \underset{\to}{n}_{i-1,i}(1) \quad \underset{\to}{n}_{i-1,i}(2) \quad \underset{\to}{n}_{i-1,i}(3)\right] \tag{2.2.3}$$

$$B_{i,i-1}' = \begin{bmatrix} \underset{\to}{\beta}_i' - \underset{\to}{f}_i^c - \underset{\to}{f}_{i+1}^c + \underset{\to}{F}_i^E \\ \underset{\to}{\beta}_i^{*'} + \underset{\to}{r}_i^* \times \underset{\to}{\beta}_i' - \underset{\to}{N}_i^t \end{bmatrix} \tag{2.2.4}$$

$$\sigma_{i,i-1}' = \begin{bmatrix} \underset{\to}{z}_{i-1}\ddot{\theta}_i \\ \phi \end{bmatrix} \tag{2.2.5}$$

$$\sigma_{i,i-1}^{'*} = \begin{bmatrix} \underset{\to}{\omega}_{i-1} \times \left(\underset{\to}{\omega}_{i-1} \times \underset{\to}{P}_i^*\right) + 2\,\underset{\to}{\omega}_{i-1} \times \underset{\to}{z}_{i-1}\dot{\theta}_i \\ \phi \end{bmatrix} \tag{2.2.6}$$

$$Z_i' = [0 \;\; 0 \;\; 1 \;\; 0 \;\; 0 \;\; 0]^T \tag{2.2.7}$$

The remaining variables are defined as in the rotational joints case.

Therefore, the equations of motion for any link i may be written in the following form:

$$L_i U_i = F_i$$

where the formulas obtained in the rotational link case still hold. Note that the only distinction between rotational and translational joints is through the use of either $\underset{\to}{\beta}_i$, $\underset{\to}{\beta}_i{}^*$, U_i, $\sigma_{i,i-1}$, $\sigma^*{}_{i,i-1}$, and Z_i for rotational links or $\underset{\to}{\beta}_i{}'$, $\underset{\to}{\beta}_i{}^{*'}$, $U_i{}'$, $\sigma'{}_{i,i-1}$, $\sigma^{*'}{}_{i,i-1}$, and $Z_i{}'$ for translational links.

2.3 TOPOLOGICAL TREE

The case of a manipulator with tree topology does not alter the formulation in a fundamental manner. In fact, only the root links must be treated differently.

Consider the system shown in figure 2-2. For any branch b_i, we can proceed as in the open chain case until the root link is reached. Denote the root link by K; hence,

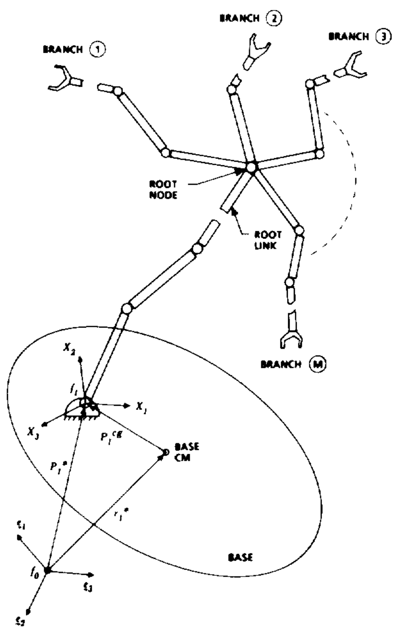


Figure 2-2.

$$\left(U^{R}_{K,K+1}\right)_K = \left(U^{R^1}_{K,K+1}\right)_K + \left(U^{R^2}_{K,K+1}\right)_K + \ldots + \left(U^{R^m}_{K,K+1}\right)_K \tag{2.3.1}$$

Recall that in the open chain case $(U_{i,i+1}{}^{R})_i$ was transformed to the $i - 1$ frame and expanded in terms of $A^*_{i+1,i-1}$, $q_{i-1,i-1}$, and $B^*_{i+1,i-1}$.

Therefore, we get

$$\left(U^{R}_{K,K+1}\right)_{K-1} = \left(U^{R^1}_{K,K+1}\right)_{K-1} + \left(U^{R^2}_{K,K+1}\right)_{K-1} + \ldots + \left(U^{R^m}_{K,K+1}\right)_{K-1} \tag{2.3.2}$$

or

$$A^*_{K,K-1} = A_{K,K-1} + \sum_{j=1}^{m} A^{*j}_{K+1,K-1} \tag{2.3.3}$$

$$B^*_{K,K-1} = B_{K,K-1} + \sum_{j=1}^{m} B^{*j}_{K+1,K-1} \tag{2.3.4}$$

For any j, the definition of $A^{*j}_{K+1,K-1}$ and $B^{*j}_{K+1,K-1}$ is the same as that of the open chain.

3. ALGORITHM SUMMARY AND COMPUTATIONAL EFFICIENCY

3.1 OPEN KINEMATIC CHAIN

Start at the free end, $i = N$.

3.1.1 INBOUND PASS

Repeat the following sequence for $i = N, N - 1, \ldots$:

1. Compute $A^*_{i+1,i-1}$ and $B^*_{i+1,i-1}$ (may be skipped for link N).
2. Compute $A^*_{i,i-1}$ and $B^*_{i,i-1}$.

3. Compute L_i and G_i.
4. $i = i - 1$ and repeat until $i = 2$.

3.1.2 OUTBOUND PASS

Prescribed base motion: $\dot{\underset{\sim}{v}}_1, \dot{\underset{\sim}{\omega}}_1, \underset{\sim}{\omega}_1$, and $\underset{\sim}{v}_1$ are given. Repeat the following steps for $i = 2, 3, \ldots N$.

1. Compute either F_i or $F_i'(i = 1)$.
2. Compute $\ddot{\theta}_i$.
3. Compute $(\dot{\underset{\sim}{\omega}}_i)_i$ and $(\dot{\underset{\sim}{v}}_i)_i$.
4. $i = i + 1$ and repeat steps 1 through 3.

3.2 TOPOLOGICAL TREE

3.2.1 INBOUND PASS

Apply the open kinematic chain procedure to all branches until the base node is reached in this case.

1. Compute $A^{*j}_{K+1,K-1}$ and $B^{*j}_{K+1,K-1}$ or $A^{*j'}_{K+1,K-1}$ and $B^{*j'}_{K+1,K-1}$ for all $j = 1, 2, \ldots, m$ where m is the number of branches at the base node.
2. Compute $A^{*}_{K,K-1}$ and $B^{*}_{K,K-1}$.
3. Repeat steps 2, 3, and 4 as in the open chain unless another is reached; in such case, repeat steps 1 and 2.

3.2.2 OUTBOUND PASS

No change.

3.3 COMPUTATIONAL EFFICIENCY

The number of multiplies is equal to $258N - 119$, and the number of adds is equal to $191N - 83$, where N is the number of links.

4. CONCLUSIONS

A general procedure for the formulation and solution of the equations of motion for a rigid manipulator has been presented. This procedure includes a solution for the tree topology. The extension to a closed kinematic chain follows naturally. However, the presentation of this extension is pending formal implementation and verification.

5. ACKNOWLEDGMENTS

I wish to thank Mr. Ken Hopping for the model implementation and for his suggestions and recommendations. I wish to express my deep appreciation to Mr. Carl Adams for his support throughout the development and verification process - in particular, in taking on the tedious task of verifying the two-link planar manipulator case analytically.

APPENDIX
LINK COORDINATE FRAME AND NOTATION

We adopt a dynamic reference frame. This frame is used here with the Denavit and Hartenberg convention (ref. 3). The joints are points of articulation between links and are numbered such that joint i connects link $i - 1$ and link i. Consequently, joints i and $i + 1$ are the proximal and distal joints, respectively, of link i. Each link i is assigned a Cartesian coordinate frame, (x_i, y_i, z_i), which is fixed on the link and therefore moves with it. (See figure A-1.)

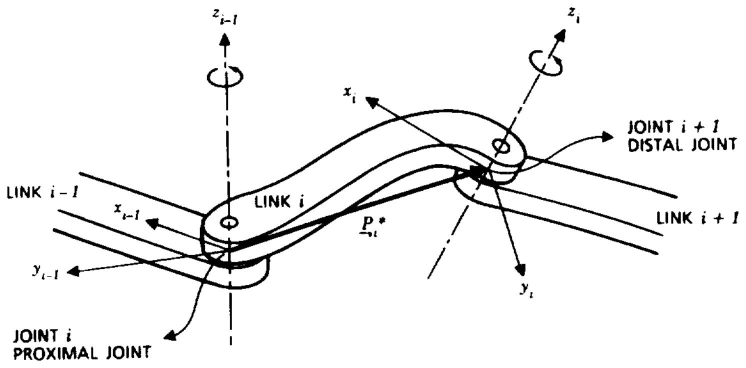


Figure A-1

The $\underset{\rightarrow}{z}_i$ axis is the axis of the rotation/translation of the distal joint of link i. The $\underset{\rightarrow}{x}_i$ axis is directed along the common normal from $\underset{\rightarrow}{z}_{i-1}$ to $\underset{\rightarrow}{z}_i$. The $\underset{\rightarrow}{y}_i$ axis equals $\underset{\rightarrow}{z}_i \times \underset{\rightarrow}{x}_i$ to complete the right-handed system.

In order to associate a particular vector with the coordinate frame, an indexed parenthesis notation is introduced as follows.

$(\underset{\rightarrow}{\theta}_i)_{i-1}$ = the link i relative displacement with respect to and expressed in the $i-1$ frame
$(\underset{\rightarrow}{P}^*_i)_{i-1}$ = the position vector of the i frame relative to and expressed in the $i-1$ frame

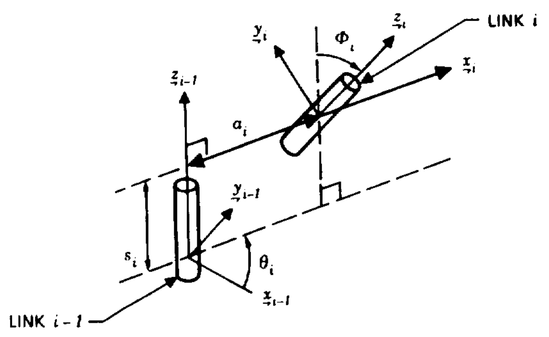
To relate two neighboring coordinate frames, a transformation <u>from</u> the $i-1$ frame <u>to</u> the i frame is defined as successive rotations of θ_i about the $\underset{\rightarrow}{z}_{i-1}$ axis followed by ϕ_i about the x_i axis. (See figure A-2.) This is denoted as

$$R_{i,i-1} = Rot_{x_i}\left(\phi_i\right) Rot_{z_{i-1}}\left(\theta_i\right)$$

$$= \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\cos\phi_i \sin\theta_i & \cos\phi_i \cos\theta_i & \sin\phi_i \\ \sin\phi_i \sin\theta_i & -\sin\phi_i \cos\theta_i & \cos\phi_i \end{bmatrix} \tag{A.1.1}$$

$$R^{-1}_{i,i-1} = R^{T}_{i,i-1} = R_{i-1,i} \tag{A.1.2}$$

$$\left(\underset{\rightarrow}{P}^*_i\right)_{i-1} = \begin{bmatrix} a_i \cos\theta_i \\ a_i \sin\theta_i \\ s_i \end{bmatrix} \tag{A.1.3}$$

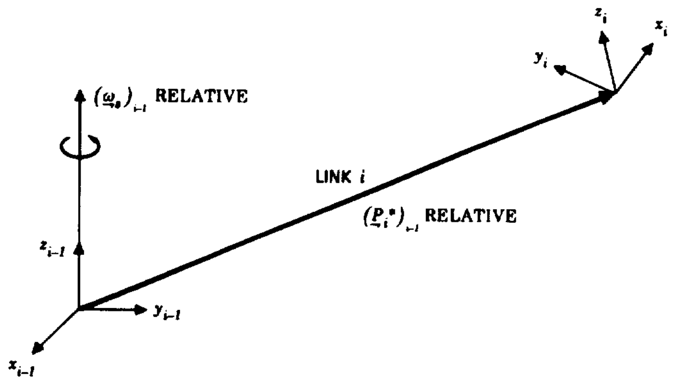


Note: When the z_{i-1} and z_i axes are aligned, it implies that $\theta_i = 0$.

Figure A-2

The following is a set of standard kinematic relations (see figure A-3) for the motion of a rigid body relative to a moving reference frame.

$$\left(\underset{\rightarrow}{\omega}_s\right)_{i-1} = \begin{cases} \underset{\rightarrow}{z}_{i-1}\dot{\theta}_i \\ 0 \end{cases} \quad and \quad \left(\underset{\rightarrow}{\dot{\omega}}_s\right)_{i-1} = \begin{cases} \underset{\rightarrow}{z}_{i-1}\ddot{\theta}_i & \textit{if link i is rotational} \\ 0 & \textit{if link i is translational} \end{cases} \tag{A.2.1}$$



z_i
x_i
y_i
$(\underset{\rightarrow}{\omega}_s)_{i-1}$ RELATIVE
LINK i
$(\underset{\rightarrow}{P}_i^*)_{i-1}$ RELATIVE
z_{i-1}
y_{i-1}
x_{i-1}

Figure A-3

$$\left(\underset{\rightarrow}{\theta}_i\right)_{i-1} = \begin{bmatrix} 0 & 0 & \theta_i \end{bmatrix}^T_{i-1} \tag{A.2.2}$$

$$\left(\underset{\rightarrow}{\dot{\theta}}_i\right)_{i-1} = \left(\underset{\rightarrow}{\omega}_s\right)_{i-1} = \begin{bmatrix} 0 & 0 & \dot{\theta}_i \end{bmatrix}^T_{i-1} \tag{A.2.3}$$

$$\left(\underset{\rightarrow}{\ddot{\theta}}_i\right)_{i-1} = \left(\underset{\rightarrow}{\dot{\omega}}_s\right)_{i-1} = \begin{bmatrix} 0 & 0 & \ddot{\theta}_i \end{bmatrix}^T_{i-1} \tag{A.2.4}$$

$$\left(\underset{\rightarrow}{\omega}_i\right)_{i-1} = \left(\underset{\rightarrow}{\omega}_{i-1}\right)_{i-1} + \left(\underset{\rightarrow}{\omega}_s\right)_{i-1} \tag{A.2.5}$$

REFERENCES

1. Orin, D. E.; and Walker, M. W.: Efficient Dynamic Computer Simulation of Robot Mechanisms. J. Dynamic Systems, Measurement, Control 104, 1982, pp. 205-211.

2. Armstrong, W. W.: Recursive Solution to the Equations of Motion of an *n*-Link Manipulator. Proc. 5th World Congress on Theory of Machines and Mechanisms, Montreal, July 1979, pp. 1343-1346.

3. Denavit, J.; and Hartenberg, R.S.: A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. ASME Journal of Applied Mechanics, June 1955, pp. 215-221.

N90-29833

Kinematic Sensitivity of Robot Manipulators

Marko I. Vuskovic
Department of Mathematical Sciences
San Diego State University
San Diego, CA 92182-0314

Abstract

Kinematic sensitivity vectors and matrices for open-loop, n degrees-of-freedom manipulators are derived. First-order sensitivity vectors are defined as partial derivatives of the manipulator's position and orientation with respect to its geometrical parameters. Considered is the four-parameter kinematic model, as well as the five-parameter model in case of nominally parallel joint axes. Sensitivity vectors are expressed in terms of coordinate axes of manipulator frames. Second-order sensitivity vectors, the partial derivatives of first-order sensitivity vectors, are also considered. It is shown that second-order sensitivity vectors can be expressed as vector products of the first-order sensitivity vectors.

1. Introduction

Sensitivity Theory plays an important role in Systems Theory and in Control Engineering. Its major part, Sensitivity Analysis, is studying the effects of small variations of system parameters on its dynamic behavior and performance criteria. This information can be used for identification of the system's mathematical model, and for the optimal design of the system's controller. Sensitivity theory is also concerned with methods of efficient generation of sensitivity functions in real-time, which can be used for adaptive control. The main results of Sensitivity theory with application in control were obtained in the early sixties. An excellent survey of the Sensitivity Theory at that time is given by two of its important contributors, Kokotovic and Rutman [9]. More recent overviews of the Sensitivity Theory are given by Cruz [3] and Frank [5].

Bearing in mind the substantial influence of the Sensitivity Theory on the development of Control Theory, the question can naturally be raised, whether it can play a similar role in Robotics. In fact, there are many areas in Robotics where sensitivity functions are implicitly used. An example is Robot Calibration, which has been established as an important discipline of Robotics [12], and which can be considered as a counterpart of System Identification, a discipline of System Theory. Finally, one of the most important quantities in Robotics, the manipulator Jacobian matrix, is a sensitivity matrix of the robot position and orientation with respect to joint angles.

The terms "sensitivity", or "kinematic sensitivity" is explicitly used in Robotics by Togai [13]. He has proposed the kinematic sensitivity matrix as a new quantitative measure for the capability for accurate positioning and orienting of a manipulator. This measure is proposed as an attribute complementing Yoshikawa's robot manipulability [15]. Asada and Hara [1] have also defined and analyzed the sensitivity of the actuator torque of the direct-drive arm with respect to the inertial loads. Both papers are dealing with sensitivity only partially, in the context of other problems, and without particular attention given to the problem of computing sensitivity functions in the general case.

This paper considers sensitivity vectors and matrices more generally, although restricted to Robot Kinematics. Sensitivity vectors are defined as partial derivatives of the manipulator's position and orientation with respect to its geometric parameters: link twists, link distances, link offsets and joint angles. Their explicit expressions are derived in terms of link coordinate axes. The case of nominally parallel joint axes, i.e. the five parameter model proposed by Hayati [6-8], is also considered. The sensitivities with respect to link twists about y-axes are derived for this case. Sensitivity vectors, or more precisely, the first-order sensitivity vectors, are then used to derive second-order sensitivity vectors, which are second-order partial derivatives of the manipulator position and orientation with respect to link parameters. It is shown that second-order sensitivity vectors can be entirely expressed in terms of first-order sensitivity vectors. The appendix supplied at the end of the paper reviews the basic formulas from robot kinematics which are used in the derivation of kinematic sensitivities. It also presents an efficient recursive algorithm for computation of link coordinate axes, for both the four and the five parameter models of forward kinematics.

2. Sensitivity vectors and matrices

Position and orientation of an n-DOF manipulator are characterized by its position vector $\mathbf{p} = [\,p_1\, p_2\, p_3\,]^T$ and orientation matrix $\mathbf{R} = [r_{ij}]_3$. These quantities can be referred to any link with respect to any coordinate system. We will consider position and orientation of the n-th link, called the wrist, with the 0-th link, called the base, as a reference coordinate system, i.e. $\mathbf{p} = {}^{o}\mathbf{p}_n$ and $\mathbf{R} = {}^{o}_{n}\mathbf{R}$.

The manipulator geometry is defined by its link parameters. We will use the modified four-parameter Denavit-Hartenberg model as proposed by Craig[2], in which the link parameters are: a_i (link distances), α_i (link twists), d_i (link offsets) and θ_i (joint angles). Therefore vector $\mathbf{p}$ and matrix $\mathbf{R}$ are functions of these parameters:

$$\mathbf{p} = \mathbf{p}(\mathbf{a},\alpha,\mathbf{d},\theta), \qquad \mathbf{R} = \mathbf{R}(\mathbf{a},\alpha,\mathbf{d},\theta) \tag{1}$$

where: $\mathbf{a} = [a_0\ a_1\ \dots a_{n-1}\ ]^T$, $\alpha = [\alpha_0\ \alpha_1 \dots \alpha_{n-1}\ ]^T$, $\mathbf{d} = [d_1\ d_2\ \dots d_n\ ]^T$ and $\theta = [\theta_1\ \theta_2 \dots \theta_n]^T$.

In order to study variations of $\mathbf{p}$ and $\mathbf{R}$ caused by small variations of link parameters we have to consider their partial derivatives $\partial\mathbf{p}/\partial c$ and $\partial\mathbf{R}/\partial c$, where c stands for any of the link parameters. The first partial derivative, $\partial p/\partial c$, we call the *positional sensitivity vector.*

The orientation sensitivity is not so straightforward. The derivative $\partial\mathbf{R}/\partial c$ does not give convenient information about the variation of the manipulator's orientation. It would be more appropriate to express it in terms of three angles instead of a nine-element orientation matrix. Therefore we represent a small change of the manipulator's orientation through three infinitesimal orthogonal rotations $\Delta\varphi = [\Delta\varphi_1\ \Delta\varphi_2\ \Delta\varphi_3]^T$ about the axes of the base coordinate system. This can be written:

$$\mathbf{R}(c+\Delta c) = \Phi(\Delta\varphi)\ \mathbf{R}(c), \tag{2}$$

where

$$\Phi(\Delta\varphi) = \mathbf{rot}(e_3,\Delta\varphi_3)\ \mathbf{rot}(e_2,\Delta\varphi_2\ )\ \mathbf{rot}(e_1,\Delta\varphi_1). \tag{3}$$

The definition of the **rot** operator is given in the Appendix (see (A-4)). For sufficiently small $\Delta\varphi_i$, (3) becomes [11]:

$$\Phi(\Delta\varphi) \approx \mathbf{I} + \Lambda(\Delta\varphi) \tag{4}$$

where $\mathbf{I}$ is the 3×3 identity matrix and $\Lambda(\Delta\varphi)$ is a skew-symmetric operator (see (A-5)). Thus

$$\frac{\partial\mathbf{R}}{\partial c} = \lim_{\Delta c\to 0} \frac{\mathbf{R}(c+\Delta c) - \mathbf{R}(c)}{\Delta c} = \lim_{\Delta c\to 0} \frac{\Lambda(\Delta\varphi)}{\Delta c}\mathbf{R}(c) = \Lambda(\frac{\partial\varphi}{\partial c})\,\mathbf{R}(c), \tag{5}$$

where

$$\frac{\partial\varphi}{\partial c} = \lim_{\Delta c\to 0} \frac{\Delta\varphi}{\Delta c}$$

we denote as the *orientation sensitivity vector.* The relation between the orientation sensitivity vector and the partial derivative of the orientation matrix is given by:

$$\Lambda(\frac{\partial\varphi}{\partial c}) = \frac{\partial\mathbf{R}}{\partial c}\mathbf{R}^{\mathrm{T}}. \tag{6}$$

Sensitivity vectors for various link parameters can be joined together to form the sensitivity matrices:

$$S^{p}_{a} = \left[\frac{\partial p}{\partial a_0} \cdots \frac{\partial p}{\partial a_{n-1}}\right],\quad S^{\varphi}_{a} = \left[\frac{\partial\varphi}{\partial a_0} \cdots ,\frac{\partial\varphi}{\partial a_{n-1}}\right],\quad S^{p}_{\alpha} = \left[\frac{\partial p}{\partial \alpha_0} \cdots \frac{\partial p}{\partial \alpha_{n-1}}\right],\quad S^{\varphi}_{\alpha} = \left[\frac{\partial\varphi}{\partial \alpha_0} \cdots \frac{\partial\varphi}{\partial \alpha_{n-1}}\right]$$

$$S^{p}_{d} = \left[\frac{\partial p}{\partial d_1} \cdots \frac{\partial p}{\partial d_n}\right],\quad S^{\varphi}_{d} = \left[\frac{\partial\varphi}{\partial d_1} \cdots \frac{\partial\varphi}{\partial d_n}\right],\quad S^{p}_{\theta} = \left[\frac{\partial p}{\partial \theta_1} \cdots \frac{\partial p}{\partial \theta_n}\right],\quad S^{\varphi}_{\theta} = \left[\frac{\partial\varphi}{\partial \theta_1} \cdots \frac{\partial\varphi}{\partial \theta_n}\right]. \tag{7}$$

3. Derivation of sensitivity vectors

Positional sensitivity vectors with respect to parameters a_j and d_j can be directly obtained if we express the manipulator position **p** explicitly in terms of these parameters. Such an expression is given in the Appendix. Since coordinate axes are independent of these parameters, (A-8) is giving:

$$\frac{\partial \mathbf{p}}{\partial a_j} = \mathbf{x}_j, \qquad \frac{\partial \mathbf{p}}{\partial d_j} = \mathbf{z}_j. \tag{8}$$

Hence

$$\boxed{\mathbf{S}^P_a = [\mathbf{x}_0\, \mathbf{x}_1 \dots \mathbf{x}_{n-1}] \qquad \mathbf{S}^P_d = [\mathbf{z}_1\, \mathbf{z}_2 \dots \mathbf{z}_n]} \tag{9}$$

In order to derive positional sensitivity vectors with respect to α_j and θ_j, we first find partial derivatives of coordinate axes with respect to these parameters. Combining (A-3) and (A-7) we can write:

$$\frac{\partial}{\partial \alpha_{i-1}} {}^{i-1}_{i}\mathbf{R} = \Lambda(\mathbf{e}_1)\, {}^{i-1}_{i}\mathbf{R}, \qquad \frac{\partial}{\partial \theta_i} {}^{i-1}_{i}\mathbf{R} = {}^{i-1}_{i}\mathbf{R}\, \Lambda(\mathbf{e}_3). \tag{10}$$

Now applying (10) and (A-6) to (A-8) and assuming $j<i$, we obtain:

$$\frac{\partial \mathbf{x}_i}{\partial \alpha_j} = \frac{\partial}{\partial \alpha_j}\left({}^{0}_{j}\mathbf{R}\; {}^{j}_{j+1}\mathbf{R}\; {}^{j+1}_{i}\mathbf{R}\; \mathbf{e}_1\right) = {}^{0}_{j}\mathbf{R}\, \Lambda(\mathbf{e}_1)\, {}^{j}_{i}\mathbf{R}\, \mathbf{e}_1 = \Lambda({}^{0}_{j}\mathbf{R}\mathbf{e}_1)\, {}^{0}_{i}\mathbf{R}\, \mathbf{e}_1 = \Lambda(\mathbf{x}_j)\, \mathbf{x}_i = \mathbf{x}_j \times \mathbf{x}_i.$$

Thus:

$$\frac{\partial \mathbf{x}_i}{\partial \alpha_j} = \begin{cases} -\mathbf{x}_i \times \mathbf{x}_j & j<i \\ 0 & j \geq i \end{cases}. \tag{11}$$

In a similar way we can obtain other partial derivatives:

$$\frac{\partial \mathbf{z}_i}{\partial \alpha_j} = \begin{cases} -\mathbf{z}_i \times \mathbf{x}_j & j<i \\ 0 & j \geq i \end{cases}, \qquad \frac{\partial \mathbf{x}_i}{\partial \theta_j} = \begin{cases} -\mathbf{x}_i \times \mathbf{z}_j & j\leq i \\ 0 & j > i \end{cases}, \qquad \frac{\partial \mathbf{z}_i}{\partial \theta_j} = \begin{cases} -\mathbf{z}_i \times \mathbf{z}_j & j<i \\ 0 & j \leq i \end{cases}. \tag{12}$$

Applying now (11) and (12) to (A-8) we get:

$$\frac{\partial \mathbf{p}}{\partial \alpha_j} = \sum_{i=1}^{n} \frac{\partial \mathbf{x}_{i-1}}{\partial \alpha_j} a_{i-1} + \frac{\partial \mathbf{z}_i}{\partial \alpha_j} d_i = -\sum_{i=j+1}^{n} (\mathbf{x}_{i-1} a_{i-1} + \mathbf{z}_i d_i) \times \mathbf{x}_j. \tag{13}$$

Finally, substituting (A-9) into (13) gives us:

$$\frac{\partial \mathbf{p}}{\partial \alpha_j} = \mathbf{x}_j \times \mathbf{r}_j. \tag{14}$$

Similarly we get:

$$\frac{\partial \mathbf{p}}{\partial \theta_j} = \mathbf{z}_j \times \mathbf{r}_j, \tag{15}$$

thus:

$$\boxed{\mathbf{S}^P_\alpha = [\mathbf{x}_0 \times \mathbf{r}_0\;\; \mathbf{x}_1 \times \mathbf{r}_1 \dots \mathbf{x}_{n-1} \times \mathbf{r}_{n-1}] \qquad \mathbf{S}^P_\theta = [\mathbf{z}_1 \times \mathbf{r}_1\;\; \mathbf{z}_2 \times \mathbf{r}_2 \dots \mathbf{z}_{n-1} \times \mathbf{r}_{n-1}\;\; 0]} \tag{16}$$

In order to get orientation sensitivity vectors we first differentiate **R** with respect to α_j and θ_j:

$$\frac{\partial \mathbf{R}}{\partial \alpha_j} = \frac{\partial}{\partial \alpha_j}\left({}^{o}_{j}\mathbf{R}\ {}^{j}_{j+1}\mathbf{R}\ {}^{j+1}_{n}\mathbf{R}\right) = {}^{o}_{j}\mathbf{R}\,\Lambda(\mathbf{e}_1)\ {}^{j}_{j+1}\mathbf{R}\ {}^{j+1}_{n}\mathbf{R} = \Lambda({}^{o}_{j}\mathbf{R}\,\mathbf{e}_1){}^{o}_{n}\mathbf{R} = \Lambda(\mathbf{x}_j)\,\mathbf{R}. \tag{17}$$

Substituting (17) into (6) gives us:

$$\Lambda\left(\frac{\partial \varphi}{\partial \alpha_j}\right) = \Lambda(\mathbf{x}_j),$$

which yields:

$$\frac{\partial \varphi}{\partial \alpha_j} = \mathbf{x}_j.$$

In a similar way we obtain:

$$\frac{\partial \varphi}{\partial \theta_j} = \mathbf{z}_j.$$

Since $\mathbf{R}$ does not depend on a_j and d_j it follows that:

$$\frac{\partial \varphi}{\partial a_j} = \frac{\partial \varphi}{\partial d_j} = 0. \tag{18}$$

The result we can summarize as:

$$\boxed{\begin{array}{ll} S^{\varphi}_{a} = [0\ 0\ \ldots\ 0] & S^{\varphi}_{d} = [0\ 0\ \ldots\ 0] \\ S^{\varphi}_{\alpha} = [\mathbf{x}_0\ \mathbf{x}_1\ \ldots\ \mathbf{x}_{n-1}] & S^{\varphi}_{\theta} = [\mathbf{z}_1\ \mathbf{z}_2\ \ldots\ \mathbf{z}_n] \end{array}} \tag{19}$$

By comparing (19) with (9) we notice $S^{\varphi}_{\alpha} = S^{p}_{a}$ and $S^{\varphi}_{\theta} = S^{p}_{d}$. We also notice that sensitivity matrices S^{p}_{θ} and S^{φ}_{θ} constitute the manipulator Jacobian $\mathbf{J}_{\theta}$ in the form which is originally given by Whitney [14].

4. Parallel joint axes

As pointed out by Hayati [6-8], if two consecutive joint axes are nominally parallel, small axis missalignment can cause large variations in link parameters. This invalidates standard calibration algorithms based on Denavit-Hartenberg's four-parameter model for forward kinematics. Therefore he has proposed a fifth parameter, β, which is an additional rotation of the link about its y-axis.

In order to study sensitivities respect to the new parameter, we assume the general case in which all links are described by five parameters. In this case, expressions (A-3) expand to (A-11), which gives:

$$\frac{\partial}{\partial \beta_{i-1}}\,{}^{i-1}_{i}\mathbf{R} = \mathbf{rot}(\mathbf{e}_1,\alpha_{i-1})\ \Lambda(\mathbf{e}_2)\,\mathbf{rot}(\mathbf{e}_2,\beta_{i-1})\ \mathbf{rot}(\mathbf{e}_3,\theta_i) =$$

$$= \Lambda(\mathbf{rot}(\mathbf{e}_1,\alpha_{i-1})\,\mathbf{e}_2)\ {}^{i-1}_{i}\mathbf{R} = \Lambda(\cos(\alpha_{i-1})\,\mathbf{e}_2 + \sin(\alpha_{i-1})\,\mathbf{e}_3)\ {}^{i-1}_{i}\mathbf{R}$$

Consequently:

$$\frac{\partial \mathbf{x}_i}{\partial \beta_j} = \frac{\partial}{\partial \beta_j}\left({}^{o}_{j}\mathbf{R}\ {}^{j}_{j+1}\mathbf{R}\ {}^{j+1}_{i}\mathbf{R}\,\mathbf{e}_1\right) = {}^{o}_{j}\mathbf{R}\,\Lambda(\cos(\alpha_j)\,\mathbf{e}_2 + \sin(\alpha_j)\,\mathbf{e}_3)\,{}^{j}_{i}\mathbf{R}\,\mathbf{e}_1 =$$

$$= \Lambda(\cos(\alpha_j)\,\mathbf{y}_j + \sin(\alpha_j)\,\mathbf{z}_j)\mathbf{x}_i. \tag{20}$$

Comparing (20) with (A-12) we see that the argument of the Λ-operator is $\mathbf{v}_j$. Hence:

$$\frac{\partial \mathbf{x}_i}{\partial \beta_j} = \begin{cases} -\mathbf{x}_i \times \mathbf{v}_j & j<i \\ 0 & j \geq i \end{cases} .$$

Similarly we get:

$$\frac{\partial \mathbf{z}_i}{\partial \beta_j} = \begin{cases} -\mathbf{z}_i \times \mathbf{v}_j & j<i \\ 0 & j \geq i \end{cases} .$$

These partial derivatives we use to obtain the positional sensitivity vector with respect to β_j:

$$\frac{\partial \mathbf{p}}{\partial \beta_j} = \mathbf{v}_j \times \rho_j, \quad \rho_j = \mathbf{r}_j - a_j \mathbf{x}_j = \mathbf{r}_{j+1} + d_{j+1}\mathbf{z}_{j+1}.$$

Using the method shown in the preceding section, we also find the orientation sensitivity vector with respect to β_j:

$$\frac{\partial \varphi}{\partial \beta_j} = \mathbf{v}_j .$$

Since the five-parameter model is used with nominally parallel joints, that is $\alpha_j = 0$, $\mathbf{v}_j$ will be identical to $\mathbf{y}_j$ (see (A-12)). This finally gives:

$$\boxed{S^{p}_{\beta} = [\, \mathbf{y}_0 \times \rho_0 \;\; \mathbf{y}_1 \times \rho_1 \; \dots \; \mathbf{y}_{n-1} \times \rho_{n-1} \,] \qquad S^{\varphi}_{\beta} = [\, \mathbf{y}_0 \;\; \mathbf{y}_1 \; \dots \; \mathbf{y}_{n-1} \,]}$$

5. Second-order sensitivity vectors

The second-order sensitivity vectors we define as partial derivatives of the first order sensitivity vectors obtained in third section. From (8) or (9) it is clear that:

$$\frac{\partial^2 \mathbf{p}}{\partial a_i \partial a_j} = \frac{\partial^2 \mathbf{p}}{\partial a_i \partial d_j} = \frac{\partial^2 \mathbf{p}}{\partial d_i \partial a_j} = \frac{\partial^2 \mathbf{p}}{\partial d_i \partial d_j} = 0.$$

In addition, from (14) and (15) we get:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i \partial a_j} = \begin{cases} \mathbf{x}_i \times \mathbf{x}_j & i<j \\ 0 & i \geq j \end{cases}, \qquad \frac{\partial^2 \mathbf{p}}{\partial \alpha_i \partial d_j} = \begin{cases} \mathbf{x}_i \times \mathbf{z}_j & i<j \\ 0 & i \geq j \end{cases},$$

$$\frac{\partial^2 \mathbf{p}}{\partial \theta_i \partial a_j} = \begin{cases} \mathbf{z}_i \times \mathbf{x}_j & i \leq j \\ 0 & i>j \end{cases}, \qquad \frac{\partial^2 \mathbf{p}}{\partial \theta_i \partial d_j} = \begin{cases} \mathbf{z}_i \times \mathbf{z}_j & i<j \\ 0 & i \geq j \end{cases}.$$

In order to obtain the other second-order sensitivities, we find first the derivatives of $\mathbf{r}_j$. Starting from (A-9), and knowing that $\mathbf{x}_i$ and $\mathbf{z}_i$ are independent from the link distances and link offsets, we can directly write:

$$\frac{\partial \mathbf{r}_j}{\partial a_i} = \begin{cases} 0 & i<j \\ \mathbf{x}_i & i \geq j \end{cases}, \qquad \frac{\partial \mathbf{r}_j}{\partial d_i} = \begin{cases} 0 & i \leq j \\ \mathbf{z}_i & i>j \end{cases}.$$

For the link twists we have:

$$\frac{\partial \mathbf{r}_j}{\partial \alpha_i} = \sum_{k=j+1}^{n} \left(\frac{\partial \mathbf{x}_{k-1}}{\partial \alpha_i} a_{k-1} + \frac{\partial \mathbf{z}_k}{\partial \alpha_i} d_k \right) . \tag{21}$$

Applying (11) and (12), and assuming $i < j$, (21) becomes:

$$\frac{\partial \mathbf{r}_j}{\partial \alpha_i} = \sum_{k=j+1}^{n} [(\mathbf{x}_i \times \mathbf{x}_{k-1})\, a_{k-1} + (\mathbf{x}_i \times \mathbf{z}_k)\, d_k] = \mathbf{x}_i \times \sum_{k=j+1}^{n} (\mathbf{x}_{k-1}\, a_{k-1} + \mathbf{z}_k\, d_k) = -\mathbf{r}_j \times \mathbf{x}_i, \tag{22}$$

For $i = j$ we write (see (A-9)):

$$\frac{\partial \mathbf{r}_j}{\partial \alpha_i} = \frac{\partial}{\partial \alpha_j}\left(\mathbf{x}_j a_j + \mathbf{z}_{j+1}\, d_{j+1} + \mathbf{r}_{j+1} \right) = \frac{\partial \mathbf{x}_j}{\partial \alpha_j} a_j + \frac{\partial \mathbf{z}_{j+1}}{\partial \alpha_j} d_{j+1} + \frac{\partial \mathbf{r}_{j+1}}{\partial \alpha_j}$$

Applying (11), (12) and (22), we get:

$$\frac{\partial \mathbf{r}_j}{\partial \alpha_i} = -(\mathbf{z}_{j+1}\, d_{j+1} + \mathbf{r}_{j+1}) \times \mathbf{x}_j.$$

Since $\mathbf{x}_j \times \mathbf{x}_j = 0$ we have:

$$\frac{\partial \mathbf{r}_j}{\partial \alpha_i} = -(\mathbf{z}_{j+1}\, d_{j+1} + \mathbf{x}_j a_j + \mathbf{r}_{j+1}) \times \mathbf{x}_j = -\mathbf{r}_j \times \mathbf{x}_j, \qquad i = j. \tag{23}$$

Combining (22) and (23) we finally obtain:

$$\frac{\partial \mathbf{r}_j}{\partial \alpha_i} = \begin{cases} -\mathbf{r}_j \times \mathbf{x}_i & i \le j \\ -\mathbf{r}_i \times \mathbf{x}_i & i > j \end{cases} . \tag{24}$$

Similarly we obtain:

$$\frac{\partial \mathbf{r}_j}{\partial \theta_i} = \begin{cases} -\mathbf{r}_j \times \mathbf{z}_i & i \le j \\ -\mathbf{r}_i \times \mathbf{z}_i & i > j \end{cases} .$$

Using now (16) we get:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \alpha_j} = \frac{\partial}{\partial \alpha_i}\left(\frac{\partial \mathbf{p}}{\partial \alpha_j}\right) = \frac{\partial}{\partial \alpha_i}(\mathbf{x}_j \times \mathbf{r}_j) = \frac{\partial \mathbf{x}_j}{\partial \alpha_i} \times \mathbf{r}_j + \mathbf{x}_j \times \frac{\partial \mathbf{r}_j}{\partial \alpha_i} . \tag{25}$$

Assuming $i < j$ and applying (11) and (24), (25) becomes:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \alpha_j} = -(\mathbf{x}_j \times \mathbf{x}_i) \times \mathbf{r}_j - \mathbf{x}_j \times (\mathbf{r}_j \times \mathbf{x}_i). \tag{26}$$

Since $(\mathbf{a} \times \mathbf{b}) \times \mathbf{c} = \mathbf{a} \times (\mathbf{b} \times \mathbf{c}) - \mathbf{b} \times (\mathbf{a} \times \mathbf{c})$ for any three vectors **a**, **b** and **c** (see (A-6)), (26) becomes:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \alpha_j} = \mathbf{x}_i \times (\mathbf{x}_j \times \mathbf{r}_j),$$

For $i \ge j$ (25) yields:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \alpha_j} = \mathbf{x}_j \times (\mathbf{x}_i \times \mathbf{r}_i).$$

Thus:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \alpha_j} = \begin{cases} \mathbf{x}_i \times (\mathbf{x}_j \times \mathbf{r}_j) & i<j \\ -(\mathbf{x}_i \times \mathbf{r}_i) \times \mathbf{x}_j & i \geq j \end{cases} .$$

Similarly we can obtain other second-order derivatives:

$$\frac{\partial^2 \mathbf{p}}{\partial \theta_i\, \partial \alpha_j} = \begin{cases} \mathbf{z}_i \times (\mathbf{x}_j \times \mathbf{r}_j) & i<j \\ -(\mathbf{z}_i \times \mathbf{r}_i) \times \mathbf{x}_j & i \geq j \end{cases} \qquad \frac{\partial^2 \mathbf{p}}{\partial \theta_i\, \partial \theta_j} = \begin{cases} \mathbf{z}_i \times (\mathbf{z}_j \times \mathbf{r}_j) & i<j \\ -(\mathbf{z}_i \times \mathbf{r}_i) \times \mathbf{z}_j & i \geq j \end{cases}$$

Derivatives with reversed order of differentiation can be obtained using Schwarz's theorem for mixed derivatives. For example:

$$\left. \frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \theta_j} \right|_{i<j} = \left. \frac{\partial^2 \mathbf{p}}{\partial \theta_j\, \partial \alpha_i} \right|_{j>i} = -(\mathbf{z}_j \times \mathbf{r}_j) \times \mathbf{x}_i = \mathbf{x}_i \times (\mathbf{z}_j \times \mathbf{r}_j)$$

The results for second-order positional sensitivity vectors are summarized in Table 1. The second-order orientation sensitivity vectors can be obtained by differentiating corresponding first order sensitivity vectors, which are given in (19). The results are summarized in Table 2.

It is interesting to note that second order sensitivity vectors can be expressed in terms of first-order sensitivity vectors. Comparing the results from the tables we can, for example, write:

$$\frac{\partial^2 \mathbf{p}}{\partial \alpha_i\, \partial \theta_j} = \begin{cases} \dfrac{\partial \varphi}{\partial \alpha_i} \times \dfrac{\partial \mathbf{p}}{\partial \theta_j} & i<j \\ -\dfrac{\partial \mathbf{p}}{\partial \alpha_i} \times \dfrac{\partial \varphi}{\partial \theta_j} & i \geq j \end{cases} .$$

This observation can be sumarized as follows:

$$\frac{\partial^2 \mathbf{p}}{\partial c_i\, \partial \xi_j} = \begin{cases} 0 & i<j \\ -\dfrac{\partial \mathbf{p}}{\partial c_i} \times \dfrac{\partial \varphi}{\partial \xi_j} & i \geq j \end{cases} , \quad \frac{\partial^2 \mathbf{p}}{\partial \xi_i\, \partial \eta_j} = \begin{cases} \dfrac{\partial \varphi}{\partial \xi_i} \times \dfrac{\partial \mathbf{p}}{\partial \eta_j} & i<j \\ -\dfrac{\partial \varphi}{\partial \eta_i} \times \dfrac{\partial \mathbf{p}}{\partial \xi_j} & i \geq j \end{cases} , \quad \frac{\partial^2 \varphi}{\partial \xi_i\, \partial \eta_j} = \begin{cases} \dfrac{\partial \varphi}{\partial \xi_i} \times \dfrac{\partial \varphi}{\partial \eta_j} & i<j \\ 0 & i \geq j \end{cases}$$

where c can be symbolically replaced by a or d, while ξ and η can be replaced by α or θ.

Table 1. SECOND-ORDER POSITIONAL SENSITIVITY VECTORS

	a_j	d_j	α_j	θ_j
a_i	0 0	0 0	0 $-\mathbf{x}_i \times \mathbf{x}_j$	0 $-\mathbf{x}_i \times \mathbf{z}_j$
d_i	0 0	0 0	0 $-\mathbf{z}_i \times \mathbf{x}_j$	0 $-\mathbf{z}_i \times \mathbf{z}_j$
α_i	$\mathbf{x}_i \times \mathbf{x}_j$ 0	$\mathbf{x}_i \times \mathbf{z}_j$ 0	$\mathbf{x}_i \times (\mathbf{x}_j \times \mathbf{r}_j)$ $-(\mathbf{x}_i \times \mathbf{r}_i) \times \mathbf{x}_j$	$\mathbf{x}_i \times (\mathbf{z}_j \times \mathbf{r}_j)$ $-(\mathbf{x}_i \times \mathbf{r}_i) \times \mathbf{z}_j$
θ_i	$\mathbf{z}_i \times \mathbf{x}_j$ 0	$\mathbf{z}_i \times \mathbf{z}_j$ 0	$\mathbf{z}_i \times (\mathbf{x}_j \times \mathbf{r}_j)$ $-(\mathbf{z}_i \times \mathbf{r}_i) \times \mathbf{x}_j$	$\mathbf{z}_i \times (\mathbf{z}_j \times \mathbf{r}_j)$ $-(\mathbf{z}_i \times \mathbf{r}_i) \times \mathbf{z}_j$

(Upper value is for $i < j$, lower value is for $i \geq j$)

Table 2. SECOND-ORDER ORIENTATION SENSITIVITY VECTORS

	a_j	d_j	α_j	θ_j
a_i	0 0	0 0	0 0	0 0
d_i	0 0	0 0	0 0	0 0
α_i	0 0	0 0	$\mathbf{x}_i \times \mathbf{x}_j$ 0	$\mathbf{x}_i \times \mathbf{z}_j$ 0
θ_i	0 0	0 0	$\mathbf{z}_i \times \mathbf{x}_j$ 0	$\mathbf{z}_i \times \mathbf{z}_j$ 0

(Upper value is for $i < j$, lower value is for $i \geq j$)

6. Conclusion

Kinematic sensitivity vectors and matrices with respect to link parameters have been defined and derived for open-loop, n DOF manipulators. Sensitivity vectors are expressed in terms of coordinate axes of manipulator links. A recursive algorithm for efficient computation of coordinate axes has been also presented. Second-order sensitivity vectors are also derived. It is shown that the second-order sensitivity vectors can be expressed as vector products of the first-order sensitivity vectors. The results obtained can be used for numeric and symbolic computation of kinematic sensitivities for a particular manipulator type.

Acknowledgement

Author would like to thank Dr. David Carlson from Department of Mathematical Sciences, San Diego State University, for helpful discussions during the final preparation of this paper. Verification of the results and many related computations were done by the help of Robot Shell.

References

[1] Asada, H and K. Hara, "Load Sensitivity Analysis and Adaptive Control of a Direct-Drive Arm", Proc. of the American Control Conference, pp. 799-804, Seattle, WA, 1986.
[2] Craig, J.J., *Introduction to Robotics: Mechanics and Control*, Addison-Wesley Publishing Co., Reading, MA, 1986.
[3] Cruz, J.B. (ed.), *System Sensitivity Analysis*, Benchmark Papers in Electrical Engineering and Computer Science, Dowden Hutchinson and Ross, Inc., Stroudsburg, PA, 1973.
[4] Denavit, J. and R.S. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices:, *J. Appl. Mech.*, pp. 215-221, June 1955.
[5] Frank, P.M., *Introduction to System Sensitivity Theory* , Academic press, New York, NY, 1987.
[6] Hayati, S.A., "Robot Arm Geometric Link Parameter Estimation", Proc. of the 22nd Conferenceon Decision and Control, Vol. 3, pp. 1477-1483, 1983.
[7] Hayati, S.A. and M. Mirmirani, "Improving the Absolute Positioning Accuracy of Robot Manipulators", *Journal of Robotic Systems*, Vol. 2, No. 4, pp. 397-423, 1985.
[8] Hayati, S.A., S. Tso Kam and G. Roston, "Robot Geometri Calibration", IEEE International Conference on Robotics and Automation, Vol. 2, pp. 947-951, Philadelphia, PA, 1988.
[9] Kokotovic, P.V. and R.S. Rutman, "Sensitivity of Automatic Control Systems " (Survey), *Autom. Remote Control* (USSR), Vol. 26, pp. 727-749, 1965.
[10] Medvedev, V.S., "Mathematical Models of Robot Dynamics" , pp. 23-61, in: E.P. Popov (ed.), *Modern Robot Engineering* , MIR Publishers, Moscow, 1982.
[11] Paul, R.P., *Robot Manipulators: Mathematics, Programming, and Control*, The MIT Press, Cambridge, MA, 1981.
[12] Roth, Z.S., B.W. Mooring and B. Ravani, "An Overview of Robot Calibration", *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 5, pp. 377-385, October 1987.
[13] Togai, M., "Manipulability and Sensitivity for Design and Evaluation of Industrial Robots: Kinematic Consideration", 15th ISIR , 1987.
[14] Whitney, D.E., "The mathematics of Coordinated Control of Prosthetic Arms and Manipulators", *Trans ASME , J. Dynamic Systems, Measurement and Control*, Vol. 122, pp. 303-309, 1972.
[15] Yoshikawa, T., "Manipulability of Robotic Mechanisms", Proc. of the 2nd Int. Symp. of Robotics Research, pp. 91-98, Kyoto, August 1984.

Appendix

The orientation **R** and position **p** of an n-DOF manipulator are given by:

$$\mathbf{R} = {}^{o}_{n}\mathbf{R} = \prod_{i=1}^{n} {}^{i-1}_{i}\mathbf{R}\ , \qquad \mathbf{p} = {}^{o}\mathbf{p}_{n} = \sum_{i=1}^{n} {}^{o}_{i-1}\mathbf{R}\ {}^{i-1}\mathbf{p}_{i}, \tag{A-1}$$

where ${}^{i-1}_{i}\mathbf{R}$ and ${}^{i-1}\mathbf{p}_i$ are the relative orientation matrix and position vector of the *i*-th link, and

$${}^{o}_{i}\mathbf{R} = {}^{o}_{i-1}\mathbf{R}\ {}^{i-1}_{i}\mathbf{R}\ . \tag{A-2}$$

If we suppose the four-parameter model originally proposed by Denavit and Hartenberg [4] and modified by Craig [2], then:

$${}^{i-1}_{i}\mathbf{R} = \mathbf{rot}(\mathbf{e}_1,\alpha_{i-1})\,\mathbf{rot}(\mathbf{e}_3,\theta_i), \qquad {}^{i-1}\mathbf{p}_i = \mathbf{e}_1\, a_{i-1} + \mathbf{rot}(\mathbf{e}_1,\alpha_{i-1})\,\mathbf{e}_3\, d_i\,, \tag{A-3}$$

where $\mathbf{e}_1 = [1\ 0\ 0]^T$, $\mathbf{e}_2 = [0\ 1\ 0]^T$ and $\mathbf{e}_3 = [0\ 0\ 1]^T$.

The rotation operator used in (A-3) can be expressed in general form as rotation by angle ϕ about unit vector $\mathbf{k} = [\mathbf{k}_1\ \mathbf{k}_2\ \mathbf{k}_3\,]^T$, $|\mathbf{k}| = 1$:

$$\mathbf{rot}(\mathbf{k},\phi) = I + \sin\phi\, \Lambda(\mathbf{k}) + (1-\cos\phi)\, \Lambda(\mathbf{k})^2, \qquad \text{(A-4)}$$

where $\mathbf{I}$ is the 3×3 identity matrix and $\Lambda(\mathbf{k})$ is the skew-symmetric operator:

$$\Lambda(\mathbf{k}) = \begin{bmatrix} 0 & -\mathbf{k}_3 & \mathbf{k}_2 \\ \mathbf{k}_3 & 0 & -\mathbf{k}_1 \\ -\mathbf{k}_2 & \mathbf{k}_1 & 0 \end{bmatrix}. \qquad \text{(A-5)}$$

Note that operator Λ has the following interesting properties[10]:

$$\begin{array}{lll} \Lambda(\mathbf{a})^T = -\Lambda(\mathbf{a}) & \Lambda(\mathbf{a}+\mathbf{b}) = \Lambda(\mathbf{a}) + \Lambda(\mathbf{b}) & \Lambda(\mathbf{k})^2 = \mathbf{k}\mathbf{k}^T - \mathbf{I} \\ \Lambda(\mathbf{a})\mathbf{b} = \mathbf{a}\times\mathbf{b} & \Lambda(\mathbf{a})\,\Lambda(\mathbf{b}) - \Lambda(\mathbf{b})\,\Lambda(\mathbf{a}) = \Lambda(\Lambda(\mathbf{a})\mathbf{b}) & \Lambda(\mathbf{k})^3 = -\,\Lambda(\mathbf{k}), \\ \Lambda(\mathbf{ca}) = \mathbf{c}\Lambda(\mathbf{a}) & \mathbf{B}\Lambda(\mathbf{a}) = \Lambda(\mathbf{Ba})\mathbf{B} & \end{array} \qquad \text{(A-6)}$$

where $\mathbf{a}$ and $\mathbf{b}$ are arbitrary vectors, $\mathbf{k}$ is a unit vector, c is scalar and $\mathbf{B}$ is an orthogonal 3×3 matrix.
Using these properties, it can be shown:

$$\frac{\partial}{\partial\phi}\mathbf{rot}(\mathbf{k},\phi) = \mathbf{rot}(\mathbf{k},\phi)\,\Lambda(\mathbf{k}) = \Lambda(\mathbf{k})\,\mathbf{rot}(\mathbf{k},\phi)\,. \qquad \text{(A-7)}$$

If we denote the coordinate axes of the i-th link by: $\mathbf{x}_i = {}^{o}\mathbf{x}_i$, $\mathbf{y}_i = {}^{o}\mathbf{y}_i$ and $\mathbf{z}_i = {}^{o}\mathbf{z}_i$, i.e ${}^{o}_{i}\mathbf{R} = [\mathbf{x}_i\ \ \mathbf{y}_i\ \mathbf{z}_i\,]$,

$\mathbf{x}_i = {}^{o}_{i}\mathbf{R}\ \mathbf{e}_1$, $\mathbf{y}_i = {}^{o}_{i}\mathbf{R}\ \mathbf{e}_2$, $\mathbf{z}_i = {}^{o}_{i}\mathbf{R}\,\mathbf{e}_3$, then we can express $\mathbf{p}$ in terms of these vectors:

$$\mathbf{p} = \sum_{i=1}^{n} \mathbf{x}_{i-1}\, \mathbf{a}_{i-1} + \mathbf{z}_i\, \mathbf{d}_i\ . \qquad \text{(A-8)}$$

Similarly we can express the distance of the i-th link from the last, n-th, link:

$$\mathbf{r}_i = \mathbf{r}_{i+1} + \mathbf{x}_i\,\mathbf{a}_i + \mathbf{z}_{i+1}\,\mathbf{d}_{i+1} = \sum_{j=i+1}^{n} \mathbf{x}_{j-1}\,\mathbf{a}_{j-1} + \mathbf{z}_j\,\mathbf{d}_j, \quad i = n\text{-}1, n\text{-}2,\ldots,1, \qquad \mathbf{r}_n = 0, \qquad \mathbf{p} = \mathbf{r}_1. \qquad \text{(A-9)}$$

Note that we have assumed here coordinate assignments as proposed by Craig [2] , where the z-axis of the i-th frame, $\mathbf{z}_i$, is colinear with the i-th joint axis, and the origin of the i-th link is lying on the axis.

Vectors $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_i$ can be computed recursively. Using (A-2), (A-3) and (A-4) we can obtain:

$$\begin{array}{ll} \mathbf{v}_i = \cos(\alpha_{i-1})\,\mathbf{y}_{i-1} + \sin(\alpha_{i-1})\,\mathbf{z}_{i-1}, & \mathbf{y}_i = c_i\mathbf{v}_i - s_i\mathbf{x}_{i-1}\,, \\ \mathbf{x}_i = s_i\mathbf{v}_i + c_i\mathbf{x}_{i-1}\,, & \mathbf{z}_i = \cos(\alpha_{i-1})\,\mathbf{z}_{i-1} - \sin(\alpha_{i-1})\,\mathbf{y}_{i-1}\,, \qquad i = 1,\ldots,n \end{array} \qquad \text{(A-10)}$$

In the five-parameter model proposed by Hayati [6-8], relative orientations and positions of links become:

$${}^{i-1}_{\ \ i}\mathbf{R} = \mathbf{rot}(\mathbf{e}_1,\alpha_{i-1})\,\mathbf{rot}(\mathbf{e}_2,\beta_{i-1})\,\mathbf{rot}(\mathbf{e}_3,\theta_i), \qquad {}^{i-1}\mathbf{p}_i = \mathbf{e}_1\,\mathbf{a}_{i-1} + \mathbf{rot}(\mathbf{e}_1,\alpha_{i-1})\,\mathbf{rot}(\mathbf{e}_2,\beta_{i-1})\,\mathbf{e}_3\,\mathbf{d}_i. \qquad \text{(A-11)}$$

This will result in a similar set of recursive relations for coordinate axes:

$$\begin{array}{ll} \mathbf{v}_i = \sin(\alpha_{i-1})\,\mathbf{z}_{i-1} + \cos(\alpha_{i-1})\,\mathbf{y}_{i-1}\,, & \mathbf{x}_i = s_i\mathbf{v}_i + c_i\mathbf{u}_i, \\ \mathbf{w}_i = \cos(\alpha_{i-1})\,\mathbf{z}_{i-1} - \sin(\alpha_{i-1})\,\mathbf{y}_{i-1}\,, & \mathbf{y}_i = c_i\mathbf{v}_i - s_i\mathbf{u}_i, \\ \mathbf{u}_i = \cos(\beta_{i-1})\,\mathbf{x}_{i-1} - \sin(\beta_{i-1})\,\mathbf{w}_i, & \mathbf{z}_i = \sin(\beta_{i-1})\,\mathbf{x}_{i-1} + \cos(\beta_{i-1})\,\mathbf{w}_i, \quad i = 1,\ldots,n. \end{array} \qquad \text{(A-12)}$$

Note, for $\beta_{i-1} = 0$, $\mathbf{u}_i = \mathbf{x}_{i-1}$, and (A-12) reduces to (A-10).

N90-29864

Efficient Conjugate Gradient Algorithms for Computation of the Manipulator Forward Dynamics

Amir Fijany and Robert E. Scheid

Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California

Abstract

In this paper, we investigate the applicability of conjugate gradient algorithms for computation of the manipulator forward dynamics. The redundancies in the previously proposed conjugate gradient algorithm is analyzed [7]. A new version is developed which, by avoiding these redundancies, achieves a significantly greater efficiency. A preconditioned conjugate gradient algorithm is also presented. A diagonal matrix whose elements are the diagonal elements of the inertia matrix is proposed as the preconditioner. In order to increase the computational efficiency, an algorithm is developed which exploits the synergism between the computation of the diagonal elements of the inertia matrix and that required by the conjugate gradient algorithm.

I. INTRODUCTION

The manipulator forward dynamics problem, which concerns the determination of the motion resulting from the application of a set of joint forces/torques, is essential for the dynamic simulation of robot manipulators. The motivation for devising fast algorithms for the forward dynamics solution stems from applications which require extensive off-line simulation as well as applications which require real-time dynamic simulation. In particular, for many anticipated space teleoperation applications, a faster-than-real-time simulation capability will be essential. In fact, in the presence of the unavoidable delay in information transfer, such a capability would allow a human operator to preview a number of scenarios before run-time [1].

The forward dynamics problem can be stated as follows: given the vector of the actual joint positions (Q) and velocities (Q), and the vector of applied joint forces/torques (τ), find the vector of the joint accelerations (Q). Integrating Q leads to the new values for Q and Q. The process is then repeated for the next τ. The first step in the computation of the forward dynamics is to derive a linear relation (for the given Q) between the vector of joint accelerations and the vector of joint inertia forces/torques. Given the dynamic equations of motion as

$$A(Q)\ddot{Q} + C(Q,\dot{Q}) + G(Q) + J^{t}(Q)F_{E} = \tau \qquad (1)$$

and the bias vector (b) as

$$b = C(Q,\dot{Q}) + G(Q) + J^{t}(Q)F_{E} \qquad (2)$$

the linear relation is derived as

$$A(Q)\ddot{Q} = \tau - b = \Gamma \qquad (3)$$

where A(Q) is an nxn symmetric, positive definite, inertia matrix and J is the 6xn Jacobian matrix (t denote matrix transpose). Q, $\dot{Q}$, $\ddot{Q}$, τ, b, $\Gamma \in \mathbb{R}^n$, and F_E is the 6x1 vector which is a compact representation of the external force (f_E) and moment (n_E) exerted on the End-Effector (EE). The bias vector represents the contribution due to the nonlinear terms as well as the external force and moment. Hence, Γ stands for the vector of applied inertia forces/torques. The bias vector can be obtained by computing the inverse dynamics, using the Newton-Euler (N-E) formulation [2], for the actual value of Q, $\dot{Q}$, and F_E while setting $\ddot{Q}$ to zero. The evaluation of b and Γ, i.e., the derivation of Eq. (3), is necessarily the first step in the computation of forward dynamics.

The proposed algorithms for computation of the forward dynamics differ in their approaches to solving Eq. (3), which directly affect their asymptotic computational complexity. These algorithms can be classified as $O(n)$ algorithms [3]-[6], the $O(n^2)$ algorithms [7], and the $O(n^3)$ algorithms [7]. However, any analysis of the efficiency of these algorithms should be based on the realistic size of the problem, i.e. the number of Degrees-Of-Freedom (DOF). In fact, the comparative study in [3] shows that the $O(n^3)$ composite rigid-body algorithm is the most efficient for n less than 12. It also shows that, due to the large coefficient of n^2 terms on the polynomial complexity, the conjugate gradient algorithm of [7] does not become more efficient than the composite rigid-body algorithm except for very large n, making the algorithm almost impractical.

In this paper, we develop two conjugate gradient algorithms which are significantly more efficient than that of [7]. The better efficiency of these algorithms is mainly achieved by a significant reduction of the coefficient of n^2 terms on the polynomial complexity. The first is a Classical Conjugate Gradient (CCG) algorithm which improves the computation cost of each iteration by eliminating the redundancy in the *extrinsic* equations, i.e., by a better choice of coordinate frame for projection of the *intrinsic* equations. With this reduction in the cost of each iteration, a further efficiency can be achieved by reducing the number of iterations through the use of a preconditioner. The second is a Preconditioned Conjugate Gradient (PCG) algorithm which uses a positive definite diagonal matrix, whose elements are the diagonal elements of the inertia matrix, as a preconditioner. An efficient algorithm for computation of the diagonal elements of the inertia matrix is also developed.

However, despite these improvements, the developed algorithms are, in general, still less efficient than the best $O(n^3)$ algorithm. It should be pointed out that the efficiency of this algorithm is further increased by a recently developed algorithm [8]-[9] which achieves greater efficiency in computing the inertia matrix over the composite rigid-body algorithm in [7]. Despite the improvement in the efficiency of the serial algorithms, even the fastest serial algorithm is far from providing the required efficiency for real-time or faster-than-real-time simulation. This observation clearly suggests that the exploitation of parallelism in the computation is the key factor in achieving the desired efficiency.

The analysis of the parallel efficiency of different algorithms is more complex than that of the serial efficiency [9]. Our investigation indicates that the PCG algorithm presents excellent features for parallel computation [10]. In fact, the parallel version of the PCG algorithm, while requiring a simple architecture, may potentially become the most efficient alternative for parallel computation of the forward dynamics. In fact, such a potential has motivated us to further investigate the PCG algorithm and the impact of the preconditioning on its convergence. In this paper the preliminary results of our investigation are presented.

This paper is organized as follows. In Section II, the CCG and PCG algorithms are briefly reviewed and the particular features of these algorithms in the context of the forward dynamics computation are discussed. In Section III, the CCG algorithm is developed. In Section IV, the PCG algorithm and the algorithm for computation of the diagonal elements of the inertia matrix are presented. Finally, some discussion and concluding remarks are made in Section V.

II. CONJUGATE GRADIENT METHOD AND RESULTING ALGORITHMS

The conjugate gradient method is one of the most widely used methods for the iterative solution of linear systems of equations such as

$$Ax = b \qquad x,\ b \in \mathbb{R}^n \tag{4}$$

where $A \in \mathbb{R}^{n\times n}$ is a symmetric positive-definite matrix. An attractive feature of the method is the guarantee of the convergence in at most n steps. Several developments have contributed to the wide application of the method [13]; they include analysis and experimentation leading to the identification of the most stable versions of the method, an understanding of its error propagation, and the fact that the solution of Eq. (4) arises in many applications.

The discussion given here is mainly based on the treatment found in [12] where the basic algorithm is given as follows:

```
$x_0 = 0$
$r_0 = 0$
    For $j = 1, 2, \dots, n$
        if $r_{j-1} = 0$ then set $x = x_{j-1}$ and quit
            else
                $\beta_j = r^t_{j-1} r_{j-1} / r^t_{j-2} r_{j-2}$        $\beta_1 \equiv 0$        (5)
                $p_j = r_{j-1} + \beta_j p_{j-1}$                        $p_1 \equiv 0$        (6)
                $\alpha_j = r^t_{j-1} r_{j-1} / p^t_j A p_j$                                  (7)
                $x_j = x_{j-1} + \alpha_j p_j$                                               (8)
                $r_j = r_{j-1} - \alpha_j A p_j$                                             (9)
 $x = x_n$
```

This is the Classical Conjugate Gradient (CCG) algorithm which has been analyzed in considerable detail under general conditions.

The interest in the conjugate gradient method has been further increased by the development of the preconditioning strategies to accelerate convergence of the algorithm. Furthermore, while the CCG algorithm and its preconditioned versions are not naturally suitable for parallel computation, they are well matched for vector supercomputers, i.e., they can be efficiently vectorized [13]-[14]. The key concept in achieving a faster convergence resides in improving the condition of matrix A by preconditioning [12]. Let C be some nonsingular symmetric matrix and define $\tilde{A} = C^{-1}AC^{-1}$, $\tilde{b} = C^{-1}b$, and $\tilde{x} = C^{-1}x$. Then the algorithm can be applied to the equivalent transformed system $\tilde{A}\,\tilde{x} = \tilde{b}$ where for an appropriate choice of C the convergence may be accelerated considerably. Let $M = C^2$. The algorithm (for n steps) is written as [12]:

```
$x_0 = 0$
$r_0 = 0$
  For $j = 1, 2, \ldots, n$
    if $r_{j-1} = 0$ then set $x = x_{j-1}$ and quit
      else
        Solve $MZ_{j-1} = r_{j-1}$ for $Z_{j-1}$                          (10)
        $\beta_j = Z^t_{j-1}r_{j-1}/Z^t_{j-2}r_{j-2}$        $\beta_1 \equiv 0$      (11)
        $p_j = Z_{j-1} + \beta_j p_{j-1}$                  $p_1 \equiv 0$      (12)
        $\alpha_j = Z^t_{j-1}r_{j-1}/p^t_j\,Ap_j$                               (13)
        $x_j = x_{j-1} + \alpha_j p_j$                                          (14)
        $r_j = r_{j-1} - \alpha_j Ap_j$                                         (15)
 $x = x_n$
```

This is the Preconditioned Conjugate Gradient (PCG) algorithm and the symmetric positive definite matrix M is called the preconditioner. In order for M to be effective as a preconditioner, it is essential to be able to easily solve the linear systems in Eq. (10). A well chosen preconditioner can lead to rapid convergence, often after $O(n^{1/2})$ iterations [12]. Note that if $M^{-1} = A^{-1}$, then the iteration converges immediately. So one hopes that when $M^{-1} \approx A^{-1}$ (in some sense) the iteration converges very quickly. In fact, this is what has been shown in [15]. As a result, if the matrix A is diagonally dominant then M = Diag (A) may be an excellent preconditioner since M closely approximates A. Furthermore, with the a diagonal matrix the solution of Eq. (10) is trivial. The choice of M = Diag (A) is known as Diagonal Scaling or PCG-DS. Note that, compared to the cost of each iteration of CCG, such a choice leads to only an additional cost of n divisions per iteration of PCG-DS. Given the faster convergence, this represents an efficient tradeoff which explains the preference for the use of PCG-DS over CCG even where A is not diagonally dominant.

However, the serial and parallel computation of the conjugate gradient algorithms, when applied to the forward dynamics solution, differs from its application to more generic problems. In fact, it is usually assumed that the matrix A is given which is not the case for the forward dynamics problem.

For serial processing, note that, the basic operation in the CCG and PCG algorithms is the matrix-vector multiplication in Eqs. (7) and (13) with the computation complexity of $O(n^2)$. Given n iterations, this leads to $O(n^3)$ computational complexity of the algorithms. For forward dynamics problem, this operation represents the evaluation of the vector of joint inertia forces/torques, i.e., $\Gamma(j)$, for a given vector of joint acceleration (p_j), which can be computed in $O(n)$ steps, using the N-E formulation. This can be done for CCG algorithm without explicit computation of A which has also been exploited in [7]. Note that, the derivation of the dynamic models of the industrial manipulators, in symbolic form, shows that their inertia matrices can be practically considered as diagonal dominant [16]. Therefore, the PCG-DS algorithm can be expected to achieve a rapid convergence in solving the forward dynamics problem. However, the application of PCG-DS algorithm requires the computation of the diagonal elements of A. Hence, the algorithmic efficiency in computing the diagonal elements is a key factor in the successful application of PCG-DS algorithm to the forward dynamics solution.

In the context of the forward dynamics solution, the CCG and PCG-DS also provide suitable features for parallel processing. Exploiting maximum parallelism, the matrix-vector multiplication in Eqs. (7) and (13) can be performed in $O(\log_2 n)$ steps with $O(n^2)$ processors. However, besides using too many processors, exploitation of maximum parallelism requires a complex processor interconnection. For the forward dynamics problem, this operation, as is shown in [18], can be performed in $O(\log_2 n)$ steps with n processor and a rather simple interconnection. This leads to the $O(n\log_2 n)$ parallel CCG algorithm. It is shown that, using the same architecture as in [18], the diagonal elements of the inertia matrix can be computed in $O(\log_2 n)$ steps [11]. This implies that, if PCG-DS algorithm converges in $O(n^{1/2})$ iterations, then its parallel version can achieve a computational time of $O(n^{1/2}\log_2 n)$ with n processor and a simple processor interconnection structure. In fact, the parallel PCG-DS may represent the fastest stable algorithm for computation of the forward dynamics problem [10].

III. THE CCG ALGORITHM

III.1 Notations and Preliminaries

The N-E formulation can be expressed as a function g_1 which, given Q, $\dot{Q}$, $\ddot{Q}$, and F_E, evaluates τ as [4]:

$$\tau = g_1(Q, \dot{Q}, \ddot{Q}, F_E) \tag{16}$$

The matrix-vector operation in Eqs. (7) and (13) is a special application of g_1 which evaluates a set of vectors of inertia forces/torques as:

$$\Gamma(j) = g_1(Q_a, 0, \ddot{Q}_j, 0) = g_2(Q_a, \ddot{Q}_j) \tag{17}$$

where Q_a is the vector of joint positions representing the manipulator's configurations for which Eq. (17) is evaluated for a set of $\ddot{Q}_j$'s.

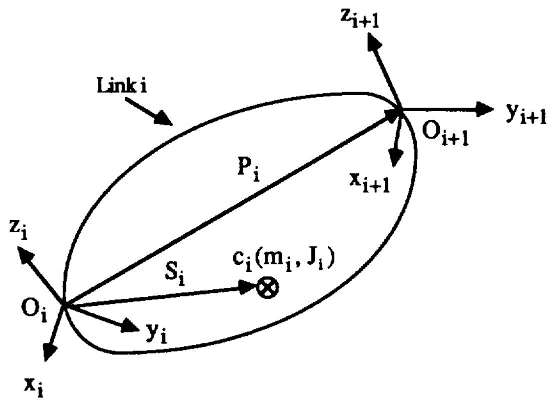


Fig.1. Link, Frames, and Kinematic and Dynamic Parameters

$q_i, \dot{q}_i, \ddot{q}_i$	position, velocity, and acceleration of joint i, respectively.
$\dot{\omega}_i$	Angular acceleration of link i
$\dot{V}_i$	Linear acceleration of link i (point O_i).
$\dot{V}_{ic}$	Linear accelerations of center of mass of link i (point cm_i).
F_i and N_i	Force and moment exerted on center of mass of link i.
f_i and n_i	Force and moment exerted on link i by link i-1.

Table I. Notion Used in the Derivation of the Algorithms.

The major redundancy in the evaluation of Eq. (17) by the algorithm of [7] results from the choice of coordinate frame for projection of the intrinsic equations. Note that the evaluation of the original N-E formulation in link coordinate frames requires $O(n)$ transformations for link-to-link propagation of the variables. Hence, using the link frames for n times evaluation of Eq. (17), as is done in [7], requires $O(n^2)$ transformations. However, if n times evaluation of Eq. (17) is performed in a fixed frame then only $O(n)$ transformations for projection of the vectors and the tensors are required.

In deriving the algorithms, we first develop the intrinsic equations, i.e., the coordinate-free representation of equations. This provides a suitable abstraction since the equations can be derived from the intrinsic physical relationships, which are independent of any coordinate frame. More important, this allows us to distinguish between the redundancy in the intrinsic and that in the extrinsic equations. In order to derive the intrinsic equations, we need to recall some notations. In this paper, according to Gibbs notation, vectors are underlined once and tensors (tensors of order 2) twice. The projection of the vectors and the tensors results in 3x1 (column matrix) and 3x3 scalar matrix wherein the superscript denotes the coordinate frame on which the projection is performed. To any vector $\underline{V}$ a tensor $\hat{\underline{\underline{V}}}$ can be associated whose projection is a 3x3 skew symmetric scalar matrix as:

$$\hat{V} = \begin{bmatrix} 0 & -V_{(z)} & V_{(y)} \\ V_{(z)} & 0 & -V_{(x)} \\ -V_{(y)} & V_{(x)} & 0 \end{bmatrix}$$

Note that $\underline{\underline{\hat{V}}}_1\underline{V}_2 = \underline{V}_1 x \underline{V}_2 = -\underline{V}_2 x \underline{V}_1 = -\underline{\underline{\hat{V}}}_2\underline{V}_1$. Also, a set of notations, presented in Fig. 1 and Table I, are used in the derivation of the algorithms.

III.2 A Variant of The N-E Formulation

Let us write the N-E formulation for link i (Fig. 1) with the nonlinear terms being excluded.

$$\dot{\underline{\omega}}_i = \dot{\underline{\omega}}_{i-1} + \underline{z}_i\ddot{q}_i \tag{18}$$
$$\dot{\underline{V}}_i = \dot{\underline{V}}_{i-1} + \dot{\underline{\omega}}_{i-1} x \underline{P}_{i-1} \tag{19}$$
$$\dot{\underline{V}}_{ic} = \dot{\underline{V}}_i + \dot{\underline{\omega}}_i x \underline{S}_i \tag{20}$$
$$\underline{F}_i = m_i\dot{\underline{V}}_{ic} \tag{21}$$
$$\underline{N}_i = \underline{\underline{J}}_i\dot{\underline{\omega}}_i \tag{22}$$
$$\underline{f}_i = \underline{F}_i + \underline{f}_{i+1} \tag{23}$$
$$\underline{n}_i = \underline{N}_i + \underline{S}_i x \underline{F}_i + \underline{n}_{i+1} + \underline{P}_i x \underline{f}_{i+1} \tag{24}$$
$$\Gamma_i = \underline{z}_i \cdot \underline{n}_i \tag{25}$$

where Γ_i is the ith component of Γ which indicates the inertia force/torque of joint i. Eqs. (18)-(25) describe the procedure for computation of the vector $\Gamma(j)$ or the function g_2. Note that, for the sake of simplicity, an all revolute joints manipulator is considered. However, with small changes, the results can be extended to the manipulator with sliding joint(s).

A variant of g_2 can be derived by replacing Eqs. (20)-(22) into Eqs. (23) and (24) as

$$\underline{f}_i = m_i\dot{\underline{V}}_i + \dot{\underline{\omega}}_i x (m_i\underline{S}_i) + \underline{f}_{i+1} \tag{26}$$
$$\underline{n}_i = \underline{\underline{J}}_i\dot{\omega}_i + \underline{S}_i x [m_i\dot{\underline{V}}_i + \dot{\omega}_i x (m_i\underline{S}_i)] + \underline{n}_{i+1} + \underline{P}_i x \underline{f}_{i+1}$$
$$= (\underline{\underline{J}}_i - m_i\underline{\underline{\hat{S}}}_i\underline{\underline{\hat{S}}}_i)\dot{\underline{\omega}}_i + (m_i\underline{S}_i) x \dot{\underline{V}}_i + \underline{n}_{i+1} + \underline{P}_i x \underline{f}_{i+1} \tag{27}$$

The terms $\underline{\underline{J}}_i - m_i\underline{\underline{\hat{S}}}_i\underline{\underline{\hat{S}}}_i$ and $m_i\underline{S}_i$ represent the first and the second moment of mass of link i with respect to point O_i which are designated as $\underline{\underline{k}}_i$ and $\underline{h}_i$, respectively. Note that $\underline{\underline{k}}_i$ and $\underline{h}_i$ are constant in link i coordinate frame, i.e., coordinate frame i+1, and can be given as the link parameters. The variant of the N-E formulation for computation of the vector Γ, designated as g_3, is written as:

$$\dot{\underline{\omega}}_i = \dot{\underline{\omega}}_{i-1} + \underline{z}_i\ddot{q}_i \tag{28}$$
$$\dot{\underline{V}}_i = \dot{\underline{V}}_{i-1} + \dot{\underline{\omega}}_{i-1} x \underline{P}_{i-1} \tag{29}$$
$$\underline{f}_i = m_i\dot{\underline{V}}_i + \dot{\underline{\omega}}_i x \underline{h}_i + \underline{f}_{i+1} \tag{30}$$
$$\underline{n}_i = \underline{\underline{k}}_i\dot{\underline{\omega}}_i + \underline{h}_i x \dot{\underline{V}}_i + \underline{n}_{i+1} + \underline{P}_i x \underline{f}_{i+1} \tag{31}$$
$$\Gamma_i = \underline{z}_i \cdot \underline{n}_i \tag{32}$$

In the above procedure the explicit computation of the linear acceleration of, and the force and the moment exerted on, the link's center of mass is avoided. Note that the computation cost of g_2 and g_3 is the same. However, if the equations of both procedures are projected on some fixed coordinate frame then evaluation of g_2 requires the transformation of J_i and S_i while that of g_3 requires the transformation of k_i and h_i. For the CCG algorithm, since the evaluation of the original N-E formulation, i.e., g_1, requires the transformation of J_i and S_i, it is more efficient to use g_2. However, g_3 will be used to derive the algorithm for computation of the diagonal elements of the inertia matrix and the evaluation of Γ in PCG-DS algorithm.

III.3 Computation of the CCG Algorithm

As stated before, it is more efficient to project the equations on some fixed frame. To do so, $\underline{P}_i$, $\underline{S}_i$, $\underline{z}_i$, and $\underline{\underline{J}}_i$ should be projected onto the fixed frame. We use the EE (n+1 th) coordinate frame which is slightly more efficient since $\underline{P}_n$, $\underline{S}_n$, $\underline{z}_n$, and $\underline{J}_n$ are constant in this coordinate frame. Let *m* and *a* denote the cost of multiplication and addition, respectively. The computational steps of the CCG algorithm are performed as follows where, for each step, its computational cost is also indicated.

Step 1: Projection of the vectors and the tensors

For i = 1, 2, ..., n

1) Evaluate ${}^{i+1}R_i$

2) ${}^{n+1}R_i = {}^{n+1}R_{i+1}\,{}^{i+1}R_i$ (33)

3) ${}^{n+1}Z_i = {}^{n+1}R_i z_0$ With $z_0 = [0 \quad 0 \quad 1]^t$ (34)

4) ${}^{n+1}S_i = {}^{n+1}R_i\,{}^iS_i$ (35)

5) ${}^{n+1}P_i = {}^{n+1}R_i\,{}^iP_i$ (36)

6) ${}^{n+1}J_i = {}^{n+1}R_i\,{}^iJ_i\,{}^iR_{n+1}$ (37)

The computation cost of this step is obtained as $4nm + (n-1)(96m + 63a)$. In the following the absence of the superscript denotes that the vectors and the tensors are described with respect to the EE coordinate frame.

Step 2: Computation of $r_0 = \tau - \tau_a$

1) Compute $\tau_a = g_1(Q_a, \dot{Q}_a, \ddot{Q}_a, F_e)$

a) For i = 1, 2, ..., n

$\omega_i = \omega_{i-1} + z_i\dot{q}_{ai}$ $\qquad \omega_1 = 0$ (38)

$\dot{\omega}_i = \dot{\omega}_{i-1} + \omega_{i-1} \times z_i\dot{q}_{ai} + Z_i\ddot{q}_{ai}$ $\qquad \dot{\omega}_1 = 0$ (39)

$\dot{V}_i = \dot{V}_{i-1} + \dot{\omega}_{i-1} \times P_{i-1} + \omega_{i-1} \times (\omega_{i-1} \times P_{i-1})$ $\qquad \dot{V}_1 = GZ_1$ (40)

$\dot{V}_{ic} = \dot{V}_i + \dot{\omega}_i \times S_i + \omega_i \times (\omega_i \times S_i)$ (41)

$F_i = m_i\dot{V}_{ic}$ (42)

$N_i = J_i\dot{\omega}_i + \omega_i \times (J_i\omega_i)$ (43)

b) For i = n, n-1, ..., 1

$f_i = F_i + f_{i+1}$ $\qquad f_{n+1} = f_E$ (44)

Algorithm	General		n = 6	
	Mul.	Add.	Mul.	Add.
CCG in [7]	$76n^2+120n-21$	$56n^2+87n-6$	3435	2532
This paper	$47n^2+177n-117$	$46n^2+118n-87$	2637	2277

Table II. Comparison of the CCG algorithms

$$n_i = N_i + S_i \times F_i + n_{i+1} + P_i \times f_{i+1} \qquad n_{n+1} = n_E \tag{45}$$

$$\tau_{ai} = n_i . z_i \tag{46}$$

2) Compute $r_0 = \tau - \tau_a$ (47)

Note that $G = 9.8061 m/s^2$ denotes the acceleration due to the gravity which is along the direction of z_1. The cost of this step is $n(87m+78a)-(21m+24a)$. The rest of the computation is carried out according to Eqs. (5)-(9) where the matrix-vector operation in Eq. (7) is performed by using the function $g_2(Q_a, p_j)$. Each iteration of Eqs. (5)-(9) requires $n(47m+46a)-(10m+23a)$ which, taking n iterations, leads to the total cost of the CCG algorithm as $n^2(47m+46a)+n(177m+118a)-(117m+87a)$. The cost of the developed algorithm is compared to the CCG algorithm of [7]. Note that the algorithm of this paper achieves a better efficiency by a significant reduction in the coefficient of n^2 terms.

IV. THE PCG-DS ALGORITHM

IV.1 An Algorithm for Computation of the Diagonal Elements of Inertia Matrix

From Eq. (3) the diagonal elements of the inertia matrix can be computed as

$$a_{ii} = \Gamma_i \tag{48}$$

for the conditions given by

$$\ddot{q}_i = 1 \text{ and } \ddot{q}_{k\neq i} = 0 \qquad \text{For } k = 1, 2, \ldots, n \tag{49}$$

An algorithm for computation of the terms a_{ii}, using g_3, can be derived as

$$a_{ii} = g_{3i}(Q_{ai}, e_i) \tag{50}$$

where subscript i denotes that g_3 is evaluated for the last n-i+1 links, Q_{ai} is the vector of actual position of the last n-i+1 joints. e_i is an ix1 vector as $e_i = [1\ 0\ \ldots\ 0]^t$. With the conditions given by Eq. (49), let $\dot{\omega}_{j(i)}$ and $\dot{V}_{j(i)}$, and $f_{j(i)}$ and $n_{j(i)}$ (j>i) stand for angular and linear acceleration of, and force and moment exerted on, link j (point O_j) due to the unit acceleration of joint i. For link j, Eqs. (28)-(31) are written as

$$\dot{\underline{\omega}}_{j(i)} = \underline{z}_i \tag{51}$$

$$\dot{\underline{V}}_{j(i)} = \underline{z}_i x \underline{P}_{j,i} \tag{52}$$

$$\underline{f}_{j(i)} = m_j(\underline{z}_i x \underline{P}_{j,i}) + \underline{z}_i x\ \underline{h}_j + \underline{f}_{j+1(i)} \tag{53}$$

$$\underline{n}_{j(i)} = \underline{\underline{k}}_j \underline{z}_i + \underline{h}_j x(\underline{z}_i x \underline{P}_{j,i}) + \underline{n}_{j+1(i)} + \underline{P}_j x \underline{f}_{j+1(i)} \tag{54}$$

and, for link i, these equation are written as

$$\dot{\underline{\omega}}_{i(i)} = \underline{z}_i \tag{55}$$

$$\dot{\underline{V}}_{i(i)} = 0 \tag{56}$$

$$\underline{f}_{i(i)} = \underline{z}_i x\ \underline{h}_i + \underline{f}_{i+1(i)} \tag{57}$$

$$\underline{n}_{i(i)} = \underline{\underline{k}}_i \underline{z}_i + \underline{n}_{i+1(i)} + \underline{P}_i x\ \underline{f}_{i+1(i)} \tag{58}$$

$$a_{ii} = \Gamma_i = \underline{z}_i \cdot \underline{n}_{i(i)} \tag{59}$$

Using Eqs. (51)-(54), Eqs. (57)-(58) can be rewritten as

$$\begin{aligned}\underline{f}_{i(i)} &= \underline{z}_i x \underline{h}_i + \sum_{k=i+1}^{n}\Big[m_k(\underline{z}_i x \underline{P}_{k,i}) + \underline{z}_i x \underline{h}_k\Big] = \underline{z}_i x\Big[\underline{h}_i + \sum_{k=i+1}^{n}\Big[m_k \underline{P}_{k,i} + \underline{h}_k\Big]\\ &= \underline{z}_i x\Big[\underline{h}_i + \sum_{k=i+1}^{n} m_k \underline{P}_i + \sum_{k=i+1}^{n}\Big[m_{k+1}\underline{P}_{k,i} + \underline{h}_k\Big] = \underline{z}_i x \underline{H}_i\end{aligned} \tag{60}$$

$$\begin{aligned}\underline{n}_{i(i)} &= \underline{\underline{k}}_i \underline{z}_i + \sum_{k=i+1}^{n}\Big[\underline{\underline{k}}_k \underline{z}_i + \underline{h}_k x(\underline{z}_i x \underline{P}_{k,i}) + \underline{P}_{k,i} x[m_k(\underline{z}_i x \underline{P}_{k,i}) + \underline{z}_i x \underline{h}_k]\Big]\\ &= \Big\{\underline{\underline{k}}_i + \sum_{k=i+1}^{n}\Big[\underline{\underline{k}}_k - m_k \hat{\underline{\underline{P}}}_{k,i} \hat{\underline{\underline{P}}}_{k,i} - \hat{\underline{\underline{h}}}_k \hat{\underline{\underline{P}}}_{k,i} - \hat{\underline{\underline{P}}}_{k,i} \hat{\underline{\underline{h}}}_k\Big]\Big\}\underline{z}_i = \underline{\underline{K}}_i \underline{z}_i\end{aligned} \tag{61}$$

Note that the conditions given by Eq. (49) imply that link i through link n do not have any relative motion, i.e., are rigidly connected, and form a composite rigid-body. In comparison with Eqs. (57)-(58), $\underline{H}_i$ and $\underline{\underline{K}}_i$ represent the first and the second moment of mass of the composite system composed of link i through link n (denoted as composite system i) about point O_i. From Eqs. (60)-(61), the recursions for computation of $\underline{H}_i$ and $\underline{\underline{K}}_i$ are derived as

$$\underline{H}_i = \underline{h}_i + M_{i+1}\underline{P}_i + \Big\{\underline{h}_{i+1} + \sum_{k=i+2}^{n}\Big[m_k \underline{P}_{k,i+2} + \underline{h}_k\Big] = \underline{h}_i + M_{i+1}\underline{P}_i + \underline{H}_{i+1} \tag{62}$$

where M_{i+1} is the mass (zeroth moment mass) of the composite system i+1, and

$$\underline{\underline{K}}_i = \underline{\underline{k}}_i + \sum_{k=i+1}^{n}\Big[\underline{\underline{k}}_k - m_k \hat{\underline{\underline{P}}}_{k,i} \hat{\underline{\underline{P}}}_{k,i} - \hat{\underline{\underline{h}}}_k \hat{\underline{\underline{P}}}_{k,i} - \hat{\underline{\underline{P}}}_{k,i} \hat{\underline{\underline{h}}}_k\Big] = \underline{\underline{k}}_i +$$

$$\sum_{k=i+1}^{n}\Big[\underline{\underline{k}}_k - m_k(\hat{\underline{\underline{P}}}_{k,i+1} + \hat{\underline{\underline{P}}}_i)(\hat{\underline{\underline{P}}}_{k,i+1} + \hat{\underline{\underline{P}}}_i) - \hat{\underline{\underline{h}}}_k(\hat{\underline{\underline{P}}}_{k,i+1} + \hat{\underline{\underline{P}}}_i) - (\hat{\underline{\underline{P}}}_{k,i+1} + \hat{\underline{\underline{P}}}_i)\hat{\underline{\underline{h}}}_k\Big]$$

which after some manipulations and by using Eq. (62) can be written as

$$\underline{\underline{K}}_i = \underline{\underline{k}}_i + M_{i+1}\hat{\underline{\underline{P}}}_i \hat{\underline{\underline{P}}}_i - \hat{\underline{\underline{H}}}_{i+1}\hat{\underline{\underline{P}}}_i - \hat{\underline{\underline{P}}}_i \hat{\underline{\underline{H}}}_{i+1} + \Big\{\underline{\underline{k}}_{i+1} + \sum_{k=i+2}^{n}\Big[\underline{\underline{k}}_k - m_k \hat{\underline{\underline{P}}}_{k,i+1} \hat{\underline{\underline{P}}}_{k,i+1} -$$

$$\hat{\underline{\underline{h}}}_k \hat{\underline{\underline{P}}}_{k,i+1} - \hat{\underline{\underline{P}}}_{k,i+1} \hat{\underline{\underline{h}}}_k \Big] = \underline{\underline{k}}_i - M_{i+1} \hat{\underline{\underline{P}}}_i \hat{\underline{\underline{P}}}_i - \hat{\underline{\underline{H}}}_{i+1} \hat{\underline{\underline{P}}}_i - \hat{\underline{\underline{P}}}_i \underline{\underline{H}}_{i+1} + \underline{\underline{K}}_{i+1} \tag{63}$$

The diagonal elements of the inertia matrix, or $\mathcal{M}$ = Diag(A), are computed as

For i = n, n-1, ..., 1

$$M_i = M_{i+1} + m_i \qquad M_n = m_n \tag{64}$$

$$\underline{H}_i = \underline{h}_i + M_{i+1}\underline{P}_i + \underline{H}_{i+1} \qquad \underline{H}_n = \underline{h}_n \tag{65}$$

$$\underline{\underline{K}}_i = \underline{\underline{k}}_i + \underline{\underline{k}}_{i+1} - M_{i+1} \hat{\underline{\underline{P}}}_i \hat{\underline{\underline{P}}}_i - \hat{\underline{\underline{H}}}_{i+1} \hat{\underline{\underline{P}}}_i - \hat{\underline{\underline{P}}}_i \underline{\underline{H}}_{i+1} \qquad \underline{\underline{K}}_n = \underline{\underline{k}}_n \tag{66}$$

$$a_{ii} = \underline{\underline{k}}_i \cdot \underline{z}_i \tag{67}$$

It should be pointed out that Renaud [17] used the notion of augmented-body to derive the equations similar to Eqs. (62)-(63). However, our derivation of Eqs. (62)-(63) shows that this notion is implicit in the N-E formulation. The improved efficiency of the above algorithm, compared to the composite rigid-body algorithm in [7], results from the elimination of the redundancy in the intrinsic equations. Note that by directly computing the first and second moment of mass of composite system i about point O_i the redundant computation of the center of mass of composite systems and the force and moments acting on the centers of mass are avoided (see [8] for more discussion regarding these two algorithms).

IV.2 Computation of the PCG-DS Algorithm

Eqs. (64)-(67) describe the intrinsic relation between the diagonal elements of the inertia matrix, which are scalar, and the links kinematic and dynamics parameters, which consist of scalars, vectors, and tensors. In order to compute the diagonal elements, Eqs. (65)-(67) should be projected on some coordinate frame. Due to the evaluation of g_1 and g_2(or g_3) in EE frame, it is more efficient to project Eqs. (65)-(67) on this frame which allows the exploitation of synergism in different computations. To do so h_i and k_i need to be evaluated in EE frame and then H_i, K_i, and a_{ii} can be computed from Eqs. (65)-(67) with the vectors and tensors being described with respect to EE frame. The cost of evaluating $\mathcal{M}$ is then obtained as (n-1)(51*m*+50*a*) where the symmetry of matrices in Eq. (66) is exploited. Note that the additional cost of PCG-DS algorithm, due to the evaluation of $\mathcal{M}$, is almost equal to one iteration of Eqs. (5)-(9) for CCG algorithm. The best algorithm for computation of inertia matrix requires n(69*m*+62*a*)-(57*m*+58*a*) for evaluation of the diagonal elements [8]. Hence, the geater efficiency of the developed algorithm results from the exploitation of synergism in different computation. Having computed $\mathcal{M}$, the second step of the PCG-DS algorithm is performed similar to CCG algorithm and the rest of the computation is carried out according to Eqs. (10)-(15).

V. CONCLUSION

In this paper we investigated the applicability of conjugate gradient algorithms for computation of the manipulator forward dynamics. Two algorithms were presented and their computational efficiency was analyzed. The preconditioned algorithm is particularly promising because of its potentially rapid convergence as well as its suitability for parallel computation. We are currently investigating in greater detail the effect of the preconditioner on the convergence of the algorithm. This work includes analysis of error estimates as well as simulations with actual manipulators.

ACKNOWLEDGEMENT

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under the contract with the National Aeronautics and Space Administration (NASA).

References

1. M.H. Milman and G. Rodriguez,"Cooperative Dual Arm Manipulator Issues and Task Approach," JPL Eng. Memorandum (internal document), Nov. 1987.
2. J.Y.S. Luh, M.W. Walker, and R.P. Paul,"On-line Computation Scheme for Mechanical Manipulator," Trans. ASME J. Dyn. Syst., Meas., and Control, Vol. 102, pp. 69-76, June 1980.
3. R. Featherstone, *Robot Dynamics Algorithms*. Ph.D. Dissertation, Univ. of Edinburgh, 1984.
4. R. Featherstone,"The Calculation of Robot Dynamics Using Articulated-Body Inertia,"Int. J. Robotics Research, Vol.2(2), 1983.
5. G. Rodriguez,"Kalman Filtering, Smooting and Recursive Robot Arm Forward and Inverse Dynamics," IEEE Trans. Robotics&Automation, Vol. RA-5, Dec. 1987. Also in Jet Propulsion Laboratory Publication 86-48, Dec. 1986.
6. G. Rodriguez and K.K. Kreutz,"Recursive Mass Matrix Factorization and Inversion: An Operator Approach to Open and Closed Chain Multibody Dynamics," Jet Propulsion Laboratory Publication 88-11, May 1988.
7. M.W. Walker and D.E. Orin,"Efficient Dynamic Computer Simulation of Robotics Mechanisms," Trans. ASME J. Dyn. Syst., Meas., Control, vol. 104, pp. 205-211, Sept. 1982.
8. A. Fijany and A.K. Bejczy,"An Efficient Method for Computation of the Manipulator Inertia Matrix," Submitted to the J. of Robotic systems.
9. A. Fijany, *Parallel Algorithms and Architectures in Robotics.* Ph.D. Dissertation, Univ. of Paris XI (Orsay, Paris Sud), Sept. 1988.
10. A. Fijany and A.K. Bejczy,"Parallel Algorithms and Architecture for Computation of the Manipulator Forward Dynamics," Submitted to IEEE Trans. Syst., Man, and Cybernetics.
11. A. Fijany and A.K. Bejczy,"Parallel Algorithms for Computation of the Manipulator Inertia Matrix," Submitted to IEEE J Robotics & Automation.
12. G.H. Golub and C.F. Van Loan, *Matrix Computation*. The Johns Hopkins Univ. Press, Baltimore, Maryland, 1983.
13. D.P. O'Leary,"The Block Conjugate Algorithm and Related Methods," Linear Algebra and its Applications, Vol. 29, pp. 293-322, 1980.
14. M. K. Seager,"Parallelizing Conjugate Gradient for the Cray X-MP," Parallel Computing, Vol. 3, pp. 35-47, 1896.
15. P. Concus, G.H.Golub, and D.P. O'Leary,"A Generalized Conjugate Gradient Method for the Numerical Solution of Elliptic Partial Differential Equations," in *Sparse Matrix Computation*, J. R. Bunch, and D.J. Rose (Eds.), Academic Press, New York, 1976.
16. J. Chen,"The Effects of Gear Reduction on Robot Dynamics," Proc. of NASA Conf. on Space Telerobotics, JPL Publication 89-7 (this proceedings).
17. M. Renaud,"An Efficient Iterative Analytical Procedure for Obtaining a Robot Manipulator Dynamic Model," Proc. 1st. Int. Symp. on Robotics Research, 1983.
18. C.S.G. Lee and P.R. Chang,"Efficient Parallel Algorithms for Robot Forward Dynamics Computation," IEEE Trans. Syst., Man, and Cybern., Vol. 18(2), pp. 238-251, March/April 1988.

N90-29835

On the Stability of Robotic Systems with Random Communication Rates*

H. Kobayashi[†], X. Yun, R. P. Paul
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104-6389

Abstract

This paper studies control problems of sampled data systems which are subject to random sample rate variations and delays. Due to the rapid growth of the use of computers more and more systems are controlled digitally. Complex systems such as space telerobotic systems require the integration of a number of sub-systems at different hierarchical levels. While many sub-systems may run on a single processor, some sub-systems require their own processor or processors. The sub-systems are integrated into functioning systems through communications. Comunications between processes sharing a single processor are also subject to random delays due to memory management and interrupt latency. Communications between processors involve random delays due to network access and to data collisions. Furthermore, all control processes involve delays due to causal factors in measuring devices and to signal processing.

Traditionally, sampling rates are chosen to meet the worst case communication delay. Such a strategy is wasteful as the processors are then idle a great proportion of the time; sample rates are not as high as possible resulting in poor performance or in the over specification of control processors; there is the possibility of missing data no matter how low the sample rate is picked.

Randomly sampled systems have been studied since later 1950's, however, results on this subject are very limited and they are not applicable to practical systems. This paper studies asymptotical stability with probability one for randomly sampled multi-dimensional linear systems. A sufficient condition for the stability is obtained. This condition is so simple that it can be applied to practical systems. A design procedure is also shown.

1 Introduction

Many complex systems today involve the integration of a number of different subsystems at various hierarchical levels. Examples of hierarchical subsystems are, for example, in the case of spacecraft:

Level 1 – Assignment of systems to tasks;

Level 2 – Assignment of subsystems to task systems, such as the shuttle manipulator, one of more cameras, an astronaut on EVA;

Level 3 – Control of individual subsystems, cameras comprised of pan tilt, zoom, focus, feature tracking, exception warning; or control of machine tools comprised of spindle, table, tool changer, gauge;

Level 4 – Control of elements, control of manipulator joints, end-effector force measurement, machine tool spindle drive, elevator motor drive, submarine plane control.

These systems all comprise many components which may be ranked hierarchically. Many of the components are now computer controlled and are integrated by means of digital busses or networks. The integration

*This material is based upon work supported by the National Science Foundation under Grant No. ECS-11879. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

†Dept. of Precision Eng., Meiji Univ., Japan. He is currently visiting at Dept. of Computer and Information Science, Univ. of Penn.

of these components into functioning feedback systems, camera – force-sensor – manipulator roll sensor, pitch sensor – main propulsion – planes, implies in addition to data communications, communication rates.

In the case of communication networks, the data rates of point to point communication busses are well known. The rates at which a computer can respond to communication data interrupts requests add a variance to the data rates. In the case of shared networks, such as Ethernet, data collisions add considerable variance to the data rate frequently exceeding the data rate itself. However, these shared networks are very attractive from both the reliability and flexibility standpoints.

Because of their flexibility in programming and speed in computing, digital computers are now regularly employed as integral components of dynamic feedback control systems. They are easily programmed to realize desired compensators. Due to the discrete nature of digital computers, variables in dynamic systems are sampled and quantized before sending to the computers. The well established discrete time system theory (e.g., [8]) provides methods to analyze the behavior of sampled data systems, based on the assumption that the sampling rates are fixed and the same, and the sampling operations on different channels of the systems are synchronized. If the sampling rates are fixed but different on different channels, known as multi-rate sampling, the system analyses are simple if the sampling rates have integral ratios [6, 10].

Due to random delays in measurement devices, signal processing, interrupt latency, priority scheduling, conditional branching, network communications, etc., sampling rates vary randomly in many systems, and the system performance could be expected to be improved if a theory supporting random sampling rates was used. Systems with random sampling processes are called randomly sampled systems. The behavior of a randomly sampled system is, presumably, related to the statistical properties of the random sampling processes as well as system parameters. Randomly sampled systems have been studied by Kalman [11], Leneman [16], Kushner and Tobias [15], Agniel and Jury [2], and others. One of the major motivations for studying randomly sampled systems in late 1950's and early 1960's was the introduction of digital computers in control systems. However as the speed of computers improved dramatically, time delays caused by computers became practically negligible in simple single processor controlled systems compared to other delays, and research on randomly sampled systems came to an end. Nowadays, development of computer controlled systems has reached beyond the stage of single processor control. Many subsystems are integrated into large systems. Furthermore, many complex dynamic systems impose demanding computation requirement. For example, computation time becomes a bottleneck in the implementation of dynamic control algorithms of multi-joint robot manipulators. Delay caused by computation and communication is no longer a negligible factor.

Early researchers in the area of randomly sampled systems primarily considered stability conditions of the systems. Their work is briefly summarized below. Kalman carried out a comprehensive study of sampling systems [11]. He classified sampling into six categories: conventional sampling, nonsynchronous sampling, multiple-order sampling, multi-rate sampling, noninstantaneous sampling, and random sampling. For randomly sampled systems, Kalman showed that if the second moment of the output of an autonomous system is stable, the second moment of the output remains bounded when a bounded input is applied to the system. Based on his state space method [13], Kalman [12] also discussed the regulator problem and stability of a linear system described by independent random functions. This class of systems include randomly sampled systems. Thus the stability conditions obtained for this class of systems are applicable for randomly sampled systems. Kushner and Tobias [15] studied an autonomous linear system with linear and nonlinear feedback. Using a stochastic Lyapunov function, criteria for stability with probability one and s-th moment stability ($s > 0$) were obtained for scalar linear systems, and criteria for stability with probability one and second moment stability were obtained for multi-dimensional linear systems. Agniel and Jury [2] investigated asymptotic stability with probability one of a linear system with a saturating type nonlinear component. A computational procedure was provided to determine the largest stability sector of the nonlinearity for asymptotic stability with probability one. Using a stochastic Lyapunov function, Agniel and Jury in another paper [1] gave a condition for the asymptotic stability with probability one and the second moment asymptotic stability for single-input single-output multi-dimensional linear systems. They also showed that if an autonomous system exhibits asymptotic stability with probability one, the system is almost surely bounded input–bounded output. Leneman [16] studied a single-input single-output first order linear system with feedback. He derived the second moment of the output for the cases with and without input. The input is a stationary stochastic process independent of the sampling process. Consequently, a condition for the second moment stability was given. Assuming the independence of the sampling times and the signals, Dannenberg and Melsa [7] took the expectation of a linear system equation, obtaining a system equation of expectation of the states and outputs. The first moment stability analysis is similar to that of deterministic sampled-data systems. An example of a spacecraft control problem was given, in which it is assumed that there is a probability of missing messages. The problem of random sampling of a random signal was studied by Bergen [4] and Leneman [17]. Their focus was on deriving expressions of the spectral density of a random signal after a random sampling.

This paper studies the stability of randomly sampled systems in relation to the random sampling processes. Though Kalman [11] and Kushner [15] have obtained necessary and sufficient conditions for the stability in the second moment, it is not so easy to apply these conditions to practical systems. This paper studies asymptotic stability with probability one and gives a necessary and sufficient condition for one-dimensional systems and a sufficient condition for multi-dimensional systems. These conditions are easy to verify for given sampling distributions and are thus applicable to practical systems.

In the next section, the asymptotical stability with probability one is defined. A sufficient condition is given for multi-dimensional linear time-invariant randomly sampled systems which is also necessary for one-dimensional systems. A design procedure todetermine feedback gains is obtained in Section 3. If we use a nonlinear compensator such as a computed torque controller for a robotic control system, then we would have a set of simple two-dimensional linear systems. In Section 4, the stability of such two-dimensional systems is considered and the design prcedure is shown for a Bernoulli distribution, a uniform distribution and a mixed uniform distribution.

2 Stability

Consider following linear time-invariant control system.

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

where x is an n-dimensional state vector, u an r-dimensional control vector, and A and B are $n \times n$ and $n \times r$ matrices, respectively. For this system, we apply a constant state feedback input

$$u(t) = Kx(t_k), \tag{2}$$

from $t = t_k$ to $t = t_{k+1} (= t_k + \Delta_k)$, where K is an $r \times n$ matrix. Then $x(t_{k+1})$ is given as follows.

$$x(t_{k+1}) = (\Phi(\Delta_k) + \Psi(\Delta_k)K)x(t_k), \tag{3}$$

where

$$\Phi(\Delta_k) = \exp(A\Delta_k), \text{ and } \Psi(\Delta_k) = \int_0^{\Delta_k} \exp(A\tau)d\tau B.$$

Sampling interval Δ_k is assumed to be subject to some probability distribution function $F(\Delta)$ or distribution density function $f(\Delta)$ and Δ_i and $\Delta_j (i \neq j)$ are statistically independent of each other. For simplicity, we write Eq. (3) as follows

$$x_{k+1} = \Gamma(\Delta_k)x_k. \tag{4}$$

In this paper, we use the following matrix norm which is compatible with usual Euclid norm for vectors:

$$||\Gamma|| = \{\sigma(\Gamma^*\Gamma)\}^{1/2}, \tag{5}$$

where Γ^* is the conjugate transformed matrix and $\sigma(\Gamma)$ denotes the maximum eigenvalues of the matrix Γ. Note, however, that while the stability of the system (1) or (3) is invariant under a similarity transformation of the state variables, the matrix norm depends on the transformation, namely in general

$$||\Gamma|| \neq ||T^{-1}\Gamma T||.$$

The stability of randomly sampled control system Eq. (1) is defined as follows.

Definition 1 (Stability) *The randomly sampled control system Eq. (1) is asymptotically stable with probability one if*

$$Prob[\lim_{k\to\infty} \|x_k\| = 0] = 1$$

for any initial state x_0, where $\|x\|$ is the Euclid norm of vector x.

Now we define the following notation:

$E[\omega]$	: Expectation of random variable ω,
$V[\omega]$	: Variance of random variable ω,

and assume that

$$E[\{\log(\|\Gamma(\Delta)\|)\}^2] < \infty. \tag{6}$$

Then a sufficient condition of the asymptotical stability is given in the next proposition.

Proposition 1 (Sufficient Condition) *Randomly sampled control system (1) is asymptotically stable with probability one if*

$$E = E[\log(\|T^{-1}\Gamma(\Delta)T\|)] < 0, \tag{7}$$

We also have

$$Prob[\|T^{-1}x_k\| < \|T^{-1}x_0\| \exp\{k(E+\epsilon)\}] > 1 - \frac{V}{k\epsilon^2}, \tag{8}$$

for any $\epsilon > 0$, where $V = V[\log(\|T^{-1}\Gamma(\Delta)T\|)]$.

< proof > Assuming $x_0 \neq 0$ without loss of generality, from Eq. (4) we have

$$\log(\|T^{-1}x_k\|/\|T^{-1}x_0\|) \leq \sum_{i=0}^{k-1} \log(\|T^{-1}\Gamma(\Delta_i)T\|).$$

Then the proposition is easily proved by the statistical independence of Δ_i's and Thebyshev's inequality.

< end of proof >

We note that for one-dimensional systems the condition stated in the above proposition is *necessary and sufficient* for the aymsptotic stability with probability one [14]. If the sampling interval is constant, the condition in Prop. 1 is also necessary for the asymptotic stability of multi-dimensional systems.

Now we define

$$\gamma(\Delta) = \log(\|T^{-1}\Gamma(\Delta)T\|), \qquad \text{and} \qquad g(\Delta) = \int_0^\Delta \gamma(\tau)d\tau, \tag{9}$$

then we have the following proposition.

Proposition 2

i. *If the sampling rate Δ is subject to a Bernoulli distribution where $\Delta = \alpha$ with probability p and $\Delta = \beta$ with probability $q = 1 - p$, then the system is asymptotically stable with probability one, if*

$$p\gamma(\alpha) + q\gamma(\beta) < 0.$$

ii. *If the sampling rate Δ is subject to a uniform distribution $U[\alpha, \beta]$, then the system is asymptotically stable with probability one, if*

$$g(\alpha) < g(\beta).$$

iii. *If the sampling rate Δ is subject to $U[\alpha, \beta]$ with probability ε and to $U[\mu, \nu]$ with probability $1 - \varepsilon$, then the system is asymptotically stable with probability one, if*

$$\varepsilon\frac{g(\beta) - g(\alpha)}{\beta - \alpha} + (1 - \varepsilon)\frac{g(\nu) - g(\mu)}{\nu - \mu} < 0.$$

The proof is straightforward, so we omit it here.

3 Design Procedure

Next we discuss a design procedure of a feedback gain K and a matrix T in the following. Now, assume that system

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{10}$$

is controllable, then it is well known that the discretized system

$$x_{k+1} = \Phi(\Delta_k)x_k + \Psi(\Delta_k)u_k \tag{11}$$

is also controllable for almost all sampling interval Δ_k [5]. Then we can assign poles $\{\lambda_i,\ i = 1, 2, \ldots, n\}$ to system (11) if poles $\{\lambda_i\}$ are symmetric with respect to the real axis. Here, we apply Hikita's pole assignment algorithm[9] to the randomly sampled control systems.

[Algorithm]

step (i) *For given $\{\lambda_i\}$, find r-dimensional vectors ξ_i, $i = 1, 2, \ldots, n$, which makes matrix $T(\hat{\Delta}) = [v_1 : \cdots : v_n]$ non-singular. Vector v_i's are given as follows where $\Phi = \Phi(\hat{\Delta})$ and $\Psi = \Psi(\hat{\Delta})$.*

- *if λ_i is a real number, then*

$$v_i = (\Phi - \lambda_i I_n)^{-1}\Psi\xi_i. \tag{12}$$

- *if λ_i and λ_{i+1} are conjugate complex numbers $\alpha_i \pm j\beta_i$, then*

$$v_i = V_{1i}\xi_i - V_{2i}\xi_{i+1},\ \text{and}\ v_{i+1} = V_{1i}\xi_i + V_{2i}\xi_{i+1}, \tag{13}$$

where

$$V_{1i} = \{(\Phi - \alpha_i I_n)^2 + \beta_i^2 I_n\}^{-1}(\Phi - \alpha_i I_n)\Psi,\ \text{and}\ V_{2i} = \{(\Phi - \alpha_i I_n)^2 + \beta_i^2 I_n\}^{-1}\beta_i\Psi. \tag{14}$$

step (ii) *Feedback gain K is given as follows.*

$$K(\hat{\Delta}) = -[\xi_1 : \cdots : \xi_n]T(\hat{\Delta})^{-1}. \tag{15}$$

step (iv) *Check the stability using Proposition 1 or 2. If not stable, return step (i) and try another $\{\lambda_i\}$ and/or $\hat{\Delta}$.*

It is easy to show that for this $T(\hat{\Delta})$ and $K(\hat{\Delta})$, we have

$$\|T^{-1}(\hat{\Delta})\Gamma(\hat{\Delta})T(\hat{\Delta})\| = \max_i\{|\lambda_i|\}. \tag{16}$$

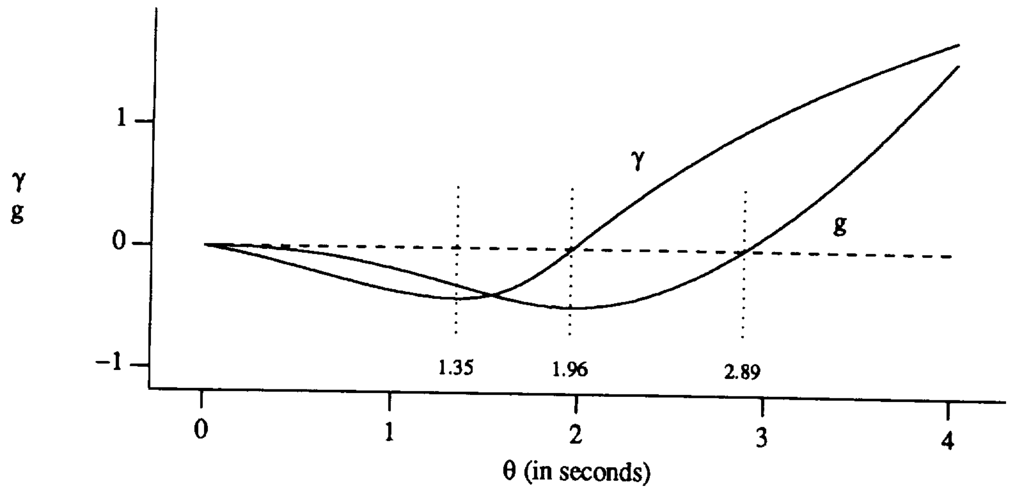
Hence we can use matrices $T(\hat{\Delta})$ and $K(\hat{\Delta})$ to calculate $\gamma(\Delta)$ and $g(\Delta)$. In the next section, we use notations $\gamma(\Delta, \hat{\Delta})$ and $g(\Delta, \hat{\Delta})$ for $\gamma(\Delta)$ and $g(\Delta)$, respectively, to show the dependence of the functions on $\hat{\Delta}$ clearly.

4 Two Dimensional Systems

In this section, we consider control of robot manipulators. We view a robot manipulator as a component of a large system, such as a space station. The robot controller communicates with the other components of the system to achieve cooperative actions. Communication between components is considered to have a longer delay than that within a component. We assume that robot controller has an inner feedback loop which compensates the nonlinearity of manipulator dynamics and operates independently of the other part of the system. The robot dynamic system together with the inner feedback loop becomes a linear system. It is feasible to treat the robot manipulator subsystem as a linear system when integrating and communicating with the other components. For example, if we use the nonlinear feedback controller developed in [3], we have n (=DOF of manipulator) decoupled two-dimensional linear systems

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \tag{17}$$

Figure 1: function $\gamma(\theta, 1)$ and $g(\theta, 1)$



where $x(t) = (e_i(t), \dot{e}_i(t))$ is the error vector for the i-th component of outputs and $u(t)$ is the corresponding input for this component of outputs. If the task is specified in joint space (the joint space control), the i-th component of output is simply the displacement of the i-th joint and the error vector is composed of the joint position error and joint velocity error.

We now study the asymptotical stability of this system under the random sampling rate. The corresponding discrete time system is easily obtained for a sampling interval Δ as follows.

$$x_{k+1} = \begin{bmatrix} 1 & \Delta_k \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \Delta_k^2/2 \\ \Delta_k \end{bmatrix} u_k. \tag{18}$$

We apply the algorithm given above to this system directly. Then we have the following proposition.

Proposition 3 (PD Controller) *Assume that $\{\lambda_i\} = \{\lambda_1, \lambda_2\}$ where $\lambda_1 \neq \lambda_2$, then we have*

$$K(\hat{\Delta}) = ((\lambda_1 + \lambda_2 - \lambda_1\lambda_2 - 1)/\hat{\Delta}^2, (\lambda_1 + \lambda_2 + \lambda_1\lambda_2 - 2)/(2\hat{\Delta})),$$

and

$$\gamma(\Delta, \hat{\Delta}) = \gamma(\theta, 1),$$

where $\theta = \Delta/\hat{\Delta}$.

The proof is obtained by direct calculation. This proposition implies that the function $\gamma(\Delta, \hat{\Delta})$ is the same as the function $\gamma(\theta, 1)$ if we use $K(\hat{\Delta}) = (k_p/\hat{\Delta}^2, k_v/\hat{\Delta})$ instead of $K(1) = (k_p, k_v)$. Therefore we have $g(\Delta, \hat{\Delta}) = \hat{\Delta} g(\theta, 1)$ for the same $K(\hat{\Delta})$. This fact is very useful to design the feedback gain. This will be shown by examples.

Fig. 1 shows $\gamma(\theta, 1)$ and $g(\theta, 1)$ for $\lambda_1 = 0.4$ and $\lambda_2 = 0.7$, where we have

$$K(1) = -(0.18, 0.81), \text{ and } T(1) = \begin{bmatrix} -0.759 & -0.934 \\ 0.651 & 0.333 \end{bmatrix},$$

and ξ_i was used to make the norm of column vectors of T matrix be equal to one.

Example 1 (Bernoulli Distribution) *Let's assume that the sampling interval is subject to Bernoulli distribution, i.e. $\Delta = \alpha$ with probability p and $\Delta = \beta$ with probalility q, where $\alpha < \beta$, $0 \leq p \leq 1$, and $q = 1 - p$. The sufficient stability condition is given as follows.*

$$p\gamma(\alpha/\hat{\Delta}, 1) + q\gamma(\beta/\hat{\Delta}, 1) < 0. \tag{19}$$

Note that if $\hat{\Delta} \geq \beta/1.96 (= \hat{\Delta}^*)$ then the system is asymptotically stable for any α because $\gamma(\theta, 1) < 0$ for any $\theta \leq 1.96$. But we are generally interested in the smallest $\hat{\Delta}$ because it gives us the fastest response.

Fig. 1 shows that the function $\gamma(\theta,1)$ reaches the minimum value -0.417 at $\theta = 1.35$. Let θ^* be the point which satisfies the following equation.

$$\gamma(\theta^*,1) = \frac{p}{q} \times 0.417.$$

Then it is clear that $\hat{\Delta}$ must be greater than $\hat{\Delta}_{min}(= \beta/\theta^*)$ for Eq. (19).

A suitable value of $\hat{\Delta}$ can be found from the range $\hat{\Delta}_{min} < \hat{\Delta} < \hat{\Delta}^*$ by a trial-and-error method using Fig. 1 or Table 1 which gives pairs of $\{\theta_1,\theta_2\}$ such that $\gamma(\theta_1,1) = \gamma(\theta_2,1)$.

(i) *Calculate $a = -(q/p)\gamma(\beta/\hat{\Delta},1)$.*

(ii) *Find $\{\theta_1,\theta_2\}$ such that $\gamma(\theta_1,1) = \gamma(\theta_2,1) \leq a$ using Fig. 1 or Table 1.*

(iii) *Check $\theta_1 < \alpha/\hat{\Delta} < \theta_2$. If so, calculate $K(\hat{\Delta})$. If not so, go back to step (i) with another $\hat{\Delta}$.*

For example, if $\alpha = 10\,msec$, $\beta = 30\,msec$, and $p = 0.75$, then θ^* is about 3.64 and $\hat{\Delta}_{min} = 8.24\,msec$, while $\hat{\Delta}^* = 15.3\,msec$. If we select $\hat{\Delta} = 11\,msec$ then $\frac{q}{p}\gamma(\beta/\hat{\Delta},1) = -0.278$ and $\alpha/\hat{\Delta} = 0.91$. Therefore we can try the 6-th row of Table. 1, and we have $\theta_1 = 0.84 < 0.91 < \theta_2 = 1.68$. Hence the system is asymptotically stable for $K = -(1488, 73.64)$.

Example 2 (Uniform Distribution) *Now assume that Δ is subject to a uniform distribution $\mathcal{U}[\alpha,\beta]$. The sufficient condition of the asymptotical stability with probability one is given as follows:*

$$g(\alpha/\hat{\Delta},1) > g(\beta/\hat{\Delta},1).$$

The function $g(\theta,1)$ has its minimum value at $\theta = 1.96$. Now we define $\hat{\Delta}^* = \beta/19.6$ and $\hat{\Delta}_{min} = \beta/2.89$. If $\hat{\Delta} \geq \hat{\Delta}^*$, then the above sufficient condition is satisfied for any α. Therefore the system is asymptotically stable if $\hat{\Delta} \geq \hat{\Delta}^*$. On the other hand, if $\hat{\Delta} \leq \hat{\Delta}_{min}$, then the above condition is not satisfied for any α.

Table 1 also gives pairs of $\{\theta_3,\theta_4\}$ and the ratio θ_3/θ_4 such that $g(\theta_3,1) = g(\theta_4,1)$. If there is a pair $\{\theta_3,\theta_4\}$ such that $\alpha/\beta > \theta_3/\theta_4$, then the system is asymptotically stable for the $K(\hat{\Delta})$ where $\hat{\Delta} = \alpha/\theta_3$. Therefor we can determine $\hat{\Delta}$ easily using this table as follows:

(i) *Calculate $a = \alpha/\beta$.*

(ii) *Find a pair $\{\theta_3,\theta_4\}$ in the Table 1 such that $a > \theta_3/\theta_4$.*

(ii) *Calculate $\hat{\Delta} = \alpha/\theta_3$ and $K(\hat{\Delta})$.*

Now assume that $\alpha = 10\,msec$ and $\beta = 30\,msec$, then we have $\hat{\Delta}^* = 15.3\,msec$, $\hat{\Delta}_{min} = 10.38\,msec$, and $\alpha/\beta = 1/3 > 0.273$ in the Table 1. Therefore we can use $\alpha/\hat{\Delta} = 0.75$ and $\hat{\Delta} = 13.33msec$. Hence the system is asymptotically stable with $K = -(1065, 62.31)$ if $\beta < 36.7\,msec$. Table 2 shows the IAE (Integration of Absolute value of the Error) for fifty random streams with the initial condition $x(0) = (1.0, 0)^T$. The table shows that when $\beta \geq 40\,msec$, the STD (STanderd Deviation) and the maximum values of IAE for the velocity error $\dot{e}_i(t)$ become very large compared to the cases where $\beta \leq 35\,msec$. This means that the system is still stable but there is a large vibration in the response for $\hat{\Delta} \geq 40\,msec$. It is interesting since $\hat{\Delta}$ selected above assures the asymptotically stability for $\beta < 36.7\,msec$.

Example 3 (Mixed Uniform Distribution) *Next we assume that Δ is subject to a uniform distribution $\mathcal{U}[\alpha,\beta]$ with probability ε and to $\mathcal{U}[\mu,\nu]$ with probability $1-\varepsilon$. The sufficient condition is given as follows:*

$$E = \varepsilon\frac{g(\beta/\hat{\Delta},1) - g(\alpha/\hat{\Delta},1)}{\beta/\hat{\Delta} - \alpha/\hat{\Delta}} + (1-\varepsilon)\frac{g(\nu/\hat{\Delta},1) - g(\mu/\hat{\Delta},1)}{\nu/\hat{\Delta} - \mu/\hat{\Delta}} < 0.$$

Though the selection of $\hat{\Delta}$ becomes a little difficult, we can use the following procedure to estimate an appropriate $\hat{\Delta}$:

(i) *Define $\bar{\alpha} = (\alpha+\beta)/2.0$, $\bar{\beta} = (\mu+\nu)/2.0$, $p = \varepsilon$, and $q = 1-p$.*

(ii) *Determine $\hat{\Delta}$ using the procedure in Exam. 1 for $\alpha = \bar{\alpha}$ and $\beta = \bar{\beta}$.*

(iii) *Check the condition. If satisfied, calculate $K(\hat{\Delta})$. If not, try another value for $\hat{\Delta}$.*

Figure 2: Simulations for Bernoulli Distribution, Uniform Distribution and Mixed Uniform Distribution

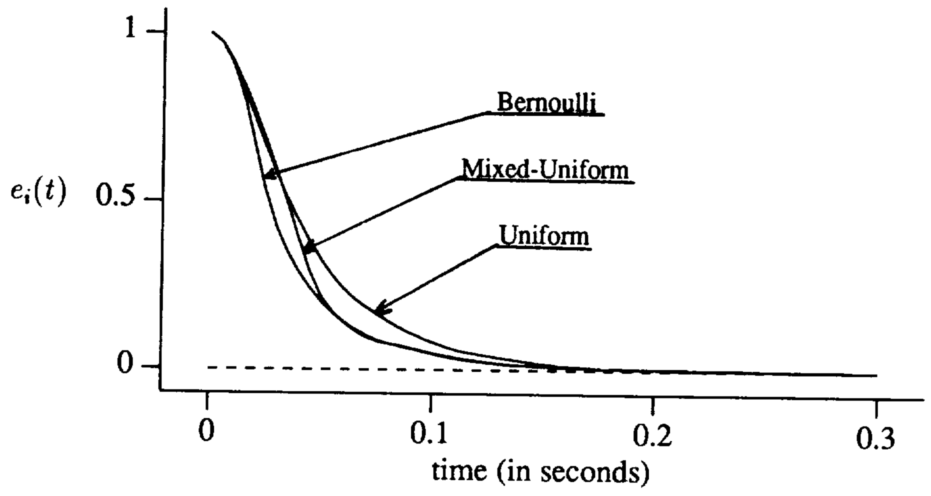


Table 1: θ_1, θ_2 ,θ_3, and θ_4

$\gamma(\theta_1,1)=\gamma(\theta_2,1)$			$g(\theta_3,1)=g(\theta_4,1)$			
$\gamma(\theta,1)$	θ_1	$\theta 2$	$g(\theta,1)$	θ_3	θ_4	θ_3/θ_4
0.00	0.00	1.96	0.000	0.00	2.88	0.000
-0.05	0.18	1.92	-0.009	0.25	2.87	0.087
-0.10	0.33	1.89	-0.039	0.50	2.84	0.176
-0.15	0.46	1.83	-0.094	0.75	2.78	0.270
-0.20	0.58	1.79	-0.173	1.00	2.69	0.372
-0.25	0.71	1.73	-0.270	1.25	2.56	0.488
-0.30	0.84	1.68	-0.373	1.50	2.39	0.628
-0.35	0.98	1.60	-0.456	1.75	2.17	0.806
-0.40	1.18	1.48	-0.467	1.80	2.12	0.849

Now assume that Δ is subject to $\mathcal{U}[5\,msec, 15\,msec]$ with probability $\varepsilon = 0.75$ and to $\mathcal{U}[20\,msec, 40\,msec]$ with probability 0.25. Then we have $\bar{\alpha} = 10msec$, $\bar{\beta} = 30msec$, $p = 0.75$, and $q = 0.25$. If we use $\hat{\Delta} = 11msec$ from the result of Exam. 1, then we have $E = -0.04 < 0$. Therefore the system is asymptotically stable for the same $K = -(1488, 73.64)$.

Fig. 2 shows the simulations of $x(t)$ for three cases discussed above where $x(0) = (1.0, 0)^T$.

It is easily shown that even if we use a PID controller

$$z_{k+1} = z_k + [1:0]x_k, \text{ and } u_k = K_1 z_k + K_2 x_k, \tag{20}$$

or a PD controller with one step delay

$$u_k = K(\Phi(\hat{\Delta})x_{k-1} + \Psi(\hat{\Delta})u_{k-1}), \tag{21}$$

instead of the PD controller given in Prop. 3, we have the similar proposition . Therefore we can determine $\hat{\Delta}$ easily.

5 Conclusions

In this paper, the stability of randomly sampled linear control systems was discussed and the following results were obtained.

1. A sufficient condition for the asymptotical stability in a norm with probability one was obtained for multi-dimensional systems.
2. For a simple two-dimensional system with PD controllers, a design procedure was shown which was easily applicable to systems with PID controllers or PD controllers with one step delay.

The results given in this paper are also easily applicable to the robotic control systems where computed torque controllers or PD controllers with a feedforward term are used at the random sampling rate. The results will be shown in the near future [14].

Table 2: IAE for $\mathcal{U}[10\,msec, \beta\,msec]$

	MEAN		STD		MAX	
β	$e_i(t)$	$\dot{e}_i(t)$	$e_i(t)$	$\dot{e}_i(t)$	$e_i(t)$	$\dot{e}_i(t)$
25	0.0531	0.9999	0.0022	0.0011	0.0572	1.0020
30	0.0511	1.0039	0.0043	0.0200	0.0561	1.1310
35	0.0498	1.0198	0.0060	0.0515	0.0589	1.2370
40	0.0509	1.2418	0.0077	0.4305	0.0717	2.7051
45	0.0709	2.6492	0.0638	4.4592	0.4354	30.276

References

[1] R. G. Agniel and E. I. Jury. Almost sure boundness of randomly sampled systems. *SIAM Journal on Control*, 9(3):372–384, August 1971.

[2] R. G. Agniel and E. I. Jury. Stability of nonlinear randomly sampled systems. In *Proceedings of 7th Annual Allerton Conference on Circuit and System Theory*, pages 710–719, University of Illinois, Urbana, Illinois, October 1969.

[3] A. K. Bejczy, T. J. Tarn, X. Yun, and S. Han. Nonlinear feedback control of puma 560 arm by computer. In *Proceedings of 24th Conference of Decision and Control*, pages 1680–1688, 1985.

[4] A. R. Bergen. On the statistical design of linear random sampling systems. In *Proceedings of 1st Congress of International Federation of Automatic Control*, pages 430–435, 1960.

[5] C-T. Chen. *Linear System Theory and Design*. Holt, Rinehart and Winston, Inc., 1970.

[6] T. C. Coffey and I. J. Williams. Stability analysis of multiloop, multirate sampled systems. *AIAA Journal*, 4:2178–2190, 1966.

[7] K. D. Dannenberg and J. L. Melsa. Stability analysis of randomly sampled digital control sytems. *Automatica*, 11:99–103, 1975.

[8] G. F. Franklin and J. D. Powell. *Digital Control of Dynamic Systems*. Addison-Wesley Publishing Co., Inc., Reading, Mass., 1980.

[9] Hikita, Koyama, and Miura. *Society of Instrument and Control Engineers*, 556–561, 1975. in Japanese.

[10] E. I. Jury. A note on multirate sampled-data systems. *IEEE Transactions on Automatic Control*, AC-12:319–320, 1967.

[11] R. E. Kalman. *Analysis and Synthesis of Linear Dynamical Systems Operating on Ramdomly Sampled Data*. PhD thesis, Columbia University, New York, 1957.

[12] R. E. Kalman. Control of randomly varying linear dynamical systems. In *Proceedings of Symposia in Applied Mathematics*, pages 287–298, 1962. vol. XIII.

[13] R. E. Kalman. A unified approach to the theory of sampling systems. *Journal of Franklin Institute*, 267:405–436, May 1959.

[14] H. Kobayashi, X. Yun, and R. P. Paul. Control of randomly sampled robotic systems. In *Proceedings of IEEE Inter. Conf. on Robotics and Automation*, May 1989. submitted.

[15] H. J. Kushner and L. Tobias. On the stability of randomly sampled systems. *IEEE Transactions on Automatic Control*, AC-14(4):319–324, August 1969.

[16] Oscar A. Z. Leneman. Random sampling of random processes: mean-square behavior of a first order closed- loop system. *IEEE Transactions on Automatic Control*, 429–432, August 1968.

[17] Oscar A. Z. Leneman. Random sampling of random processes: step-wise processes. In *Proceedings of 3rd Congress of International Federation of Automatic Control*, June 1966.

ROBOT TASK PLANNING AND ASSEMBLY

PRECEDENCE RELATIONSHIP REPRESENTATIONS OF MECHANICAL ASSEMBLY SEQUENCES

L. S. Homem de Mello
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

A. C. Sanderson
Electrical, Computer, and Systems Engineering Dept.
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

Abstract

This paper presents two types of precedence relationship representations for mechanical assembly sequences: precedence relationships between the establishment of one connection between two parts and the establishment of another connection, and precedence relationships between the establishment of one connection and states of the assembly process. Precedence relationship representations have the advantage of being very compact. The problem with these representations was how to guarantee their correctness and completeness. Two theorems are presented each of which leads to the generation of one type of precedence relationship representation guaranteeing its correctness and completeness for a class of assemblies.

1. Introduction

The generation of assembly sequences is an important capability for both autonomous and telerobotic systems for space applications. Assembly, repair, servicing, and sample acquisition are examples of tasks that are envisioned for space robotic systems. In each case, a plan or sequence of operations must be generated, usually off-line, based on prior knowledge. In real-time, it may be necessary to modify the plan based on monitoring and sensing of the execution. The desirable representation of the alternative plans for an off-line planning system may be quite different from the desirable representation for the real-time control system. The understanding of alternative representations of such plans is fundamental to their integration into a useful system.

Several methodologies for representing assembly sequences have been utilized. These include representations based on directed graphs [3], on AND/OR graphs [8], on establishment conditions [2], and on precedence relationships [3, 6]. Those based on directed graphs and on AND/OR graphs are explicit representations since there is a mapping from the assembly tasks into the elements of the representations. Those based on establishment conditions and on precedence relationships are implicit representations because they consist of conditions that must be satisfied by the assembly sequences.

In previous work [9] we have described a correct and complete algorithm for the generation of mechanical assembly sequences. This algorithm yields the AND/OR graph representation of assembly sequences. The correspondence between the AND/OR graph and the directed graph representations has also been established [10].

In this paper we address the precedence relationship representations of assembly sequences. These representations have the advantage that they are very compact and therefore might be preferred in real-time planning of assembly sequences. The problem with precedence relationship representations was the assessment of their correctness and completeness. By correctness of the representation we mean that only feasible sequences satisfy the precedence relationships. By completeness we mean that all the feasible sequences satisfy the precedence relationships.

Two types of precedence relationship representations can be used to represent mechanical assembly sequences: precedence relationships between the establishment of one connection between two parts and the establishment of another connection, and precedence relationships between the establishment of one connection and states of the assembly process. This paper describes these two representations and shows two theorems that can be used to guarantee their correctness and completeness for a class of assemblies.

2. Background

A mechanical assembly is a composition of parts interconnected forming a stable unit. Each part is a solid object. Parts are interconnected whenever they have one or more surfaces in contact. Surface contacts between parts reduce the degrees of freedom for relative motion. A cylindrical contact, for example, prevents any relative motion that is not a translation along the axis or a rotation around the axis. Attachments may act on surface contacts and eliminate all degrees of freedom for relative motion. For example, if a cylindrical contact has a pressure-fit attachment, then no relative motion between the parts is possible.

A subassembly is a nonempty subset of parts that either has only one element (i.e. only one part), or is such that every part has at least one surface contact with another part in the subset. Although there are cases in which it is possible to join the same pair of parts in more than one way, a unique assembly geometry will be assumed for each pair of parts. This geometry corresponds to their relative location in the whole assembly. A subassembly is said to be stable if its parts maintain their relative position and do not break contact spontaneously. All one-part subassemblies are stable.

The assembly process consists of a succession of tasks, each of which consists of joining subassemblies to form a larger subassembly. The process starts with all parts separated and ends with all parts properly joined to form the whole assembly. For the current analysis, it is assumed that exactly two subassemblies are joined at each assembly task, and that after parts have been put together, they remain together until the end of the assembly process.

It is also assumed that whenever two parts are joined all contacts between them are established. Due to this assumption, an assembly can be represented by a simple undirected graph $\langle P, C \rangle$ in which $P = \{ p_1, p_2, \cdots, p_N \}$ is the set of nodes, and $C = \{ c_1, c_2, \cdots, c_L \}$ is the set of edges. Each node in P corresponds to a part in the assembly, and there is one edge in C connecting every pair of nodes whose corresponding parts have at least one surface contact. The elements of C are referred to as *connections*, and the graph $\langle P, C \rangle$ is referred to as the *assembly's graph of connections*. A connection encompasses all contacts between two parts. Figure 1 shows an assembly in exploded view, and figure 2 shows its corresponding graph of connections.

• Assembly states

The state of the assembly process is the configuration of the parts at the beginning (or at the end) of an assembly task. The configuration of parts is given by the contacts that have been established. Since whenever two parts are joined all contacts between them are established, the configuration of parts is given by the connections that have been established. Therefore, a state of the assembly process can be represented by an L-dimensional binary vector $\underline{x} = [x_1, x_2, \cdots, x_L]$ in which the i^{th} component x_i is true or false respectively if the i^{th} connection is established in that state or not.

As mentioned above, it is assumed that whenever a subassembly is formed all connections between its parts are established. Therefore, any subassembly can be characterized by its set of parts. In the rest of this paper, references to subsets of parts should be understood as references to the subassemblies made up of those parts. It will always be clear from context what the whole assembly is. Because of this assumption, any state of the assembly process can also be represented by a partition of the set of parts of the whole assembly.

Given an assembly's graph of connections and one of the two representations of assembly states described above (binary vector or partition), it is straight forward to obtain the other representation.

There are partitions of the set of parts of the whole assembly that cannot characterize a state of the assembly process. For example, the partition { {CAP, HANDLE}, {RECEPTACLE}, {STICK} } cannot characterize a state of the assembly process for the assembly shown in figure 1 because the subset {CAP, HANDLE} does not characterize a subassembly. Partitions that can characterize a state of the assembly process will be referred to as *state partitions*, and partitions that cannot characterize a state will be referred to as *nonstate partitions*.

Similarly, not all L-dimensional binary vectors can characterize a state. For example, for the assembly shown in figure 1 the 5-dimensional binary vector [true, true, false, false, false] does not correspond to a state because if connections c_1 and c_2 are established then connection c_3 should also be established. L-dimensional binary vectors that can characterize a state will be referred to as *state vectors* whereas L-dimensional binary vectors that cannot characterize a state will be referred to as *nonstate vectors*.

Any state of the assembly process can be associated to a simple undirected graph $\langle P, C_k \rangle$ in which P is the set of nodes of the assembly's graph of connections, and C_k is the subset of connections ($C_k \subseteq C$) that is established in that state. This graph is referred to as the *state's graph of connections*. Except for the final state of the assembly process, a state's graph of connections has more than one component.

We will use the subassembly predicate *sa* to determine whether a subset of parts makes up a subassembly. The argument to this predicate is a subset of parts, and its value is either true or false depending on whether that subset of parts corresponds to a subassembly. For example, for the assembly shown in figure 1, $sa(\{$ RECEPTACLE, HANDLE $\})=$true, whereas $sa(\{$ CAP, HANDLE $\})=$false. From the assembly's graph of connections it is straight forward to compute *sa* for any given subset of parts.

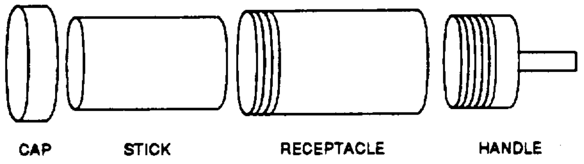


Figure 1: A simple product in exploded view.

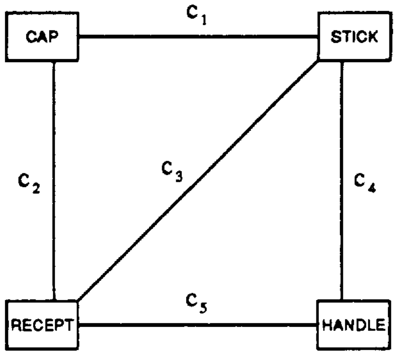


Figure 2: The graph of connections for the product shown in Figure 1

In this paper, a partition of the set of parts whose elements all satisfy the subassembly predicate is an assembly state representation, regardless of whether that state actually occurs in any of the different ways the assembly can be assembled. The corresponding L-dimensional binary vector is also an assembly state representation. And the corresponding configuration of parts is an assembly state. For example, for the assembly shown in figure 1, the partition { {CAP, RECEPTACLE, HANDLE}, {STICK} } as well as the corresponding L-dimensional binary vector [false, true, false, false, true] are assembly state representations. Yet, since it was assumed that once parts are put together they remain together, the configuration of parts (i.e. the state) corresponding to these representations cannot occur in any assembly process. Once the cap and the handle have been joined to the receptacle, it is no longer possible to join the stick.

In this paper, an assembly state representation for which all subassemblies satisfy the stability predicate is said to be a *stable assembly state representation.* For example, for the assembly shown in figure 1, the partition { {CAP, RECEPTACLE, HANDLE}, {STICK} } as well as the corresponding binary vector [false, true, false, false, true] are stable assembly state representations.

- **Assembly tasks**

Given two subassemblies characterized by their sets of parts θ_i and θ_j, we say that joining θ_i and θ_j is an assembly task if the set $\theta_k = \theta_i \cup \theta_j$ characterizes a subassembly. For example, for the assembly shown in figure 1, if $\theta_i =$ { RECEPTACLE } and $\theta_j =$ { HANDLE } then joining θ_i and θ_j is an assembly task, whereas if $\theta_i =$ { CAP } and $\theta_j =$ { HANDLE } then joining θ_i and θ_j is not an assembly task. The subassemblies θ_i and θ_j are the *input* subassemblies of the assembly task, and θ_k is the *output* subassembly of the assembly task. Due to the assumption of unique geometry, an assembly task can be characterized by its input subassemblies only and it can be represented by a set of two subsets of parts. For example, for the assembly shown in figure 1, the joining of the cap to the receptacle is represented by { {CAP}, {RECEPTACLE} }.

An assembly task is said to be *geometrically* feasible if there is a collision-free path to bring the two subassemblies into contact from a situation in which they are far apart. And an assembly task is said to be *mechanically* feasible if it is feasible to establish the attachments that act on the contacts between the two subassemblies.

- **Assembly sequences**

Given an assembly that has N parts, an ordered set of $N-1$ assembly tasks $\tau_1, \tau_2, \cdots, \tau_{N-1}$ is an assembly sequence if there are no two tasks that have a common input subassembly, the output subassembly of the last task is the whole assembly, and the input subassemblies to any task τ_i is either a one-part subassembly or the output subassembly of a task that precedes τ_i. To any assembly sequence $\tau_1, \tau_2, \cdots, \tau_{N-1}$ there corresponds an ordered sequence $s_1, s_2, \cdots, s_N$ of N assembly states of the assembly process. The state s_1 is the state in which all parts are separated. The state s_N is the state in which all parts are joined forming the whole assembly. And any two consecutive states s_i and s_{i+1} are such that only the two input subassemblies of task τ_i are in s_i and not in s_{i+1}, and only the output subassembly of task τ_i is in s_{i+1} and not in s_i. Therefore, an assembly sequence can also be characterized by an ordered sequence of states.

An example of an assembly sequence for the assembly shown in figure 1 is:

1. The first task (τ_1) consists of joining the cap to the receptacle.
2. The second task (τ_2) consists of joining the stick to the subassembly made up of the cap and the receptacle.
3. The third task (τ_3) consists of joining the handle to the subassembly made up of the cap, the stick, and the receptacle.

An assembly sequence is said to be feasible if all its assembly tasks are geometrically and mechanically feasible, and the input subassemblies of all tasks are stable. The assembly sequence described above is feasible. An example of an unfeasible assembly sequence for the assembly shown in figure 1 is:

1. The first task (τ_1) consists of joining the cap to the receptacle.
2. The second task (τ_2) consists of joining the handle to the subassembly made up of the cap and the receptacle.
3. The third task (τ_3) consists of joining the stick to the subassembly made up of the cap, the stick, and the receptacle.

This assembly sequence is unfeasible because the third task (τ_3) is not geometrically feasible since there is no collision free path to bring the stick into the receptacle, once both the cap and the handle have been joined to the receptacle.

An assembly sequence (not necessarily feasible) can be represented in different ways. We will use the following representations:

- An ordered list of task representations. The number of elements in this list is equal to the number of parts minus one.
- An ordered list of binary vectors. Each vector must correspond to a state (not necessarily stable). The number of elements in this list is the equal to the number of parts.
- An ordered list of partitions of the set of parts. Each partition must correspond to a state (not necessarily stable). The number of elements in this list is equal to the number of parts.
- An ordered list of subsets of connections. The number of elements in this list is equal to the number of parts minus one.

Given the assembly's graph of connections and an assembly sequence in any of these four representations, it is straight forward to obtain the other three representations.

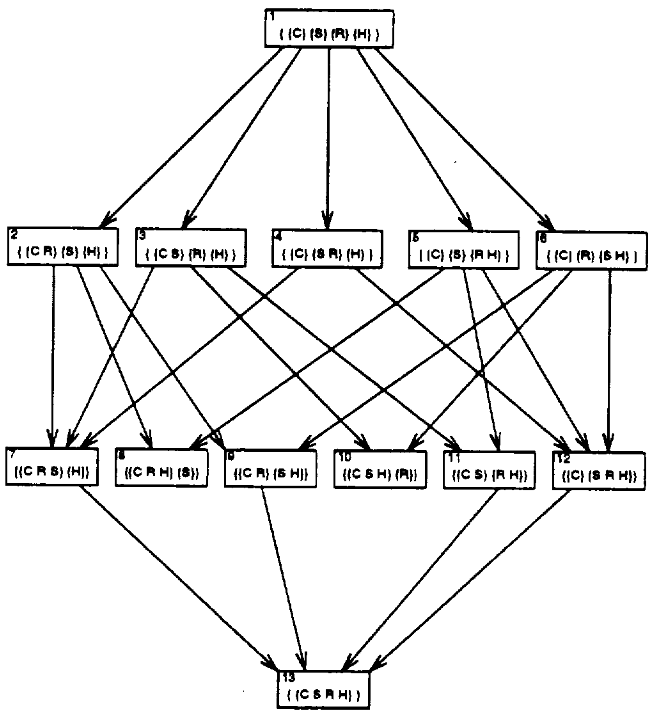
Since each assembly sequence can be represented by ordered lists, it is possible to represent the set of all assembly sequences by a set of lists, each corresponding to a different assembly sequence. It is also possible to use directed graphs, and AND/OR graphs to represent the set of all assembly sequences. Figure 3 shows the direct graph of feasible assembly sequences for the assembly shown in figure 1. The AND/OR graph of assembly sequences for the assembly shown in figure 1 has been presented elsewhere [8]. Alternatively, the set of all feasible assembly sequences can be represented by sets of precedence relationships. Sections 3 and 4 below present two types of precedence relationship representations of feasible assembly sequences.

3. Precedence relationships between the establishment of one connection and the establishment of another connection

We will use the notation $c_i < c_j$ to indicate the fact that the establishment of connection c_i must precede the establishment of connection c_j. And we will use the notation $c_i \leq c_j$ to indicate the fact that the establishment of connection c_i must precede or be simultaneous with the establishment of connection c_j. Furthermore, we will use a compact notation for logical combinations of precedence relationships; for example, we will write[1] $c_i < c_j \cdot c_k$ when we mean $(c_i < c_j) \wedge (c_i < c_k)$, and we will write $c_i + c_j < c_k$ when we mean $(c_i < c_k) \vee (c_j < c_k)$.

An assembly sequence whose representation as an ordered sequence of binary vectors is $(\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_N)$ and whose representation as an ordered sequence of subsets of connections is $(\gamma_1, \gamma_2, \cdots, \gamma_{N-1})$ satisfies the precedence relationship $c_i < c_j$ if $c_i \in \gamma_a$, $c_j \in \gamma_b$, and $a < b$. Similarly, the sequence satisfies $c_i \leq c_j$ if $c_i \in \gamma_a$,

[1]The logical operation AND will be denoted either by the symbol "$\wedge$" or by the product of the two logical variables. Similarly, the logical operation OR will be denoted either by the symbol "$\vee$" or by the sum of the two logical variables.



C = cap S = stick R = receptacle H = handle

Figure 3: Directed graph of feasible assembly sequences for the assembly shown in figure 1.

$c_j \in \gamma_b$, and $a \le b$. For example, for the assembly shown in figure 1, the assembly sequence whose representation as an ordered sequence of binary vectors is

([false, false, false, false, false]
[true, false, false, false, false]
[true, true, true, false, false]
[true, true, true, true, true])

and whose representation as an ordered sequence of subsets of connections is ($\{c_1\}$ $\{c_2, c_3\}$ $\{c_4, c_5\}$) satisfies the precedence relationships $c_2 < c_4$ and $c_2 \le c_3$ but does not satisfy the precedence relationships $c_2 < c_3$ and $c_2 \le c_1$.

Each feasible assembly sequence of a given assembly can be uniquely characterized by a logical expression consisting of the conjunction of precedence relationships between the establishment of one connection and the establishment of another connection. For example, for the assembly shown in figure 1, the assembly sequence described in the previous paragraph can be uniquely characterized by the following conjunction of precedence relationships

$$(c_1 < c_2) \wedge (c_2 < c_4) \wedge (c_2 \le c_3) \wedge (c_3 \le c_2) \wedge (c_4 \le c_5) \wedge (c_5 \le c_4)$$

The set of all M feasible assembly sequences can be uniquely characterized by a disjunction of M conjunctions of precedence relationships in which each conjunction characterizes one assembly sequence. Clearly, this logical combination of precedence relationships constitutes a correct and complete representation for the set of all assembly sequences.

It is often possible to simplify this logical combination of precedence relationships using the rules of boolean algebra. Further simplification is possible if one notices that there are logical combinations of precedence relationships that cannot be satisfied by any assembly sequence. For the assembly shown in figure 1, for example, the combination $(c_1 < c_2) \wedge (c_2 < c_3) \wedge (c_3 < c_4) \wedge (c_4 < c_5)$ cannot be satisfied by any assembly sequence. These combinations can be set as don't care conditions in the simplification of the logical combination of precedence relationships.

Whenever the assembly has the two properties described below, it is possible to obtain a simple precedence relationship representation of all assembly sequences. This representation is obtained using the result of theorem 1.

The first property is:

Property 1: Given any two states s_i and s_j, not necessarily in the same assembly sequence, let γ_i and γ_j be the sets of connections that are established in assembly tasks τ_i and τ_j from s_i and s_j respectively. If

$\langle P, C_i \rangle$ is the state's graph of connections associated to s_i,
$\langle P, C_j \rangle$ is the state's graph of connections associated to s_j,
$\gamma_i \subseteq \gamma_j$,
$C_i \subseteq C_j$, and
τ_j is geometrically and mechanically feasible,

then

τ_i is geometrically and mechanically feasible.

This property corresponds to the fact that if it is geometrically and mechanically feasible to establish a set of connections (γ_j) when many other connections (C_j) have already been established, then it is also geometrically and mechanically feasible to establish fewer connections ($\gamma_i \subseteq \gamma_j$) when fewer other connections ($C_i \subseteq C_j$) have been established. Although many common assemblies have this property, there are assemblies that don't have it.

The second property is:

Property 2: If the subsets $\theta_1, \theta_2, \cdots, \theta_k$ of the set of parts P characterize stable subassemblies, then the set $\theta = \theta_1 \cup \theta_2 \cup \cdots \cup \theta_k$ also characterizes a stable subassembly.

Like in the case of property 1, many common assemblies have this second property. Yet, there are assemblies that don't have it.

Theorem 1: Given an assembly made up of N parts whose graph of connections is $\langle P, C \rangle$ (with $C = \{c_1, c_2, \cdots, c_L\}$), let

$$\{\ (\gamma_{11}\ \gamma_{21} \cdots \gamma_{(N-1)1}),\quad (\gamma_{12}\ \gamma_{22} \cdots \gamma_{(N-1)2}),\quad \cdots,\quad (\gamma_{1M}\ \gamma_{2M} \cdots \gamma_{(N-1)M})\ \}$$

be a set of M ordered sequences of subsets of connections that represent feasible assembly sequences. If the assembly has properties 1 and 2, then any ordered sequence of $N-1$ subsets of connections that represents an assembly sequence corresponds to a feasible assembly sequence if it satisfies the set of $2L$ precedence relationships:

$$c_i \leq \sum_{j=1}^{M} T_{ij} \quad i = 1, 2, \cdots, L \qquad \text{and} \qquad \sum_{j=1}^{M} H_{ij} \leq c_i \quad i = 1, 2, \cdots, L$$

where

$$T_{ij} = \prod_{k=1}^{L} \lambda_{ik} \quad \text{with} \quad \lambda_{ik} = \begin{cases} c_k \text{ if } c_k \in \gamma_{lj} \text{ and } l \geq i \\ \text{true otherwise} \end{cases}$$

$$H_{ij} = \prod_{k=1}^{L} \lambda_{ik} \quad \text{with} \quad \lambda_{ik} = \begin{cases} c_k \text{ if } c_k \in \gamma_{lj} \text{ and } l \leq i \\ \text{true otherwise.} \end{cases}$$

The sum and the product in this theorem are the logical operations OR and AND respectively. Each term T_{ij} (for $i = 1, 2, \cdots, L$, and for $j = 1, 2, \cdots, M$) is the product of the variables corresponding to the connections that are established at the same time or after the establishment of connection c_i in the j^{th} sequence. Similarly, each term H_{ij} (for $i = 1, 2, \cdots, L$, and for $j = 1, 2, \cdots, M$) is the product of the variables corresponding to the connections that are established before the establishment of connection c_i in the j^{th} sequence. Precedence relationships that have "true" on either side are always satisfied. The proof of this theorem is presented elsewhere [7].

An example will illustrate the use of theorem 1. The assembly shown in figure 1 has properties 1 and 2. For that assembly, the set of feasible assembly sequences can be obtained from the directed graph shown in figure 3. There are ten feasible assembly sequences and they are:

$(\{c_1\}\ \{c_2,c_3\}\ \{c_4,c_5\})$ $(\{c_1\}\ \{c_5\}\ \{c_2,c_3,c_4\})$ $(\{c_2\}\ \{c_1,c_3\}\ \{c_4,c_5\})$
$(\{c_2\}\ \{c_4\}\ \{c_1,c_3,c_5\})$ $(\{c_3\}\ \{c_1,c_2\}\ \{c_4,c_5\})$ $(\{c_3\}\ \{c_4,c_5\}\ \{c_1,c_2\})$
$(\{c_4\}\ \{c_2\}\ \{c_1,c_3,c_5\})$ $(\{c_4\}\ \{c_3,c_5\}\ \{c_1,c_2\})$ $(\{c_5\}\ \{c_1\}\ \{c_2,c_3,c_4\})$
$(\{c_5\}\ \{c_3,c_4\}\ \{c_1,c_2\})$

Applying the result of theorem 1 to the above set of feasible sequences for the assembly shown in figure 1, the precedence relationships having connection c_1 alone on one side are:

$$c_1 \le c_2\cdot c_3\cdot c_4\cdot c_5 + c_2\cdot c_3\cdot c_4\cdot c_5 + c_3\cdot c_4\cdot c_5 + c_3\cdot c_5 + c_2\cdot c_4\cdot c_5 + c_2 + c_3\cdot c_5 + c_2 + c_2\cdot c_3\cdot c_4 + c_2$$

and

$$\text{true} + \text{true} + c_2\cdot c_3 + c_2\cdot c_3\cdot c_4\cdot c_5 + c_2\cdot c_3 + c_2\cdot c_3\cdot c_4\cdot c_5 + c_2\cdot c_3\cdot c_4\cdot c_5 + c_2\cdot c_3\cdot c_4\cdot c_5 + c_5 + c_2\cdot c_3\cdot c_4\cdot c_5 \le c_1 .$$

Using the rules of boolean algebra, these two precedence relationships can be simplified yielding

$$c_1 \le c_2 + c_3\cdot c_5 \qquad \text{and} \qquad \text{true} \le c_1 .$$

The second precedence relationship is always satisfied and can be ignored. Similarly, applying the result of theorem 1, simplifying the logical expressions, and deleting those precedence relationships that have "true" on either side, we obtain four additional precedence relationships. The resulting set of precedence relationships is:

$$c_1 \le c_2 + c_3\cdot c_5 \qquad c_2 \le c_1 + c_3\cdot c_4 \qquad c_3 \le c_1\cdot c_5 + c_2\cdot c_4 \qquad c_4 \le c_5 + c_2\cdot c_3 \qquad c_5 \le c_4 + c_1\cdot c_3 . \qquad (Set\ 1)$$

Set 1 of precedence relationships still contains some redundancies and can be shown to be equivalent to

$$c_3 \le c_1\cdot c_5 + c_2\cdot c_4 . \qquad (Set\ 2)$$

It should be noticed that an unfeasible assembly sequence, such as the assembly sequence whose representation as an ordered sequence of subsets of connections is $(\{c_2\}\ \{c_5\}\ \{c_1,c_3,c_4\})$, does not satisfy Set 2 of precedence relationships. It should also be noticed that there are ordered sequences of subsets of connections, such as $(\{c_3\}\ \{c_1,c_4\}\ \{c_2,c_5\})$, that do not represent an assembly sequence, but satisfy Set 2 of precedence relationships. The precedence relationships obtained using theorem 1 can only discriminate the feasible from the unfeasible assembly sequences. The information in the assembly's graph of connections allows the discrimination of assembly sequences from ordered sequences of subsets of connections that do not correspond to assembly sequences.

Theorem 1 is a sufficient condition. The set of precedence relationships obtained using this theorem is correct but not necessarily complete. But if the set of M ordered sequences of subsets of connections includes the representations of all feasible assembly sequences, then the resulting set of precedence relationships constitutes a correct and complete representation of the feasible assembly sequences.

4. Precedence relationships between the establishment of one connection and states of the assembly process

We will use the notation $c_i \rightarrow S(\underline{x})$ to indicate that the establishment of the i^{th} connection must precede any state s of the assembly process for which the value of the logical function $S(\underline{x})$ is true. The argument of $S(\underline{x})$ is the L-dimensional binary vector representation of the state s. We will use a compact notation for logical combinations of precedence relationships. For example, we will write $c_i + c_j \rightarrow S(\underline{x})$ when we mean $[c_i \rightarrow S(\underline{x})] \vee [c_j \rightarrow S(\underline{x})]$.

An assembly sequence whose representation as an ordered sequence of binary vectors is $(\underline{x}_1\,\underline{x}_2 \cdots \underline{x}_N)$ and whose representation as an ordered sequence of subsets of connections is $(\gamma_1\,\gamma_2 \cdots \gamma_{N-1})$ satisfies the precedence relationship $c_i \rightarrow S(\underline{x})$ if

$$S(\underline{x}_k) \Rightarrow \exists l\,[(l<k)\wedge(c_i \in \gamma_l)] \qquad \text{for } k=1,2,\cdots,N.$$

For example, for the assembly shown in figure 1, the assembly sequence whose representation as an ordered sequence of binary vectors is

([false, false, false, false, false]
[true, false, false, false, false]
[true, true, true, false, false]
[true, true, true, true, true])

and whose representation as an ordered sequence of subsets of connections is ($\{c_1\}$ $\{c_2, c_3\}$ $\{c_4, c_5\}$) satisfies the precedence relationship $c_1 \to x_2 \cdot x_3$ because the only states for which $S(\underline{x}) = x_2 \cdot x_3$ is true are the third and the fourth, and the establishment of connection c_1 occurs on the first assembly task. This sequence does not satisfy the precedence relationship $c_4 \to x_1 \cdot x_2 \cdot x_3$ because for the third state the value of $S(\underline{x}) = x_1 \cdot x_2 \cdot x_3$ is true but the establishment of connection c_4 occurs on the third assembly task, which occurs after the third state.

Let Ψ_S be the set of assembly states that never occur in any feasible assembly sequence. These include the unstable assembly states, the stable states from which the final state cannot be reached, and the states that cannot be reached from the initial state. Let $\Psi_X = \{\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_J\}$ be the set of L-dimensional binary vectors that represent the states in Ψ_S. Every element $\underline{x}_j$ of Ψ_X is such that the value of the logical function $G(\underline{x}_j)$ is true, where

$$G(\underline{x}) = G(x_1, x_2, \cdots, x_L) = \sum_{k=1}^{K} \prod_{l=1}^{L} \lambda_{kl}. \qquad (Eq.\ 1)$$

The sum and the product in equation 1 are the logical operations OR and AND respectively, and λ_{kl} is either the symbol x_l if the l^{th} component of $\underline{x}_k$ is true, or the symbol $\bar{x}_l$ if the l^{th} component of $\underline{x}_k$ is false. In many cases the expression of $G(\underline{x})$ can be simplified using the rules of boolean algebra. Allowing for simplifications, but keeping the logical function as a sum of products[2], equation 1 can be rewritten as

$$G(\underline{x}) = \sum_{j=1}^{J'} g_j(\underline{x}) \qquad (Eq.\ 2)$$

where each term $g_j(\underline{x})$ is the product of a subset of $\{x_1, x_2, \cdots, x_L, \bar{x}_1, \bar{x}_2, \cdots, \bar{x}_L\}$ that does not include both x_i and $\bar{x}_i$ for any i. Each term $g_j(\underline{x})$ can be rewritten grouping all the nonnegated variables first and all the negated variables last, i.e., $g_j(\underline{x}) = x_a \cdot x_b \cdot \cdots \cdot x_h \cdot \bar{x}_p \cdot \bar{x}_q \cdot \cdots \cdot \bar{x}_z$.

Any assembly sequence that includes a state that is in Ψ_S is unfeasible. Therefore, a necessary condition for the feasibility of an assembly sequence whose representation as an ordered list of binary vectors is $(\underline{x}_1\, \underline{x}_2 \cdots \underline{x}_N)$ is that $G(\underline{x}_1) = G(\underline{x}_2) = \cdots = G(\underline{x}_N) = \text{false}$. This is equivalent to $g_j(\underline{x}_i) = \text{false}$ for $i = 1, 2, \cdots, N$ and for $j = 1, 2, \cdots, J'$. If the assembly has property 1 (see section 3), this condition is also sufficient. Furthermore, if $(\underline{x}_1\, \underline{x}_2 \cdots \underline{x}_N)$ is an ordered list of binary vectors that represents an assembly sequence, the condition $g_j(\underline{x}_1) = g_j(\underline{x}_2) = \cdots = g_j(\underline{x}_N) = \text{false}$ corresponds to a precedence relationship. These facts are established by the following theorem. (The proof of this theorem is presented elsewhere [7].)

Theorem 2: Given an assembly made up of N parts whose graph of connections is $\langle P, C \rangle$ (with $C = \{c_1, c_2, \cdots, c_L\}$), let

$$G(\underline{x}) = \sum_{j=1}^{J'} g_j(\underline{x})$$

be a disjunctive normal form of the logical function that is true if and only if $\underline{x}$ is a binary-vector representation of a state that does not occur in any feasible assembly sequence. Let A_j be the set containing the indexes of the variables that are asserted in $g_j(\underline{x})$. Let N_j be the set containing the indexes of the variables that are negated in $g_j(\underline{x})$. If the assembly has property 1, and if $(\gamma_1\, \gamma_2 \cdots \gamma_{N-1})$ is an ordered sequence of subsets of connections that represents an assembly sequence, then $(\gamma_1\, \gamma_2 \cdots \gamma_{N-1})$ satisfies the set of precedence relationships

$$\sum_{k \in N_j} c_k \to \prod_{i \in A_j} x_i \quad \text{for } j = 1, 2, \cdots, J'$$

if and only if it corresponds to a feasible assembly sequence.

[2]This form of a logical function is commonly referred to as *disjunctive normal form* [4].

An example will illustrate the use of theorem 2. For the assembly shown in figure 1, which has property 1, $\Psi_X = \{ [\text{false}, \text{true}, \text{false}, \text{false}, \text{true}], [\text{true}, \text{false}, \text{false}, \text{true}, \text{false}] \}$ (these binary vectors correspond to nodes 8 and 10 in the directed graph of assembly states shown in figure 3). Therefore,

$$G(\underline{x}) = G(x_1, x_2, x_3, x_4, x_5) = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5 + x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \cdot \bar{x}_5$$

In this case the expression of $G(\underline{x})$ cannot be further simplified and we have

$$g_1(\underline{x}) = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5 \qquad A_1 = \{2, 5\} \qquad N_1 = \{1, 3, 4\}$$
$$g_2(\underline{x}) = x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \cdot \bar{x}_5 \qquad A_2 = \{1, 4\} \qquad N_2 = \{2, 3, 5\}.$$

Therefore, the precedence relationships are:

$$c_1 + c_3 + c_4 \rightarrow x_2 \cdot x_5 \qquad c_2 + c_3 + c_5 \rightarrow x_1 \cdot x_4 \qquad (Set\ 3)$$

A simpler set of precedence relationships can be obtained if in the simplification of $G(\underline{x})$ we set the nonstate vectors as don't care conditions. For the assembly shown in figure 1, the set of precedence relationships

$$c_1 \Rightarrow x_2 \cdot x_5 \qquad c_2 \Rightarrow x_1 \cdot x_4 \qquad (Set\ 4)$$

was obtained in the same fashion as Set 3, except for setting the nonstate vectors as don't care conditions in the simplification of $G(\underline{x})$. Set 4 is simpler and yet equivalent to Set 3.

It should be noticed that an unfeasible assembly sequence, such as the assembly sequence whose representation as an ordered sequence of subsets of connections is $(\{c_2\}\ \{c_5\}\ \{c_1, c_3, c_4\})$, does not satisfy both sets of precedence relationships above (i.e. Sets 3 and 4). It should also be noticed that there are ordered sequences of $N-1$ subsets of connections and their corresponding ordered sequence of binary vectors, such as $(\{c_1\}\ \{c_2\}\ \{c_3, c_4, c_5\})$, and

([false, false, false, false, false]
[true, false, false, false, false]
[true, true, false, false, false]
[true, true, true, true, true]),

that do not represent an assembly sequence, but satisfy Sets 3 and 4 of precedence relationships. The precedence relationships obtained using the result of theorem 2 can only discriminate the feasible from the unfeasible assembly sequences. The information in the assembly's graph of connections allows the discrimination of assembly sequences from ordered sequences of subsets of connections that do not correspond to assembly sequences.

In order to be able to discriminate the representations of feasible assembly sequences from any sequence of $N-1$ subsets of connections, the set Ψ_X must also include all nonstate vectors. For the assembly shown in figure 1 there are 13 distinct assembly states, two of which do not occur in any feasible assembly sequence. Since there are 32 5-dimensional binary vectors, there are 19 5-dimensional nonstate vectors for the assembly shown in figure 1. Let $G(\underline{x})$ be the logical function that is true if and only if $\underline{x}$ is one of these 21 $(2+19)$ 5-dimensional vectors. Simplifying this function, we obtain

$$G(\underline{x}) = \bar{x}_1 \cdot x_2 \cdot x_5 + \bar{x}_1 \cdot x_2 \cdot x_3 + x_1 \cdot \bar{x}_2 \cdot x_4 + x_1 \cdot \bar{x}_2 \cdot x_3 + x_1 \cdot x_2 \cdot \bar{x}_3 + \bar{x}_3 \cdot x_4 \cdot x_5 + x_3 \cdot \bar{x}_4 \cdot x_5 + x_3 \cdot x_4 \cdot \bar{x}_5$$

Therefore, the precedence relationships are:

$$c_1 \rightarrow x_2 \cdot x_5 \qquad c_1 \rightarrow x_2 \cdot x_3 \qquad c_2 \rightarrow x_1 \cdot x_4 \qquad c_2 \rightarrow x_1 \cdot x_3$$
$$c_3 \rightarrow x_1 \cdot x_2 \qquad c_3 \rightarrow x_4 \cdot x_5 \qquad c_4 \rightarrow x_3 \cdot x_5 \qquad c_5 \rightarrow x_3 \cdot x_4 \qquad (Set\ 5)$$

The ordered sequences of subsets of connections $(\{c_1\}\ \{c_2\}\ \{c_3, c_4, c_5\})$, which does not correspond to an assembly sequence but satisfies Sets 3 and 4 of precedence relationships does not satisfy Set 5. But it should be noticed that Set 5 of precedence relationships will be "satisfied" for ordered sequences of subsets of connections containing less than $N-1$ subsets. For example, the sequence $(\{c_1, c_2, c_3\}\ \{c_4, c_5\})$ "satisfies" Set 5 of precedence relationships. Yet, this sequence does not correspond to a feasible assembly sequence because it does not contain exactly $N-1$ subsets of connections.

5. Conclusion

Two types of precedence relationships that can be used to represent assembly sequences were addressed: precedence relationships between the establishment of one connection and the establishment of another connection, and precedence relationships between the establishment of one connection and states of the assembly process.

The problem of guaranteeing the correctness and completeness of precedence relationship representations of assembly sequences was solved for the class of assemblies that have properties 1 and 2 described in section 3.

In previous work [9] we have described the generation of the correct and complete AND/OR graph representation of assembly sequences. The correspondence between the AND/OR graph and the directed graph representations has also been established [10]. The results presented in this paper can be used to generate correct and complete precedence relationship representations of assembly sequences from the AND/OR graph. These results can also be used in proving the correctness and completeness of algorithms for the generation of mechanical assembly sequences that yield precedence relationship representations.

Acknowledgements

We thank Thomas L. De Fazio for helpful discussions and James Daniell for making the software for simplification of boolean functions available to us. The responsibility for the paper, of course, remains with the authors.

This research was supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (Brazil), the Jet Propulsion Laboratory of the California Institute of Technology, and The Robotics Institute of Carnegie Mellon University.

References

[1] N. Boneschanscher et al. Subassembly Stability. In *Proceedings of AAAI-88*, pages 780-785. Morgan Kaufman, August, 1988.

[2] A. Bourjault. *Contribution a une Approche Méthodologique de L'Assemblage Automatisé: Elaboration Automatique des Séquences Opératoires.* Thèse d'État, Université de Franche-Comté, Besançon, France, November, 1984.

[3] T. L. De Fazio and D. E. Whitney. Simplified Generation of All Mechanical Assembly Sequences. *IEEE Journal of Robotics and Automation* RA-3(6):640-658, December, 1987. See corrections on same journal, RA-4(6):705-708, December, 1988.

[4] H. D. Ebbinghaus et al. *Mathematical Logic*. Springer Verlag, 1984.

[5] B. R. Fox. *A Representation for Serial Robotic Tasks*. PhD thesis, Computer Science, University of Missouri-Rolla, 1987.

[6] B. R. Fox and K. G. Kempf. Opportunistic Scheduling for Robotics Assembly. In *1985 IEEE International Conference on Robotics and Automation*, pages 880-889. IEEE Computer Society, 1985.

[7] L. S. Homem de Mello. Forthcoming PhD Thesis. Department of Electrical and Computer Engineering, Carnegie Mellon University.

[8] L. S. Homem de Mello and A. C. Sanderson. AND/OR Graph Representation of Assembly Plans. In *Proceedings of AAAI-86*, pages 1113-1119. Morgan Kaufmann, 1986.

[9] L. S. Homem de Mello and A. C. Sanderson. *Automatic Generation of Mechanical Assembly Sequences*. Technical Report CMU-RI-TR-88-19, The Robotics Institute - Carnegie Mellon University, December, 1988.

[10] L. S. Homem de Mello and A. C. Sanderson. Task Sequence Planning for Assembly. In *IMACS World Congress '88 on Scientific Computation*. Paris, July, 1988.

[11] L. S. Homem de Mello and A. C. Sanderson. Planning Repair Sequences using the AND/OR Graph Representation of Assembly Plans. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 1861-1862. Computer Society Press of the IEEE, April, 1988.

[12] M. M. Lui. *Generation and Evaluation of Mechanical Assembly Sequences Using the Liaison-Sequence Method*. Master's thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, May, 1988. Also published as Report CSDL-T-990, The Charles Stark Draper Laboratory, Inc.

[13] A. C. Sanderson, M. A. Peshkin, and L. S. Homem de Mello. Task Planning for Robotic Manipulation in Space Applications. *IEEE Transactions on Aerospace and Electronic Systems* 24(5), September, 1988.

N90-29[illegible]

USING MULTIPLE SENSORS FOR PRINTED CIRCUIT BOARD INSERTION

Deepak Sood, Michael C. Repko and Robert B. Kelley

Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

Abstract

As more and more activities are performed in space, there will be a greater demand placed on the information handling capacity of people who are to direct and accomplish these tasks. A promising alternative to full-time human involvement is the use of semi-autonomous, intelligent robot systems. To automate tasks such as assembly, disassembly, repair and maintenance, the issues presented by environmental uncertainties need to be addressed. These uncertainties are introduced by variations in the computed position of the robot at different locations in its work envelope, variations in part positioning, and tolerances of part dimensions. As a result, the robot system may not be able accomplish the desired task without the help of sensor feedback. Measurements on the environment allow real time corrections to be made to the process. This paper presents a design and implementation of an intelligent robot system which inserts printed circuit boards into a card cage. Intelligent behavior is accomplished by coupling the task execution sequence with information derived from three different sensors: an overhead three-dimensional vision system, a fingertip infrared sensor, and a six degree-of-freedom wrist-mounted force/torque sensor.

1. Introduction

Robots are still far from the flexible automation tool they were envisioned to be. Robots in the present day generally operate with minimal sensing and use a control strategy based on open loop positioning. Thus the reproducibility of the task depends upon the repeatability of the robot motion and the experimental setup used. A reliable robotic system has to be able to accommodate uncertainties in the environment due to poor repeatability of the robot, changes in the workspace, and variations in the position, orientation, and dimensions of the workpieces. For this reason feedback from the workspace is required and therefore sensors have to be used. The sensory information is used to aid the robotic system in accomplishing the desired task. Vision, tactile, force/torque, proximity, and crossfire sensors can be used.

In order for robots to perform tasks like assembly, repair, or maintenance, the robot has to be programmed using motion control primitives. Robotic assembly tasks can be programmed in various languages, some of them incorporating both the robot motion primitives and the sensory interactions. These languages include AL by Mujtaba and Goldman [1], LM by Latombe and Mazer [2], and AML by Taylor, Summers and Meyer [3]. Work on fine motion planning involving sensor-guided motions to achieve part mating has been presented by Brooks [4]. A solution to the problem of mating two parts requiring sensor based strategies to deal with geometric uncertainties has been presented by Dufay and Latombe [5]. Their strategy to handle assembly forces is based on threshold monitoring, wherein the robot motion is carried out until a sensory-based condition is met. Once the condition has been met, the motion is stopped, even if the desired goal has not been achieved.

In the literature, several approaches for controlling the robot based on the forces measured from the sensors present on the manipulator have been suggested. A survey of these strategies is given in Cutkosky and Wright [6]. Two of these approaches are based on compliance, one passive and the other active. Passive compliance uses compliant tools and is described in Nevins *et al.* [7]. Active compliance is achieved by using servo loop compensators and is described in Nevins *et al.* [7] and Nevins and Whitney [8].

In this paper, a strategy based on fuzzy set theory is used to interpret the forces and torques generated during the printed circuit board insertion process. It is quite cumbersome to derive a mathematical model to describe an assembly task, such as the one described here. If such a model were to be developed, it would be specific to the details of the particular task. Goldenberg and Bazerghi [9] present an example of a method using a mathematical model for a peg-in-hole problem. The fuzzy approach uses approximate relationships instead of a mathematical model which uses absolute, numerical quantities. In the real world of robotic assembly, the goals, constraints and consequences of robot actions are not precisely known and, therefore, cannot be modelled exactly. Thus, the decisions have to be made by means of an inference mechanism that can handle uncertain and imprecise knowledge. According to Zadeh [Yager *et al.*,10], if the gap between human intelligence and machine intelligence is to be narrowed, machines should acquire the ability to manipulate fuzzy concepts and to respond to fuzzy instructions. Fuzzy logic, based on fuzzy set theory, is used for approximate reasoning about the insertion task.

The target task presented in this paper is part of a hierarchical planning and execution system. This system maps user-specified three-dimensional part assembly tasks into various target robotic workcells, and executes these tasks efficiently using manipulators and sensors available in the workcell. This system was researched and developed in the Robotics and Automation Laboratories at Rensselaer Polytechnic Institute. Details of this system are presented in Kelley and Moed [11]. As part of the hierarchy, the vision controller and the fuzzy insertion controller are each on-line, independent processes that are experts in accomplishing specific tasks. For this reason, these two controllers are called *specialists*. The specialists developed for this work are "plugable" modules which can be executed together with other existing specialists to perform a multitude of assembly tasks within the hierarchical structure.

2. Experimental Setup

The experimental set up for the printed circuit board insertion task consists of a robot, a host computer, three basic sensors, and assembly fixtures. The first sensor is a 3D vision system comprised of two CCD cameras and four lights suspended above the robot workcell. The cameras are calibrated with respect to the robot's coordinate system using a method developed by Yakimovsky and Cunningham [12, Kwak 13]. The 3D vision system is used to find the gross location of objects in the workcell. The second sensor, a finger tip mounted crossfire sensor, complements the vision system by providing more accurate position information. The third sensor, a wrist-mounted force/torque sensor, monitors the insertion process.

The printed circuit boards are placed in the robot workcell in a fixture called the pick-up rack. The boards are to be inserted in the card cage into slots called the insertion slots. The task starts by taking a picture of the workcell with each camera. In the workcell, the general location of the printed circuit board pick-up rack and the insertion slots is known *a priori*. Thus, in each image, separate windows can be defined around the nominal locations of the pick-up rack and the insertion slots. To save time, all image processing is confined to these windows. The windows are large enough so that small variations in the position and orientation of the pick-up rack or insertion slots are accommodated. A binary image of each window is created by thresholding the image.

Blob labelling and moment generating techniques are used to find a good grasping location on the printed circuit boards. Since the printed circuit boards are standing vertically, it is desirable to grasp them at the midpoint of the top edge. Because the cameras are not looking directly at the edges of the boards, the resulting image of a board is not a line (which might correspond to the top edge of a board). Instead, the board image is a parallelogram corresponding to a two dimensional projection of the board on the camera image plane. The long edges of the parallelogram correspond to the top and bottom edges of the printed circuit board. The first order moments are used to find the centroid of this parallelogram which represents a point in the center of the board. As previously mentioned, the center point of the top edge is a desirable location for grasping the printed circuit boards. Since the cameras can be in any position, the top edge of the parallelogram in the image does not necessarily correspond to the top edge of the printed circuit board. Thus, ray casting is used to determine which edge of the parallelogram represents the top edge of the board. A ray is generated from the lens of the camera through the centroid and is projected onto the x-y plane of the work surface. If the y-component of the projection is negative, then the camera is located above and to the left of the printed circuit board. In this case it can be inferred that the top edge of the parallelogram corresponds to the top edge of the board. If the y-component is positive, then the camera is above and to the right of the printed circuit board. In this case it can be inferred that the bottom edge of the parallelogram corresponds to the top edge of the printed circuit board. The binary image is scanned from the centroid towards the top edge of the board to determine the image coordinates of the grasping location on the board. Using the parameters from the camera calibration model, the image coordinates of the grasping location are transformed to robot coordinates. A similar approach is used to find the robot coordinates of the insertion slots (the only difference is that the slot height is known *a priori*). The angular rotation of the boards and the insertion slots in the image is determined using the information available from the second order moments. Again the calibration model is used to transform the orientation to the robot coordinates. Due to limitations of the camera calibration routines and inaccuracies introduced by thresholding the image, these robot coordinates are treated as only a first estimate.

The crossfire sensor is simply constructed from an infra-red emitting diode and a photo-detector. Initially, the robot is directed by the supervisor to move the gripper above the grasping location calculated by the vision system. Then the gripper is moved straight down in small incremental steps. At each step, the finger-tip crossfire sensor is polled. When the infrared beam is broken by the top edge of the board, the photo-detector output changes drastically. Thus a reasonably accurate value for the z-coordinate of the top edge of the board is determined. Next the crossfire sensor is used to find the exact center of this edge. The gripper is moved along the length of the board until it finds one side. The process is repeated in the opposite direction to find the other side of the board. From this, the x-, y,- and z-coordinates of the grasping location are calculated and the gripper is moved to that position and the fingers are closed.

The printed circuit board is picked up by the robot and moved to a position above the insertion slot. The printed circuit board is lowered to within a few centimeters above the height of the slot guides. At this point the force/torque sensor is zeroed and the fuzzy controller is used to place the board into the slot guides and then insert it in the slot.

The fuzzy controller monitors a six-component force/torque vector which relate to changes to be made in the position and orientation of the robot. If all the components of this vector are zero, the gripper is moved in the negative z-direction in small steps which lowers the board into the guides. Otherwise, the robot is moved to the new position and orientation which is obtained by adding a correction obtained from the fuzzy controller to its current values. This is repeated until the board is well into the guides. Then the board is moved in the negative z-direction in larger steps. In this phase the weighting of the vector returned from the fuzzy controller is reduced since the board is already in the guides. The board is inserted until it is close to the top of the slot. At this point the z-force is monitored in order to seat the board into the slot without damaging it. Once the board has been inserted the gripper is opened and moved up vertically to clear the height of the guides. The robot is then directed to pick up another board and repeat the entire process.

3. Use of Fuzzy Logic

The design and synthesis of conventional controllers is based on the mathematical model of the plant and involves quantitative and numeric calculations. With the advances in the area of fuzzy logic and linguistic reasoning, fuzzy controllers are being used to control systems and replace the human operator an integral part of the control process. These controllers use strategies expressed as linguistic statements, that resemble human decision making. Holmblad and Ostergaard [14] describe the application of fuzzy logic to the computer control of a rotary cement kiln. They conclude that fuzzy logic is a practical and realistic alternative to traditional means of implementing control strategies based on mathematical models. Fuzzy logic makes reasoning in the real world possible by providing the ability to deal with a continuous range of values rather than just true or false. Elements in fuzzy sets may belong only partially to a set, in contrast to traditional set theory where elements either belong to a set or not.

When an assembly task is performed by a human, the reasoning that comes into play is often of the form:

IF <sensed condition> THEN <control action> .

Depending upon the parts being assembled, the sensed conditions could be different. In the case of inserting a card into a card cage, the sensed conditions could be of the form:

"The card is not completely aligned with both the guides"

or

"The card is not being pushed in vertically."

The information about the alignment of the card can be obtained visually. The information about the direction of application of the pushing force can be obtained either visually or through contact sensing. The control actions, likewise, could be of the form :

"Reorient the card slightly to bring it in line with the guides"

or

"Change the direction of application of the force a little to eliminate the jamming."

When a robot performs the task of a printed circuit board insertion, sensors are needed to obtain the information regarding the process. In this experiment, force/torque information is used to help the robot insert a printed circuit board into a slot whose location is determined using the overhead camera system. The reasoning involved in the robotic insertion task is of the form of the IF-THEN expression as given above. The sensed conditions involve terms related to the changes in the forces and torques observed during the process. The sensed conditions are of the form:

"The x-torque is positive big"

or

"The y-torque is negative small,"

and so forth. The logic value of a condition in ordinary Boolean logic is restricted to *true* or *false* (0 or 1). In fuzzy logic, the logic value is a measure of the fulfillment of the condition and can take any value in the interval [0, 1]. Fuzzy logic is used to express each of the terms by a unique fuzzy membership function and thus establish a value in the interval [0, 1] for a given condition. The forces and torques read from the sensor are scaled in such a fashion such that the maximum and minimum values are -100 and 100 units of force (uf = 0.2 oz) respectively. The scaling factors are formulated from the signatures obtained during the trial insertion processes. Thus each sensor reading is mapped onto a universe of discourse of -100 to 100 over which the relevant fuzzy sets are defined. A parametric representation is used to represent the fuzzy sets. For example, if a fuzzy set is defined as (a, b, c, d), this means that the membership value is 1 from a to b and goes from 1 to 0 along a straight line on either side from a to a - c and from b to b + d (c and d being positive). The representation for "negative big" is (-100, -70, 0, 20). Figure 1 shows the fuzzy sets used in the course of this experiment.

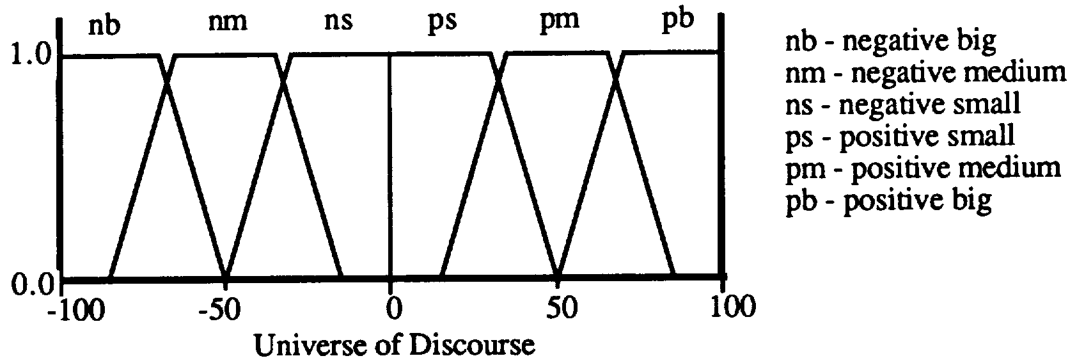


Figure 1. Fuzzy sets used in the card insertion experiment.

The forces and torques generated during a robotic assembly task reflect the status of the process: jamming, misalignment, seating, for example. These forces and torques serve as characterizing signatures of the process. Using statistical classification techniques, Fullmer [15] uses these signatures to classify the assembly process as either acceptable or unacceptable. In this paper, the force and torque signatures form a basis for the development of the fuzzy rules that are used to provide real time corrective measures to accomplish the insertion task. The 6-D force/torque signatures of an assembly process can be obtained by recording the forces and torques as the assembly proceeds. For a typical "card-into-a-slot" insertion, the signatures are obtained with respect to the position along the insertion path. In this case, the z-position of the card in the guides is used. These signatures are affected by the vibrations of the robot, and have a component that is related to the motion of the robot. Figure 2 shows these signatures for the robot going through the insertion motion without a card in the gripper. Thus, the exact values of the forces and torques generated might not be the same for two successive insertions of the same card into the same slot. It can be assumed, however, that there will be basic features and trends of the force/torque pattern which are common to successful insertions. The basic features may be parameters associated with the peaks and valleys in the signatures. In case of some uncertainty in the environment, it is difficult to classify these signals exactly and thus recognize the situation at hand. It is almost impossible to take into account all the possibilities that can arise during an assembly task and store them. The fuzzy controller implementation is able to cope with the uncertainties and to successfully interpret the signatures.

After looking at the geometry of the "card-in-a-slot" problem, a number of heuristics capturing the fuzzy reasoning process were generated. Every heuristic is represented as a condition/action fuzzy rule. The condition, or the left hand side of a rule, tests to see whether the rule is applicable to the situation at hand. The action, or the right hand side of the rule, consists of a list of actions to be performed if the rule is applicable.

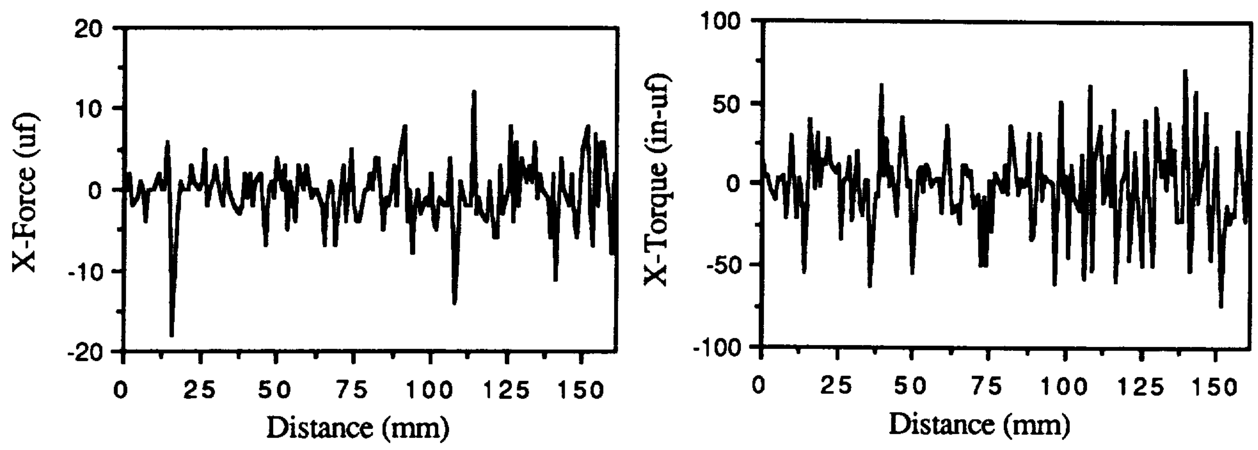


Figure 2. Typical force/torque signatures obtained by performing the insertion motion without a card in the gripper.

4. Experimental Results

In this section, the test results of the "insert_board" specialist are presented. First, the PC board is located using the "3D_vision" specialist and the position and orientation of the board is estimated. This information is provided to the robot controller and the gripper is moved to a position above the desired board. Because of the limited resolution of the overhead vision system, the crossfire sensor in the finger tips is used to determine the precise edge locations of the board. The position and orientation of the insertion slot are also determined using the vision system. Next, the board is picked up at the center of its top edge and the arm positioned above the slot where the board is to be inserted. Finally the board is inserted into the guides using the fuzzy controller with feedback from the wrist force/torque sensor.

The fuzzy rules are written based on the force/torque signatures obtained from repeated trials. Figure 3 shows two of the signatures for a perfect insertion. The increase in the z-force indicates the seating of the board into the slot. Also, the torque shows an increase in magnitude as the card is lowered into the guides because of friction and deviations in the robot motion relative to a straight line when moved in cartesian coordinates.

Consider a simple problem that might be faced by the insert_board specialist as shown in Figure 4. In this case the knowledge that is needed to insert the board into the guides is as follows:

1) A medium change in the z-force and large changes in the x- and y-torques as the board is lowered indicate that one of the corners of the board is caught on one side of the guide.

2) No appreciable changes in the forces and torques as the card is lowered into the guides indicate that the insertion is proceeding normally.

3) A large change in the z-force after the gripper has moved approximately the height of the board indicates that the board is being seated into the slot.

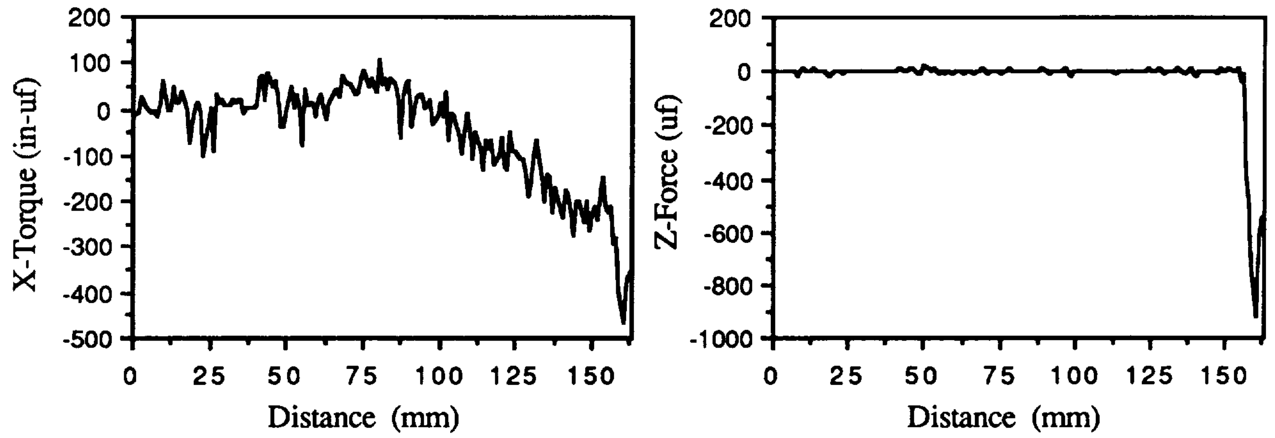


Figure 3. Typical force/torque signatures obtained by performing a perfect insertion.

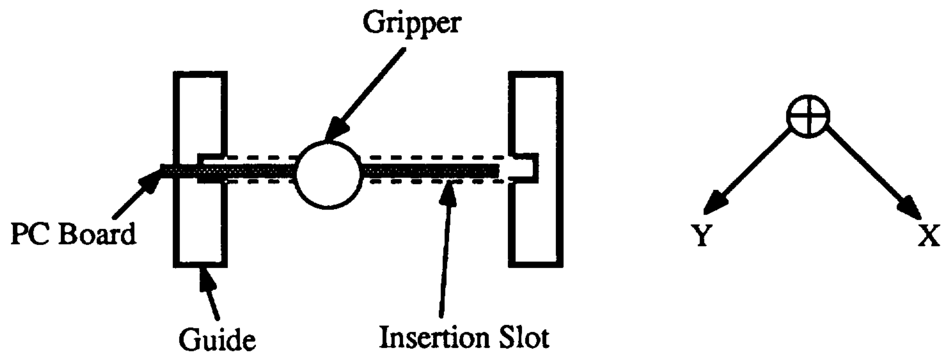


Figure 4. Typical near-miss situation to be handled by the insert board specialist and orientation of the wrist force/torque sensor coordinate system.

A typical rule written in C is shown below. The gist of the rule is that if the scaled value of the x-torque *(mapxte)* is negative big *(nb)* and the scaled value of the y-torque *(mapyte)* is positive big *(pb)* then the output, *(outset)* should be negative big *(nb).*

Rule:	*Condition*	min = *findmin*(nb, mapxte, pb, mapyte)
	Action	*truncset* (nb, min, tempset) *maxfn*(outset, tempset) .

Findmin examines the fuzzy set *nb* at the point *mapxte* and the fuzzy set *pb* at *mapyte*, and returns the minimum of the two. Then the minimum value *min* is used to truncate the fuzzy set, *nb*, and create *tempset*, a temporary set. *Maxfn* is used to accumulate the effect of the rules that influence that particular output. It takes the maximum of the two sets *outset* and *tempset* and stores it in *outset*. In this case the output variable is a change in the x-position of the robot arm. This is represented by delta-x which is obtained by defuzzifying the resulting set *outset*.

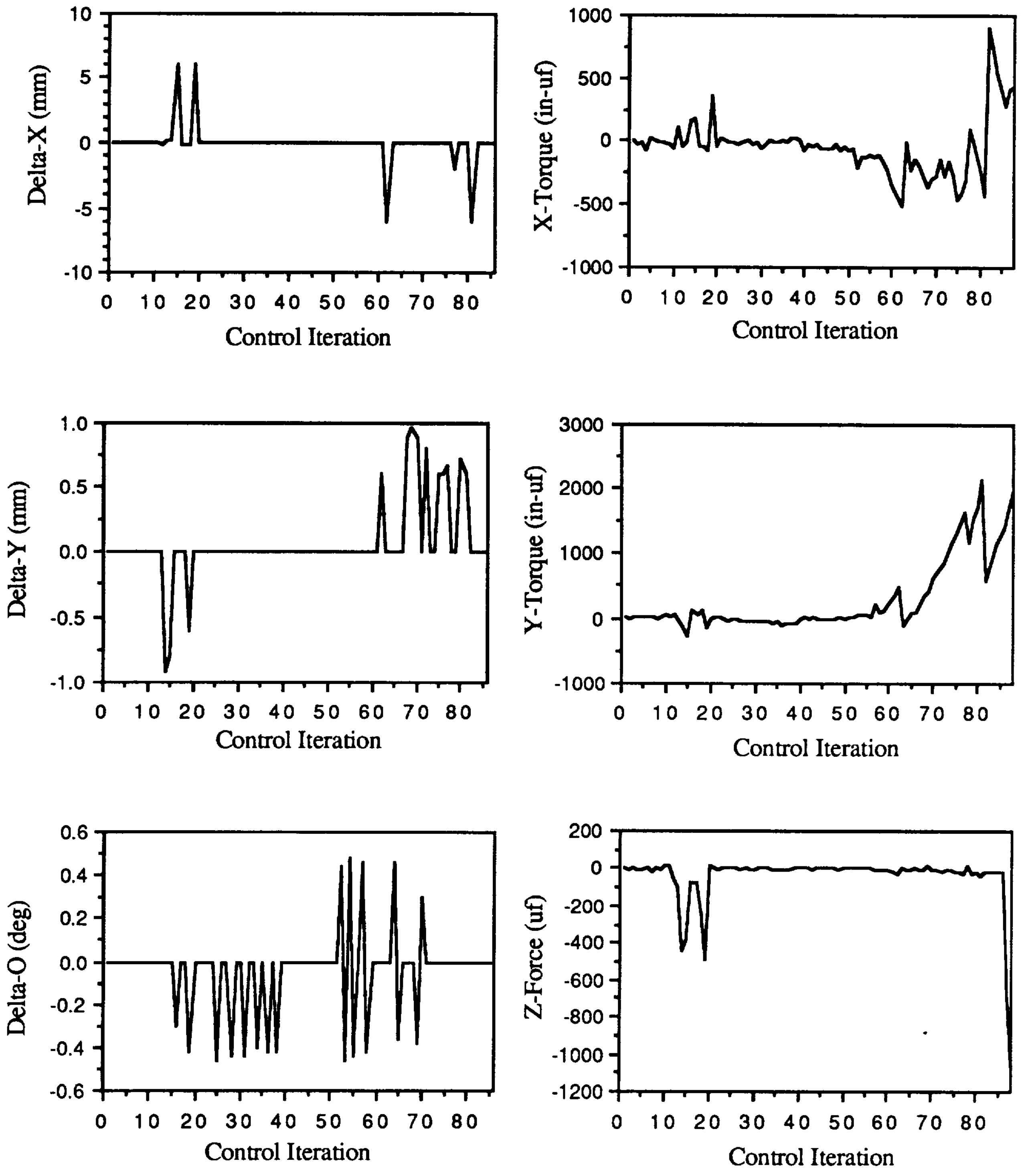


Figure 5. Results of a successful insertion process.

Figure 5 shows the results of a successful insertion process. The graphs show the modifications made to the x- and y-positions of the gripper and its rotation about the z-axis in the robot coordinate system. Also depicted are the x- and y-torques and the z-force. The control iteration axis in the graphs refers to every point where a decision is made. The following strategy is employed: If the delta-x, delta-y, and the delta-o are each below a given threshold, the board is moved in a fixed increment towards the insertion slot. If any reading is above its threshold, corrective action is taken instead. As shown in the figure, between control iterations 15-25 the near-miss situation described in Figure 4 is encountered. The delta-x and delta-y graphs show the corrective measures taken. The change in the rotation shown by delta-o relieves the torques by aligning the board. During insertion, the card became skewed due to slippage in the gripper. The iterations 65-80 show the activity in delta-x and delta-y which corrects this skew. From control iterations 60 onwards the delta-o corrections are ignored by the controller to best avoid oscillations. This explains the increase in the x- and y-torques during this period. The increase in the z-force at iteration 83 is due to the final insertion of the board into the insertion slot.

5. Conclusions

A robotic assembly system has been presented which couples the task sequence of the robot with information from different sensors. This coupling enables the system to handle uncertainties encountered during task execution. In the example task only the nominal position of the printed circuit boards and the insertion slots is known. A 3-D vision system determines the approximate position and orientation of the boards and the insertion slots. It also determines a proper grasping location on the board which is refined with the help of a fingertip crossfire sensor. A six degree-of-freedom force/torque sensor is used to sense the forces and torques generated during the assembly process. This paper described the application of fuzzy logic techniques to characterize relationships between the assembly objects and the data from the force/torque sensor. The fuzzy approach uses experienced-based approximate relationships instead of using a precise mathematical model of the insertion process. Data collected from a typical execution was shown in order to describe the information received from the sensors. A typical fuzzy control rule for this task was also included and explained. By using a fuzzy controller, printed circuit boards were successfully inserted into target insertion slots. The modeling and controller assumptions made by fuzzy set theory were validated by the repeated success of this assembly task over many trials.

6. Acknowledgements

This research is supported in part by a Grant NAGW-1333 from NASA and Grant CCA8309/188 from the US-Spain Joint Committee for Scientific and Technological Cooperation.

References

1. S. Mujtaba and R. Goldman, "AL User's Manual," 3rd ed., Report no. STAN-CS-81-889, Stanford Univ. (1981).

2. J.C. Latombe and E. Mazer, "LM: A High-Level Programming Language for Controlling Manipulators," Proc. 11th Intl. Symp. on Industrial Robots, Tokyo, Japan, pp. 683-690 (1981).

3. R.H. Taylor, P.D. Summers, and J.M. Meyer, "AML: A Manufacturing Language," Intl. J. of Robotics Research, 1(3), pp. 19-41 (1982).

4. R.A. Brooks, "Symbolic Error Analysis and Robot Planning," Intl. J. of Robotics Research 1(4), pp. 29-68 (1982).

5. B. Dufay and J.C. Latombe, "An Approach to Automatic Robot Programming Based on Inductive Learning," Intl. J. of Robotics Research, 3(4), pp. 3-20 (1984).

6. M.R. Cutkosky and P.K. Wright, "Position Sensing Wrists for Industrial Manipulators," Intl. Symp. on Industrial Robots, Vol 12, pp. 427-438 (1982).

7. J.L. Nevins, et al., "Exploratory Research in Industrial Assembly Part Mating," The Charles Stark Draper Laboratory Inc., Cambridge, MA, Seventh progress report for the National Science Foundation (1980).

8. J.L. Nevins and D.E. Whitney, "Research on Advanced Automation," Computer, pp. 24-39 (Dec. 1977).

9. A.A. Goldenberg and A. Bazerghi, "A Preview Approach to Force Control of Robot Manipulators," Mechanism and Machine Theory, 20(5), pp: 449-464 (1985).

10. R.R. Yager, S. Ovchinnikov, R.M. Tong, H.T. Nguyen, *Fuzzy Sets And Applications: Selected Papers by L.A. Zadeh*, John Wiley & Sons.

11. R.B. Kelley and M.C. Moed, "Knowledge-Based Robotic Assembly Systems," *Advances in Automation and Robotics*, Vol. 2, G.N. Saridis, ed., JAI Press Inc (1989).

12. Y. Yakimovsky and R. Cunningham, "A System for Extracting Three-Dimensional Measurements from a Stereo Pair of TV Cameras," Computer Graphics and Image Processing, 7, pp. 195-210 (1978).

13. William I. Kwak, "Design, Analysis, and Application of a 3-D Vision System in Computer Integrated Manufacturing," Ph.D dissertation, ECSE Dept., Rensselaer Polytechnic Institute, Troy, New York (1988).

14. L.P. Holmblad and J.J. Ostergaard, "Control of a Cement Kiln by Fuzzy Logic," *Fuzzy Information and Decision Processes*, M.M. Gupta and E. Sanchez, eds., North Holland Publishing Co., pp. 389-400 (1982).

15. D.M. Fullmer, "Signature Analysis of Robotic Assembly Forces - A Flexible Sensor System," Xerox Corporation Report, Webster, NY (1985).

Determining Robot Actions For Tasks Requiring Sensor Interaction

John Budenske
Honeywell Systems and Research Center
3660 Technology Drive, Minneapolis Mn., 55418
Maria Gini
Department of Computer Science
University of Minnesota, Minneapolis, Mn. 55455

Abstract

The performance of non-trivial tasks by a mobile robot has been a long term objective of robotic research. One of the major stumbling blocks to this goal is the conversion of the high-level planning goals and commands into the actuator and sensor processing controls. In order for a mobile robot to accomplish a non-trivial task, the task must be described in terms of primitive actions of the robot's actuators. Most non-trivial tasks require the robot to interact with its environment; thus necessitating coordination of sensor processing and actuator control to accomplish the task. Our contention is that the transformation from the high level description of the task to the primitive actions should be performed primarily at execution time, when knowledge about the environment can be obtained through sensors. We propose to produce the detailed plan of primitive actions by using a collection of low-level planning components that contain domain specific knowledge and knowledge about the available sensors, actuators, and sensor/actuator processing. This collection will perform signal and control processing as well as serve as a control interface between an actual mobile robot and a high-level planning system. Previous research has shown the usefulness of high-level planning systems to plan the coordination of activities such to achieve a goal, but none have been fully applied to actual mobile robots due to the complexity of interacting with sensors and actuators. This control interface is currently being implemented on a LABMATE mobile robot connected to a SUN workstation and will be developed such to enable the LABMATE to perform non-trivial, sensor-intensive tasks as specified by a planning system.*

1. Introduction

In order to perform intelligent tasks, a robot needs to interact with its environment through its sensors and actuators. Often information about the environment must be obtained before decision making, scheduling, planning, and verification of high level tasks can proceed. Since the world is large and dynamic, it is impossible to store current information a priori into the robot's internal model of the world, and thus the robot must gather the necessary information through its sensors. The gathering of information could also include the robot's observation of its own interaction with the environment through its actuators.

The intelligent behavior exhibited by the robot in its performance of a task can be depicted or represented as a plan (ordered sets of goals, events, and actions). Initially, a plan consists of a few abstract or high level goals in which information-gathering needs are implicitly understood. During the planning process, the high level goals are transformed into lower level, less abstract goals. Eventually, the implicit information needs of the plan must be identified and explicitly stated, and can be represented as "information gathering goals" within the plan.

In order for any plan to be executed, all higher level goals must eventually be mapped to the robot's sensor and actuator functions. Proper identification and mapping of the information gathering goals require the ability to:

----

*This research was sponsored in part by the National Science Foundation under grant NSF DMC 8518735.

(1) represent the relationship between high level goals and their implicit information gathering needs (ie. what kind of sensor interaction is needed for what goal),

(2) describe sensors and their ability to satisfy the information gathering needs,

(3) describe (and represent) the translation of informational needs into robot sensor and actuator actions, and

(4) replace the information gathering goals, within the plan, with sensor and actuator manipulations.

The final plan of robot sensor and actuator manipulations must be at such a level of detail so to allow proper execution of the plan by the robot. Our research will address the last three of the above abilities, and only slightly address the first one.

2. Objectives

The objective of this research is to design and develop a sensor and actuator control interface to the LAB-MATE mobile robot, resident in the Artificial Intelligence Laboratory of the University of Minnesota, Computer Science Department. This control interface has three main functions:

(1) intelligently process sensor information for use by a high-level planning and reasoning system (here on referred to as the Planning system);

(2) intelligently control the robot's sensors and actuators via the combination of directive control (from commands/plans sent to it by the Planning system) and reactive control (from reasoning about sensor data collected from the robot's environment), and

(3) provide a "knowledgeable" interface between the robot's control levels and the Planning system.

The first function of the control interface is to process the sensor data for use by a Planning system, as well as by any other sub-system of the robot. A Planning system is a symbolic level reasoning system, such as an expert system or automated planning/scheduling system. The control interface should allow any of the robot's sub-systems to access varying levels of raw and processed data from the robot's sensors. Thus both raw data and abstracted data will be available to the Planning system for reasoning.

The second function of the control interface is to allow the robot to possess a reactive behavior to its environment and yet be effectively controlled by a Planning system (here on referred to as directive behavior). A reactive behavior is the ability to conform the desired actions of the robot to correspond to the current state of the robot's environment. It requires the collection of sensor data, and the calculation of the responding actuator commands. This behavior is often cited along with survivalistic characteristics such as obstacle avoidance, and getting out of the way of big meteorites. Though reactive behavior is necessary for effective interaction with the robot's environment, directive behavior is necessary for the robot to derive the set of actions and intermediate goals which solves a high-level task. The control interface will combine both behaviors such that high-level tasks can be accomplished.

The final function is to provide a "knowledgeable" interface between the robot's control functions and the Planning system. The interface should contain within itself information on the functionality, implementation, status, etc., of the available commands to the Planning system, and this information should be accessible to the Planning system. The Planning system, then, must possess the ability to access the information, reason about it, and utilize it to interact with the robot's sensors and actuators.

3. Related Work

The problem of programming a mobile robot to perform various non-trivial and sensor-intensive tasks is not a new problem within the robotics domain. This problem has been attacked from four different directions of research over the past decade. The four are: automated planning for robots, low-level feedback-control loops, subsumption architecture, and hierarchical and distributed architectures. Each approach has definite benefits and drawbacks.

3.1. Automated Planning and Navigation for Robots

There has been a great deal of research done within the domain of automated planning and navigation, especially on domain-independent planning systems. [10,22,24,31,33] The majority of this research was aimed at creating general problem solvers which can be adapted to fit most any domain. Most of these general problem solvers are based on the same classical constructs of goal-reduction, conflict-detection/resolution, and constraint management. These classical planning systems are designed to derive a set of partially ordered primitive actions which will transform an initial world state to a given final state. Some of the classical planning research has been adapted for problems within the mobile robot domain [23,30] and there has been simulated robots running with classically-based planning systems [7,19] controlling them. Most of this research was aimed at the problems of high level task planning and did not address any of the problems of sensor processing, interaction and control.

The Stanford Cart program, [20] the HILARE robot, [6] and research by Moravec and Elfes [21] concentrated on the complexities of sensor processing, sensor data uncertainty, and actuator control within the domain of robot navigation. All attempted to solve problems in robot position uncertainty, obstacle detection/avoidance, and piloting the robot from position to position. Due to poor actuator controllers, large error propagations, long response times, and/or very weak problem solving algorithms, none of these projects attempted any non-trivial, sensor-intensive problems beyond that of simple obstacle avoidance through path planning. Kuipers and Levitt [35] both implemented navigation planning systems for different simulated robots. Both systems were concerned with how to represent space and reason about it, but neither addressed problems in sensor controls, noisy data processing and problems due to a dynamic world.

More recently, Wilkins, and Drummond [9,32] each have made specifications within their plan representations for the use of sensory data. These mechanisms assumed the availability of highly processed sensor data and did not address data processing needs, derivability, uncertainty management, and error recovery. Firby [11] has implemented a planning system to perform reactive planning within a simulation. The system has operators to determine when to use sensors, but does not have constructs to deal with errors, sensor controls and parameters, and inaccurate or noisy sensor data. Finally, Georgeff [12] has implemented a planning/control system initially in simulation and then on the SRI FLAKEY mobile robot where it exhibited 2 main problems: 1) poor response time to sensor input due to delays of interaction between the symbolic level control and the lower-level robot and sensor control; and 2) problems of control and system development due to the normal errors which embody sensor processing as well as sensor data.

Throughout most of the automated planning research for robots, there has been a great deal of simplifying assumptions on the amount of control and sensor processing and interfacing required to control an actual robot. These systems performed well in simulations based on these simplifying assumptions, but never progressed to successful implementation on actual mobile robots. Their assumptions on the management, control and interactions with sensors, sensor data and actuators were far too basic to their design, and thus limited their ability to successfully implement their designs on actual mobile robots.

3.2. Feedback-Control Loops

The research on automated planning for robots was a general problem solving approach to the robot control problem. In contrast, the next direction of research focused on narrowly defined problems where domain specific knowledge is highly applicable. The portion of this research which is best addressed towards the problem of accomplishing high-level tasks for mobile (or any) robots, has been the work in sensor-actuator feedback-control loops. Here a great deal of research has been performed to develop many different types of systems which continuously transforms a sensor input signal into an actuator control signal, controlling the actuator to accomplish a (usually very low-level) sensor-intensive task (ie. mathematically transform or through look up tables). For example, a number of sensor feedback-control loop systems have been developed for a robot arm to insert a peg into a hole based on compliant motion. [8,13,16,17,34] Compliant motion is a specification of robot motion modification in response to forces generated during the robot motion which enables the robot to carry out a task in the presence of significant sensing and control errors.

There have been many similar successful developments of sensor-actuator feedback control loops which performed single, narrowly defined, low-level, sensor-intensive tasks with actual robots. Such systems are quite sensitive to details of geometry and to error characteristics and must therefore be constructed anew for each task. Though the task complexity has increased in terms of sensor signals and actuator control, the flexibility of the control is still limited to parameterized, direct execution, with no ability to plan or coordinate multiple actions. Higher-level tasks which require flexible control for multiple actions and multiple goals have not been attempted. Also, there have been no attempts at interfacing any feedback-control loops to a Planning system such to enable more flexible control in performing higher-level tasks.

3.3. Subsumption Architecture

The subsumption architecture [5] is a layered approach to building robust sensor-actuator feedback control systems. Mobile robot control is transformed into a problem based on parallel task achieving behaviors. The key idea is that layers of a control system can run in parallel to each other. Each individual layer correspondence to a level of behavioral competence. The next higher level of overall competence (ie. improved intelligence) or enhanced capability can be obtained by adding a new layer to an existing group of layers. The new layer will run in parallel with the other layers, and will interact with the lower layers through excitation of their inputs and inhibition of their outputs. Basically, the higher layer examines the data flowing through the lower layer, and injects data into the layer, suppressing the normal flow. The lower layer runs the same regardless of the existence of the higher layer. Thus, each layer can be frozen after it is thoroughly debugged. Each layer is implemented within a set of small processors, each processor running a finite state machine, and communicating to other processors across single bit data paths.

The main emphasis of the subsumption architecture is the levels of parallel behaviors, with each behavior performing sensor-actuator feedback control tasks. Through this implemented architecture, the robot has been able to wander through an environment, avoid obstacles, search for doorways, and travel through them. Though the distributed method of control for the robot allows for many behaviors to run in parallel (thus performing many sensor-intensive tasks at once), it also disallows the ability to centrally control the robot to achieve planned goals such as are in high-level tasks. There is no ability for a reasoning system to interact with the robot, controlling it to perform high-level tasks (ie. no high-level Planning system interface).

3.4. Hierarchical and Distributed Architecture

The forth approach concentrated on developing specialized architectures for dealing with complexity of processing and data flow. Albus [1] is implementing a highly-synchronized, hierarchy of computational modules to control sensors and process sensor data as it flows though the structure. The longest planning horizons exist in the top computational module where the highest level decisions are carried out as well. At each level, goals and plans are generated, synchronized, and passed onto the appropriate submodule where they are again decomposed into subgoals, and passed on further down the hierarchy. Decomposition is based on processed input data from the sensors, information on the current state of the control hierarchy, and predictions generated by higher level modules. The system is a highly-synchronized, static, configuration of routines and subroutines.

Another hierarchical architecture enlists a declarative description of the sensors combined with procedural definitions of sensor control. This research on "Logical Sensors" [15,2] consists of logical/abstracted views of sensors and sensor processing, much like how logical I/O is used to insulate the user from the differences of I/O devices and computer operating systems. It is similar to a distributed variation of the hierarchical control approach, only it is more net-like than tree-like, and it allows more flexibility in the inter-module control structure. It provides a coherent and efficient data/control interface for acquiring of information from different sensors types.

Finally, one of the more referenced mobile robot architectures is the "blackboard", whose function is to maintain the consistency of sensor data as it is being processed, and manage the sensor control based on the informational needs of other system processes. [29,3,4,14,18] These systems have been used extensively on the DARPA ALV and other similar projects, where they have controlled mobile robots through road following, and obstacle avoidance tasks. Blackboard systems usually require that all sensor data be processed and converted into a single logic/symbolic-based format. This can hamper control processes which must first convert the data from raw sensor to numerical and then to symbolic; then reason about the resulting commands. Then the symbolic commands must be converted to low level control commands before executing them. Unlike the very quick feedback control loop approach, this approach can become very slow, limiting the robot's task performance for even the simplest of tasks.

4. Proposed Work

4.1. Rationale for the Approach and Designs

Past research has shown that the success of a robot accomplishing a task in any unstructured environment highly depends on the ability of the robot to correctly sense its environment and correctly use the information which it senses. All of the research surveyed which initially conducted the experiments in simulation and then attempted to implement in the real world have underestimated the problems with processing sensor data, controlling the sensors, and interacting with the real world such that their systems were unable to effectively control the robot.

Previous research in controlling a robot within unstructured environments have produced only limited success in achieving narrowly defined tasks (ie. road following). The low level feedback control loops may provide speedy response to sensory input, but they only can perform singly-defined tasks, and methods of combining them into goal achieving behaviors are needed. Brook's research is a step towards combining sensor-feedback control loops into intelligent, reactive behaviors, but it does not attempt to solve the problems of allowing goal directed behavior along with the reactive behavior.

One problem with the blackboard-like approaches is that they have been built from the point of view of the reasoning and planning systems. Little emphasis is placed on the needs, capabilities and limitations of the sensor and control systems. Emphasis on these needs, capabilities, and limitations is important in the design of the reasoning and planning system because in order to achieve a given task, reasoning and planning must interact exclusively with sensors and actuator controls. These reasoning systems do not contain sufficient knowledge to reason about the use of the sensors, and are thus unable to fully utilize the sensor to improve the task performance.

A second problem with complex structures and methods is the reactive delay which they exhibit. The complexity of the system causes a very slow response time, and thus not only is the main task performed very slowly, but the ability of the robot to react to unexpected events is extremely slow. A commonly used example of this problem is a robot sitting in the middle of the street, slowly reasoning about which way to move, and being hit by a truck before deciding. Slow reaction time to hazardous and dangerous events is unacceptable for any robot within an unstructured environment.

Collectively, previous research in mobile robots performing tasks in unstructured environments have not emphasized one of three important problems:

(1) the problems of sensors, sensor interaction, and sensor processing,

(2) the need to intelligently control so to accomplish high level tasks, and

(3) the need to have the robot react quickly to its environment (often called reactive behavior).

To accomplish tasks in unstructured environments requires proper interaction with the sensors. This requires intelligent management, representation, interaction and utilization of sensors and sensory data. The problems with sensors can not be ignored if success is expected.

4.2. Specific Aims of the Approach and Design

This research is aimed at overcoming the problems not emphasized by previous work. The approach is to:

(1) develop a control interface system between the robot's sensors/actuators and the Planning system's such to logically abstract the functionality of the robotic sensor/actuator processing and control from the implementation;

(2) build into the interface the capabilities to a) process and reason over sensor data so to convert it into a form usable by a Planning system, b) control sensors and actuators such to drive the robot in both directive and reactive behavior modes, c) allow reasoning, analysis, and recovery over errors occurring during the performance of tasks, and d) allow both a Planning System and the interface components to reason over the use of the interface's functionality; and

(3) test the control interface by a) adapting an available Planning system to the interface such to control the robot to successfully accomplish a high level task; and b) perform experiments on an actual robot and sensors such to attack the problems of real-time sensor control and sensor data processing;

4.3. Design of the Logical Sensor and Actuator System (LSAS)

We are currently implementing a control interface, called the Logical Sensor and Actuator System (LSAS), between the sensor and actuator drivers on the LABMATE robot and a Planning System. This control interface consists of a collection of low-level planning components that contain domain specific knowledge on the accomplishment of high-level Planning goals. These components also contain knowledge on the availability and use of sensors, actuators, and sensor/actuator processing.

The basic component-types of the LSAS are the Logical Sensor, the Sensor Driver, and the Actuator Driver. Logical Sensors are abstract views of robot sensor hardware combined with sensor processing software, which can be extended to include actuator hardware and processing such to produce sensor-actuator feedback control loops. Simple Logical Sensors will sense the environment, process the data as determined by their main functional purpose, and return an output signal back to all requesting processes. This signal can be as complex as a multi-image signal to as simple as a symbolic label. The sensing of the environment can be directly through the sensor (Sensor Driver), or through other Logical Sensors. Thus Logical Sensors can be built hierarchically to produce varying levels of processed sensor data.

Sensor Drivers and Actuator Drivers are the lowest level software interfaces to the actual sensor and actuator hardware. These entities will consist of many of the properties of Logical Sensors, but differ in that there is a one-to-one correspondence between Sensor/Actuator Drivers and the actual hardware components they drive (multiple Logical Sensors which perform the same task can exist to provide flexibility). Sensor Drivers interface directly to the sensor hardware, and control the hardware parameters in accordance to the control and data request commands sent to it by Logical Sensors. Likewise Actuator Drivers interface directly to the robot's actuator drivers and control them in accordance to the control commands sent to it by Logical Sensors.

Logical Sensors are created from the combination of Sensor Drivers, Actuator Drivers, and other Logical Sensors, thus producing a hierarchical structure of sensor-actuator processing and control functionality. For example, a Logical Sensor for searching out thermostat locations could consist of Logical Sensors for: avoiding obstacles, wandering around the environment, going through doorways, and recognizing thermostats. Each of these Logical Sensors provides a functional level of sensor (and/or) actuator processing and control, and each consists of additional sub-sensors, which also provides a sub-level of functionality.

Externally, the Logical Sensor is an abstract view of sensor-actuator functionality. Internally, Logical Sensors contain a main function, a standard set of interaction functions, and a standard set of sensor-facts. All Logical Sensors have the same base set of standard interaction functions and sensor-facts. Interaction functions include status checks on the hardware, hardware initialization routines, and control parameter manipulations. Sensor-facts include information such as the sensor's name, parameter list and types, output data types, possible side effects, power usage estimates, and a list of goals which it can aid in achieving. Each Logical Sensor subtype can add additional interaction functions and sensor-facts if needed. The complete set of functions and sensor-facts will constitute the knowledgeable interface component of the LSAS. Thus, the interface contains knowledge on how to use the Logical Sensor (parameter information); what purpose to use the Logical Sensor (goals it could achieve); what conditions will the Logical Sensor perform well under; and even how to test the Logical Sensor.

Each Logical Sensor also contains a main function which intelligently performs the sensing/actuator processing for which the Logical Sensor exists. This function can vary from performing very simple signal processing to performing complex reasoning over its current situation and sending actuator commands such to accomplish a task. For example, the main function for a simple Logical Sensor which monitors a force sensor until a threshold is reached and then signals another Logical Sensor will be one which interacts with the sensor, compares the force to the threshold, and signals upon reaching the threshold. In comparison, a more complex Logical Sensor which determines the distance to an object directly in front of the robot may first reason about the the differing characteristics of the available Logical Sensors for measuring distances (ie. sonar, infra-red, visible camera, actual distance traversed by robot's wheels,...) vs. the characteristics of the current situation, and select the most appropriate Logical Sensor. Then, it will monitor the execution of the selected Logical Sensor for possible errors. If there is an error, it will reason about its cause, and take appropriate action (which includes selecting a different Logical Sensor). The Logical Sensors which perform complex reasoning can be viewed as micro-planning systems with very small domains (ie. the domain of deriving distances), in which planning/reasoning is performed only to determine the next one or two robot actions. This type of planning is necessary due to the dynamic nature of the world, and will only work for accomplishing goals which are a few actions from success (thus the need for the Planning system to perform long-term planning and coordination of Logical Sensors).

5. Example

The intent of this research is to automatically determine mobile robot actions which will accomplish high-level sensor-intensive tasks. In order for a robot to be useful, it must be able to accomplish tasks which require interaction with its environment. Of course, interaction within unstructured environments is highly desirable, and any task requiring such interaction must be sensor-intensive. This means that a vast majority of the robot's actions require the use of contemporaneous sensor data to assure successful execution.

An example of a non-trivial, sensor-intensive task is the reconnaissance mission. One of the goals of the reconnaissance mission plan would be a follow-recon-path goal. Within this goal, the robot will enter into a known territory (previously charted, ie. by satellite), search for new objects in the environment, and attempt to record data on the object without disrupting the object's existence within the environment. This example assumes either an indoor environment (ie inside a space station), or no problems with terrain locomotion, (wheel slippage, etc.), and environmental hardening (ie. protection against the elements). These problems, though of concern in the field of robotics, are not the emphasis of this research, and thus simplifying assumptions are used.

In this example the robot is given a partially-ordered plan from the Planning system (ie. much like a plan from NOAH [22]), where one of the goals in the plan is follow-recon-path. Each of the plan's goals are given to the control interface one at a time. This example will follow the execution of the follow-recon-path goal as it is given to the control interface. The first step of the control interface in executing the follow-recon-path goal is to break it down into two operations: follow-given-path and search-for-new-object. This decomposition must take place within the control interface for two reasons. First, both operations must be executed within the control interface due to their sensor-intensive nature and the dynamic nature of the environment. Second, though each of these operations achieves a subgoal of the given goal, they must be performed in parallel in order for their combination to achieve the entire goal. Forcing the transition from Planning system to control interface to occur before such goals exist within the plan will simplify the overall process (ie. current planning research is still struggling with reasoning about parallel execution of goals).

In performing the follow-given-path operation, the robot would move through its environment following a path defined with the given goal. As the robot is following this path, it will be performing the search-for-new-object operation in parallel. This operation will consist of the robot comparing the objects found in its environment with those recorded on the given chart (or from the last time the robot was along that path). Note that a third operation of recording a map of its environment or updating that map could also be performed in parallel. Upon discovering a new (or interesting) object in its environment, the robot will then suspend the current two operations, and start a new operation which will record information on the new object. The criteria for "new or interesting" are not important at this point, but could be determined via various sensors such as magnetic fields, infra-red, etc. or via the robot performing comparisons between the current environment and an a priori chart. The objective of the new operation, record-data-on-object, is to use various sensors to record data on the newly discovered object. The recorded data can be analyzed at a later time (ie. transmitted back to a base station for more in depth analysis). A second objective of this operation is to not get too close to the new object such to disturb, disrupt, or be in danger from it. First, the robot will select two to four vantage points to collect information on the object. Interaction with the higher-level Planning system may be required to partially plan ahead as to which vantage-points to visit first as well as to predict which data-recording parameters would be necessary. The robot's Planning system only partially plans, because of world dynamics rendering complete planning untractable; thus, the robot only plans that which is necessary such to detect possible unforeseen interactions (ie. wasting time by poor selection of vantage-points).

The selection of vantage-points, the number of them, and the ordering of them will depend on the location of the new object relative to the robot, the actual existence of "good" vantage points (or any vantage points), limits of the recording devices, desired distance to maintain from the object, etc. This Planning system may need to interact with the robot's sensors such to collect this data. As the plan is being derived, its execution can begin. The robot will proceed to the first vantage point and record data on the object. If the object moves, the robot will re-select (replan) vantage-points and attempt to continue recording data on the object. If the object approaches the robot, the robot will react, and move away, keeping the desired distance. Continued aggression by the object will result in the robot aborting the data recording attempt. Upon completion or abortion of the data recording operation, the robot will return to the point of suspension, possibly requiring interaction with the Planning system, and continue on with its initial two operations (follow-given-path and search-for-new-object).

This example exhibits many characteristics of non-trivial, sensor-intensive tasks. Examples are:
1) The robot must record information on its environment for determining interesting objects.

2) The robot must perform simple navigation tasks to follow a given path and avoid obstacles.

3) The robot is able to perform parallel operations, each achieving separate subgoals of a given goal.

4) The robot is able to suspend and resume operations as dictated by by control and by input sensor data.

5) Upon finding a new object, the robot is able to determine positions for data gathering, partially plan the task of the goal of data gathering, and then accomplish that goal through execution.

6) The robot is able to react to unforeseen occurrences, and replan its actions such to achieve its goal.

This enumeration is not a complete set of characteristics for any non-trivial, sensor-intensive task, but serves as an illustration of what constitutes such a task. It is important to note the amount of sensor interaction which must occur in order for the robot to accomplish the task. The acknowledgment of this high amount of sensor interaction is the motivation of this research.

6. Summary

Mobile robot primitive actions which accomplish non-trivial, sensor-intensive tasks can be determined through the proper use and representation of the sensors and sensor data provided by the LSAS architecture. The LSAS is designed to be an control interface between high-level planning systems, and the sensor and robot hardware through hierarchical layers of sensor and actuator processing and control. Until recently, previous research only emphasized purely reactive systems (which are unable to plan, reason, and control such to achieve given goals), or purely goal directive systems (which are unable to react to unforeseen situations and threats). Within the LSAS, both directive and reactive behavior are possible through multiple parallel Logical Sensors and the combination of their outputs. Previous research which incorporated both behaviors primarily used blackboard approaches, and perform extensive amounts of processing in order to accomplish trivial tasks, thus computational time delays reduce the effectiveness of sensor response. Effective reactive behavior will be achieved in the LSAS by minimizing sensory input response time via the use of feedback control loops. Finally, one feature which many other approaches do not possess is the inclusion of a knowledgeable interface between the sensors and the Planning system. Thus, the Planning system can reason about the which, where, when, and how of using the sensors as it plans future moves.

The LSAS design addresses the need for intelligent processing of sensor data, combining directive and reactive control, and providing a knowledgeable interface to the sensors and actuators. The development of the LSAS will provide greater insight to the use of sensors for accomplishing intelligent behavior within mobile robots, and will provide further research topics in intelligent robot control, sensor interaction and control, multi-sensor fusion for the accomplishment of tasks, sensor utilization to improve task performance, and improved robot-environment interaction through error recovery and reactive behavior techniques.

7. References

1. Albus, J. S., C. R. Mclean, A. J. Barbera, and M. L. Fitzgerald, "Hierarchical Control for Robots and Teleoperators," *Proceedings of the IEEE Workshop on Intelligent Control*, pp. 39-49, 1985.
2. Allen, Peter K., "A Framework for Implementing Multi-Sensor Robotic Tasks," *Procceedings of the DARPA Image Understanding Workshop*, pp. 392-398,1987.
3. Almand, Bonni, "Sensor Information Fusion and a Robot Decisionmaking Methodology," Proceedings of the SPIE Intelligent Robots and Computer Vision vol. 579, pp. 436-441, 1985.
4. Arkin, Ronald C., Edward M. Riseman, and Allan R. Hanson, "AuRA: An Architecture for Vision-Based Robot Navigation," *Procceedings of the DARPA Image Understanding Workshop*, 1987.
5. 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.
6. Chatila, Raja and Jean-Paul Laumond, "Position Referencing and Consistent World Modeling for Mobile Robots," *IEEE International Conference on Robotics and Automation*, pp. 138-145, 1985.
7. Dean, Thomas L., R. James Firby, and David Miller, "The FORBIN Paper," YaleU/CSD/RR 550, Yale University, 1987.
8. Drake, S. H., "Using Compliance in Lieu of Sensory Feedback for Automatic Assembly", Department of Mechanical Engineering, Massachusetts Institute of Technology, 1977. Ph. D. Thesis
9. Drummond, Mark, Ken Currie, and Austin Tate, "Contingent Plan Structures for Spacecraft," JPL Workshop, p. 9, 1987.
10. Fikes, Richard E. and Nils J. Nilsson, "STRIPS: The Application of Theorem Proving to Problem Solving," *Artificial Intelligence*, vol. 2, pp. 189-208, 1971.

11. Firby, R. James, "Reactive Execution as a Basis for Strategic Planning," *AAAI Workshop on Planning for Autonomous Mobile Robots,* 1987.

12. Georgeff, M.P., A.L. Lansky, and M. Schoppers, "Reasoning and Planning in Dynamic Domains: An Experiment With a Mobile Robot," SRI Artificial Intelligents Center Technical Note 380, Artificial Intelligents Center, SRI International, Menlo Park, California, 1987.

13. Gottschlich, S.N. and A.C. Kak, "A Dynamic Approach to High-Precision Parts Mating," IEEE International Conference on Robotics and Automation, pp. 1248-1253, Computer Society Press, Philadelphia, Pa., 1988.

14. Harmon, Scott Y., "The Ground Surveillance Robot (GRS): An Autonomous Vehicle Designed to Transit Unknown Terrain," *IEEE Journal of Robotics and Automation,* vol. RA-3, no. 3, pp. 266-279, 1987.

15. Henderson, Tom, Chuck Hansen, and Bir Bhanu, "The Specification of Distributed Sensing and Control," *Journal of Robotic Systems,* vol. 2, no. 4, pp. 387-396, John Wiley & Sons, Inc., 1985.

16. Lozano-Perez, T., M. T. Mason, and R. H. Taylor, "Automatic Synthesis of Fine-Motion Strategies for Robots," *The International Journal of Robotics Research,* vol. 3, no. 1, pp. 3-24, 1984.

17. McCallion, H. and P. C. Wong, "Some Thoughts on the Automatic Assembly of a Peg and a Hole," *Industrial Automation,* vol. 2, no. 4, pp. 141-146, 1975.

18. Meystel, A., "Planning in a Hierarchical Nested Autonomous Control System," *Proceedings of the SPIE Conference on Mobile Robots,* vol. 727, pp. 42-76, Cambridge, Mass, 1986.

19. Miller, David, "Planning by Search Through Simulation," YALEU/CSD/RR #423, 1985, Ph.D. Thesis.

20. Moravec, Hans P., "The Stanford Cart and the CMU Rover," *Proceeding of the IEEE,* vol. 71, no. 7, pp. 872-884, 1983.

21. Moravec, Hans P. and A. Elfes, "High Resolution Maps from Wide Angle Sonar," International Conference on Robotics and Automation, pp. 116-121, 1985.

22. Sacerdoti, E., *A Structure for Plans and Behavior,* American Elsevier Publ. Company, 1977.

23. Sacerdoti, E., "Plan Generation and execution for robotics," Tech Note 209, SRI, 1980.

24. Tate, Austin, "A Review of Knowledge-Based Planning Techniques," AIAI-TR-9, p. 17, A.I. Applications Institute, University of Edinburgh, 1985.

29. Thorpe, Charles and Takeo Kanada, "1986 Year End Report for Road Following at Carnegie Mellon," CMU-RI-TR-87-11, p. 60, Department of Computer Science, Carnegie-Mellon University, 1987.

30. Vere, Steven A., "Planning in Time: Windows and Durations for Activities and Goals," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. PAMI-5, no. 3, pp. 246-267, 1983.

31. Wilkins, D., "Domain independent planning: representations and plan generation," *Artificial Intelligence,* vol. 22, pp. 269-301 , 1984.

32. Wilkins, David, "Recovering from Execution Errors in SIPE," *Computational Intelligence* vol. 1, no. 1, pp. 33-45, 1985.

33. Wilkins, David, Kurt Konolige, Stan Rosenschein, et al., "Research on Planning at SRI International," SRI International, 1987.

34. Xiao, Jing and Richard A Volz, "Design and Motion Constraints of Part-Mating Planning in Presence of Uncertainties," *IEEE International Conference on Robotics and Automation,* pp. 1260-1268, Computer Society Press, Philadelphia, Pa., 1988.

35. Kuipers, Benjamin J., and Tod S. Levitt, "Navigation and Mapping in Large Scale Space" *AI Magazine* vol. 9, no. 2, pp. 25-42, Summer 1988.

NASA LANGLEY RESEARCH CENTER

The Laboratory Telerobotic Manipulator Program*

J. N. Herndon, S. M. Babcock, P. L. Butler, H. M.Costello, R. L. Glassell,
R. L. Kress, D. P. Kuban, J. C. Rowe, and D. M. Williams
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831
615-576-0119

ABSTRACT

New opportunities for the application of telerobotic systems to enhance human intelligence and dexterity in the hazardous environment of space are presented by the National Aeronautics and Space Administration (NASA) Space Station Program. Because of the need for significant increases in extravehicular activity and the potential increase in hazards associated with space programs, emphasis is being heightened on telerobotic systems research and development. The Automation Technology Branch at NASA Langley Research Center currently is sponsoring the Laboratory Telerobotic Manipulator (LTM) program at Oak Ridge National Laboratory to develop and demonstrate ground-based telerobotic manipulator system hardware for research and demonstrations aimed at future NASA applications. The LTM incorporates traction drives, modularity, redundant kinematics, and state-of-the-art hierarchical control techniques to form a basis for merging the diverse technological domains of robust, high-dexterity teleoperations and autonomous robotic operation into common hardware to further NASA's research.

INTRODUCTION

New opportunities for the application of telerobotic systems to enhance human intelligence and dexterity in the hazardous environment of space are presented by the National Aeronautics and Space Administration (NASA) Space Station Program. The suited astronaut has been the mainstay of the U.S. space program to date, and this will continue. Nevertheless, with significant increases in extravehicular activity (EVA) likely and potentially increased hazards associated with future programs, heightened emphasis is being placed on telerobotic systems research and development. R&D goals are to improve overall safety and efficiency and provide significant spin-off technology to improve the productivity of the U.S. industrial sector. The Automation Technology Branch at NASA Langley Research Center currently is sponsoring the Laboratory Telerobotic Manipulator (LTM) program at Oak Ridge National Laboratory (ORNL) to develop and demonstrate ground-based telerobotic manipulator system hardware for research and demonstrations aimed at future NASA applications.

NASA plans indicate the need to rely on teleoperation for control of dexterous telerobotic systems in the construction and initial operation of the Space Station. Evolution into intelligent robotic operations is desirable. Because of present technological limitations, evolution is expected to be gradual. The unique nature of orbital operations demands that this evolution be carefully controlled. A major limitation in implementing the transition is the lack of available telerobotic hardware that can function well as a real-time teleoperator while providing a sound hardware basis for intelligent, autonomous robotic operations. The LTM is being developed as a basis for the merger of these diverse technology domains into common hardware to further NASA research.

*Research performed at Oak Ridge National Laboratory, operated by Martin Marietta Energy Systems, Inc., for the U.S. Department of Energy under Contract No. DE-AC05-840R21400, and sponsored by the National Aeronautics and Space Administration, Langley Research Center.

SYSTEM DESIGN FEATURES

Merging the mechanical and control features necessary for a force-reflecting servomanipulator and a robotic positioner into a single system is a particularly difficult task. A good force-reflecting servomanipulator designed for efficient human-in-the-loop control emphasizes end effector speed for good master response to human control input, good slave tracking of the master, high-joint back-drivability for force reflection, and low reflected friction and inertia to minimize operator fatigue. On the other hand, a good robotic positioner emphasizes end effector accuracy, end effector speeds, and mechanical and control stiffness. A major objective of the LTM design is to bridge the gap between these two technologies by providing the most important design and operational parameters of each. The LTM prototype system is composed of two force-reflecting slave arms (Fig. 1) and two force-reflecting master arms (Fig. 2) with a digital-based control system providing bilateral, position-position, force-reflecting control. End effector robotic control with kinematic redundancy resolution is being implemented.[1] Finally, joint-level robotic control of position and velocity and open-loop joint drives are provided for implementation of other robotic control options in the future.

A. Mechanical design

The LTM design uses a modular approach for joint construction, with common pitch-yaw differential joints implemented for the arm, shoulder, elbow, and wrist. An output wrist roll follows the wrist pitch-yaw differential to give a compact hemispherical wrist positioner. A simple parallel jaw gripper is provided for the slave, and a pistol grip handle is provided on the master. Each pitch-yaw joint mechanism provides these motions about orthogonal axes, and each is attached to adjacent joints by four mechanical fasteners that produce a modular mounting arrangement. This arrangement allows the LTM arms to be easily assembled and disassembled. Cabling connections are automatically engaged during mechanical connection. All cabling is routed internally to eliminate external pigtails and connectors. This modularity, shown in Fig. 3, allows the LTM arms to be easily reconfigured for changing requirements and also permits maintenance of the arms simply by replacing the failed module. Traction drives with variable loading mechanisms were chosen for torque transmission through the LTM differentials. Although traction drives have not been widely used for servocontrol applications, potentially they can provide benefits for space applications, such as zero backlash and minimal lubrication requirements. Redundant LTM kinematics provides good dexterity for work in confined spaces and allows solutions for avoiding kinematic singularities. The overall reach of 1.4 m and end effector speed of 0.9 m/s for any joint were chosen for dexterous performance as a teleoperator. All joints have an unloaded acceleration capability exceeding 1g in all directions.

The LTM has load capacities to accommodate expected requirements for orbital operation while providing counterbalanced operation for 1-g earth demonstrations. Each LTM arm has a peak load capacity of 13.6 kg and a continuous load capacity of 9.1 kg. For effective ground operation, the LTM arm is configured from joints with different torque capacities. To reduce fabrication and engineering costs, a large joint with a peak torque capacity of 186 Nm is used at both the slave shoulder and elbow positions. To optimize dexterity and minimize weight, a small joint with peak torque capacity of 49 Nm is used as the slave wrist joint. The master arms are composed entirely of small joints due to the reduced requirements for output torque. As shown in Fig. 4, each joint assembly consists of a differential drive mechanism; two dc servomotors with integral reducers, fail-safe brakes, tachometers, and optical encoders; two in-line torque sensors; and two 16-bit accuracy single-turn resolvers coupled directly to the axis of rotation at the joint output. The speed reduction ratio through the differential is approximately 3.5:1. The reducers were specially designed for LTM and utilize spring-loaded antibacklash gear trains. Commercial in-line torque sensors have been modified and incorporated directly into the joint mechanism to produce a compact arrangement. Permanent magnet fail-safe brakes coaxially mounted to each drive motor will safely support loads during power failure and are capable of supporting maximum payloads for extended periods without excessive motor heating. Their advantage is higher torque-per-unit size and weight compared to spring-set brakes.

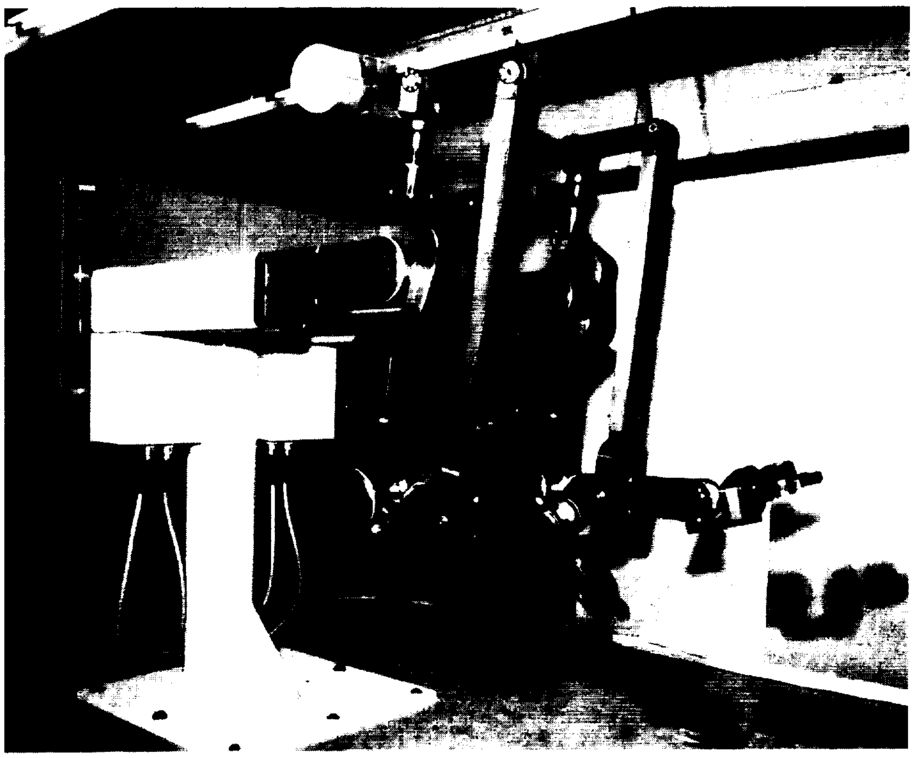


Fig. 1. LTM slave arms.

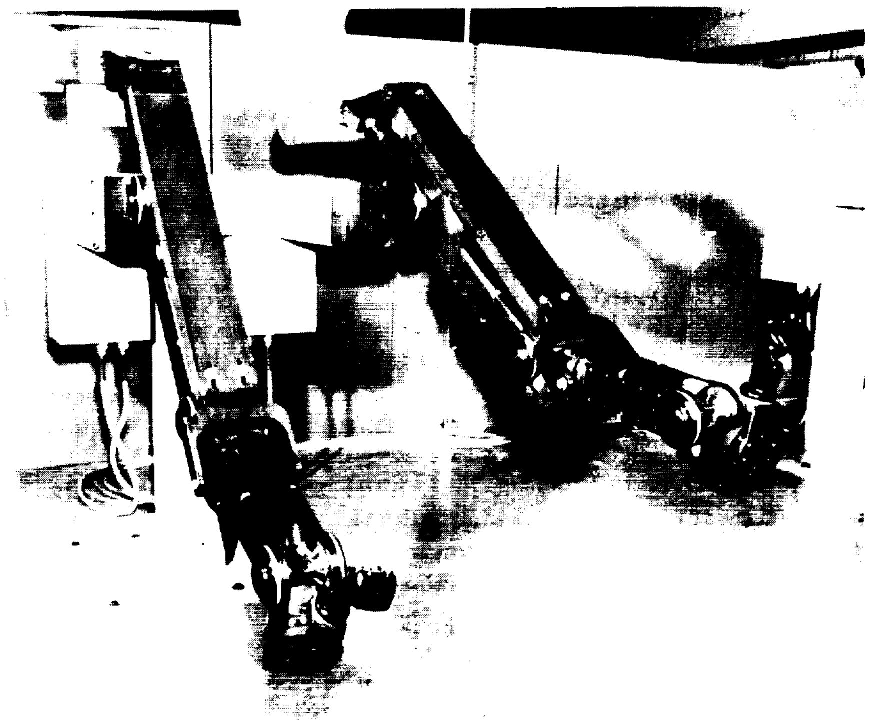


Fig. 2. LTM master arms.

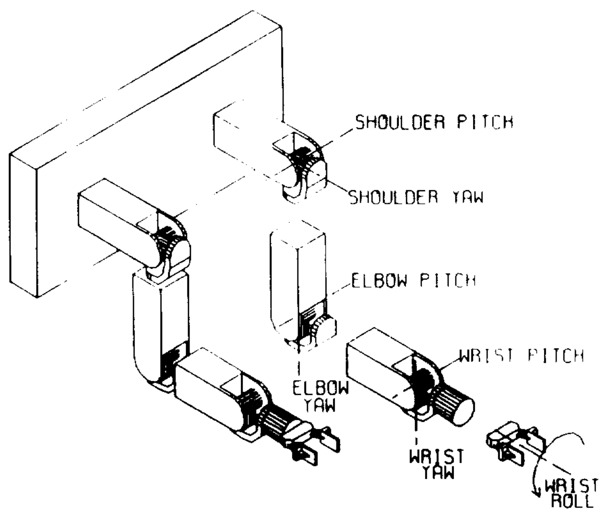


Fig. 3. LTM slave arm modularity

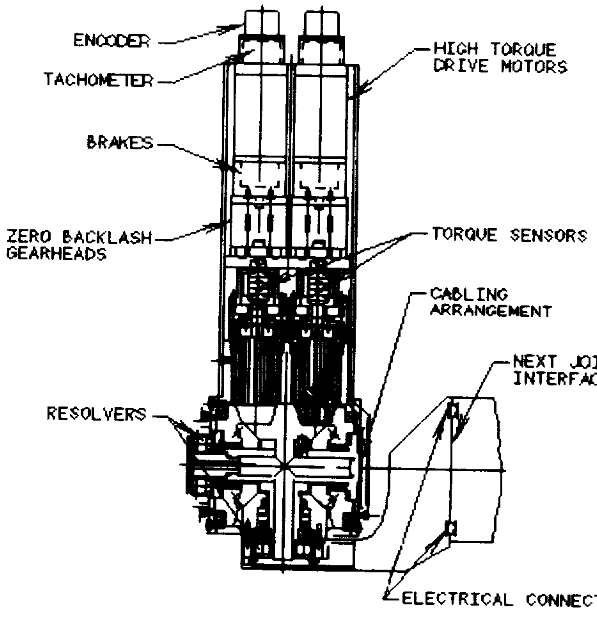


Fig. 4. LTM small pitch-yaw joint assembly.

Force transmission through the differential drive mechanism is by traction drives. Unlike force transfer through gear teeth, which generate torsional oscillation as loads transfer between the teeth, force transfer through traction is inherently smooth and steady, without backlash.[2] Two driving rollers provide input into the differential. A significant advantage in this differential setup is that each driving roller is required to transmit only one-half the total torque necessary for a particular motion, thus reducing required motor size and resulting weight. These rollers interface with two intermediate rollers that drive the pitch-yaw output roller about the pitch and yaw axes. The axis about which the pitch-yaw roller rotates depends upon the rotation direction of the driving rollers. The contact surfaces of the traction rollers are gold-plated by an ion plating process developed by NASA Lewis Research Center. This plating serves as a dry lubricant in that it prevents the substrates from making contact. The thin layer of gold is a cost-effective solution for lubrication of these rolling surfaces in space. By using resolvers directly at the output of each joint for position measurement, any creep experienced through the traction drive differential will not affect positioning characteristics of the arm.

For traction drives to function, there must be a normal force between the mating rollers that transmits torque by friction. As an alternative to the more common constant-loading mechanisms, variable loading mechanisms have been employed on the LTM in an effort to improve differential back-drivability, mechanical efficiency, and fatigue life. Constant loading mechanisms produce a constant normal load between traction drive rollers. This constant normal load must be sized to ensure adequate traction at the joint's maximum torque capacity. The obvious disadvantage of this constant normal load is that traction drive rollers and their supporting bearings are needlessly overloaded during periods of low torque transmission, not only generating extra bearing losses at low torque transmission but, more importantly, shortening the drive system fatigue life. In order to ensure adequate traction with minimum friction loss, variable loading mechanisms were developed for the LTM. These purely mechanical mechanisms produce varying normal loads between the traction rollers that are proportional to the transmitted torque. Variable loading mechanisms have been incorporated into the traction drive differential, one pair at the input rollers and one at the output pitch-yaw roller.

B. Control System Design

The LTM control system is a modular hierarchical design with expansion capabilities for future enhancements of both the hardware and software. It is based on past ORNL experiences in complex hierarchical manipulator systems[3] and the need to be consistent with the overall space station NASREM control approach.[4] A top-level block diagram illustrating the organization of the system hardware is shown in Fig. 5. At this level, the system is composed of two computer systems, one master and one slave, connected by a high-speed serial communication link to allow significant separation between master and slave arms. Each rack controls a pair of LTM arms using data acquired from sensors in the individual joints. Custom embedded computers distributed in the joints provide sensor data acquisition and data communication to the central computer systems through high-speed fiber optic links. A Macintosh II computer interfaced with the master computer system provides a graphics-based interface for system operation.

A commercial VMEbus approach is utilized for the central computer systems and is based on multiple Motorola 68020 single-board computers operating in parallel. One single-board computer coordinates the overall operation of the system, while additional single-board computers complete the control algorithm calculations required for teleoperation, robotics, and electronic counterbalancing. In addition to the single-board computers, the VME systems support digital and analog I/O, distributed communication links, terminal support, and mass storage. PWM amplifiers that provide drive signals to individual joint motors are also located in the central computer racks. The overall hardware arrangement for the master rack is shown schematically in Fig. 6.

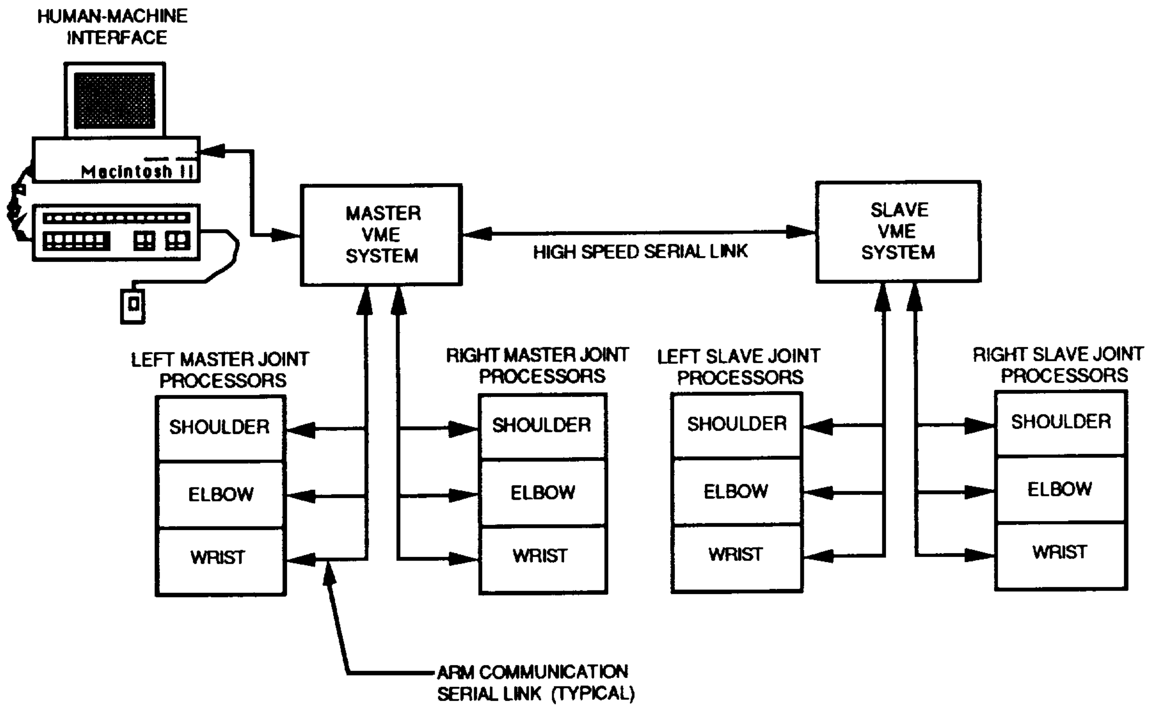

HUMAN-MACHINE INTERFACE
Macintosh II
MASTER VME SYSTEM
HIGH SPEED SERIAL LINK
SLAVE VME SYSTEM
LEFT MASTER JOINT PROCESSORS
RIGHT MASTER JOINT PROCESSORS
LEFT SLAVE JOINT PROCESSORS
RIGHT SLAVE JOINT PROCESSORS
SHOULDER
ELBOW
WRIST
ARM COMMUNICATION SERIAL LINK (TYPICAL)

Fig. 5. Block diagram on the LTM control system hardware.

Custom electronics packages were developed to reduce the number of cables required for each arm. Because the LTM utilizes an embedded cabling approach in which all power, control, and communication cables pass through the pitch/yaw joints, it was necessary to minimize the number of cables required. The custom computer packages reduce the cabling by acquiring, processing, and multiplexing the many sensor signals over serial communication links between arm modules and the VMEbus racks. The electronics packages consist of four individual systems: a joint processor logic board (JPl) in each joint, a joint processor power board (JPp) in each joint, a link processor (LP) board for each joint to interface with the VMEbus, and the fiber optic communication system. The joint processor logic board and the link processor board are high-density circuit boards using surface mount technology on both sides of a multilayer board. The JPl board is a five-layer board with 40 integrated circuits, all in surface-mount technology. The LP board is a four-layer board. Figure 7 illustrates the JPl, JPp, and LP boards.

The link processor is based on the Intel N80C196KA 16-bit microcontroller. The system has 16 kbytes of ROM to contain the startup and communication code, 4 kbytes of dual-port RAM, and 16 kbytes of SRAM to hold the application code after it is downloaded from the VME system through the DPRAM. The LP communicates with the VME system through 4 kbytes of dual-port RAM that is memory-mapped to 4-kbyte blocks in the VME memory. Communication with the JPls via the fiber optic links is controlled by an Intel N82588 2-Mbaud LAN controller. A link processor sends commands to an individual joint processor to acquire joint data and, after receiving the joint data, places the data in a portion of shared global memory containing the current world model. During operations, the LP sends data requests every millisecond independent from and asynchronous to the VME system. The LPs also pass commands and code to the joint processors from the VME system.

The joint processor logic board, like the link processor, is based on the Intel N80C196KA 16-bit microcontroller. The system also has 16 kbytes of ROM to hold the startup and communication code and 16 kbytes of SRAM to hold application code that is downloaded through the LP after startup. The JPl also utilizes the same N82588 LAN controller for communications. A joint processor acquires data from the

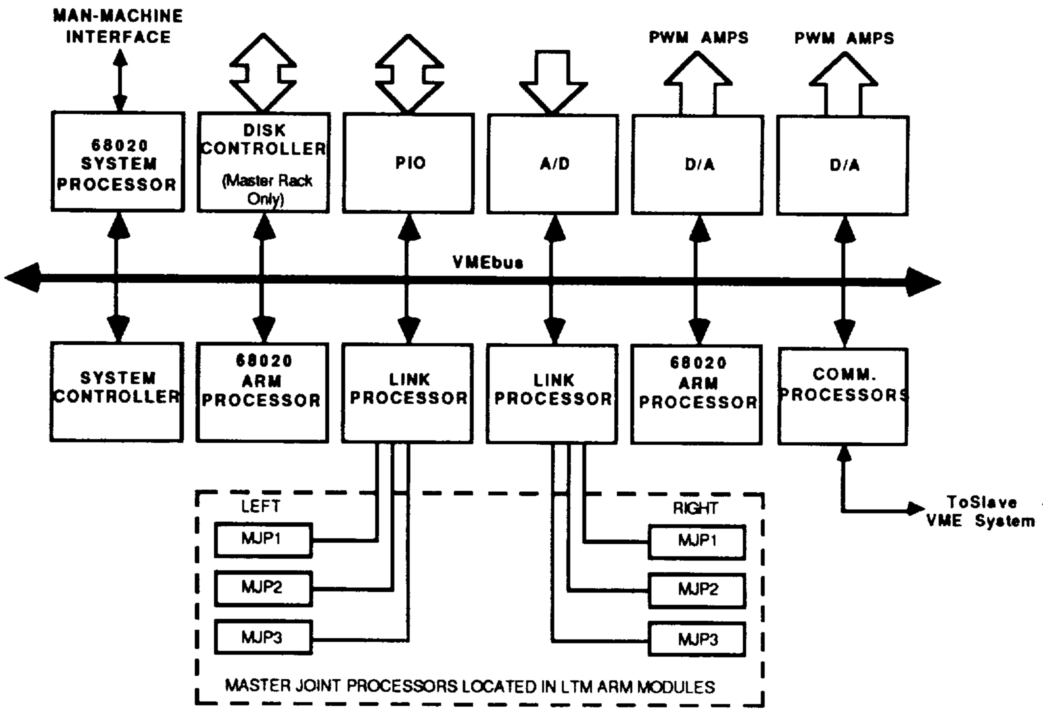


Fig. 6. Schematic of LTM master VME rack.

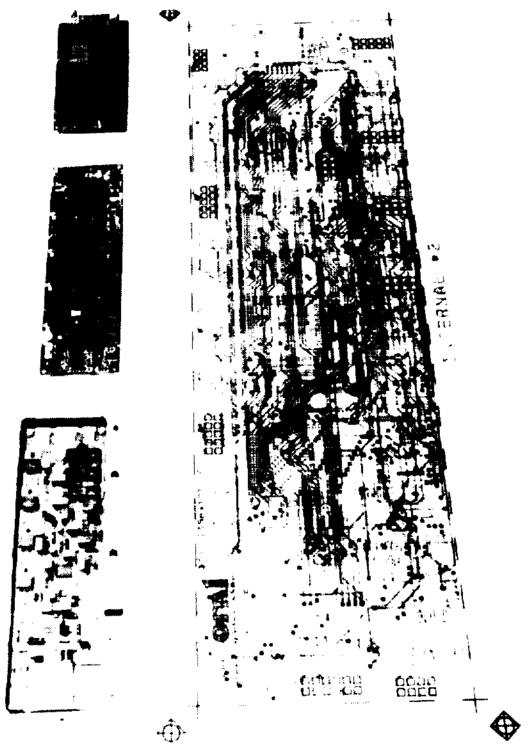


Fig. 7. LTM custom electronics hardware.

numerous sensors in a pitch/yaw joint upon demand and returns them over the fiber optic link to a paired link processor. The data consist of pitch and yaw velocity, pitch and yaw position, motor positions, motor velocities, joint torques, and joint temperatures. In addition, the JPl on the wrist joint acquires data for wrist roll and grip commands for end effector control. Man-machine interface cursor control and various mode selection button data is also acquired for the master handle.

The joint processor power board converts the 24-V dc power distributed through the arms to the +5-V dc and ±12-V dc needed by the joint processor boards. The power board also supplies power to the joint torque sensors, supplies the resolver reference drives, and contains the motor brake relays. The fiber optic system consists of two full-duplex bidirectional transceivers and a single high-strength fiber for each link and joint processor pair. The link processor transceiver is in the rack with the VME computer system, and the joint processor transceiver is on the JPp board in each joint. The transceivers use two different wavelengths of light to receive and transmit, thus providing full-duplex operation on a single fiber. A multidrop link approach could have been implemented, but the overall speed would have been significantly reduced.

The LTM software architecture, shown in Fig. 8, supports a modular hierarchical design with expansion capability for future enhancements to the system. In addition, interfaces have been defined to allow layering into hierarchical control implementations such as NASREM.[4] The operating system for the central VME computers is OS-9, a multiprogramming, multitasking, modular system that provides for position-independent code in real-time applications. Both C and FORTH are currently used for programming. In addition, FORTRAN 77, PASCAL, and BASIC are also supported if required for future developments. FORTH was chosen as the development language for the data acquisition processors distributed in the arms. FORTH allows a minimal system to have powerful debug capabilities, an important consideration with the limited ROM and RAM of the link and joint processors. In addition, the FORTH kernel is open, allowing modifications to the operating environment. FORTH has its own assembler and compiler, thus eliminating the need for a cross compiler on the VME system to generate

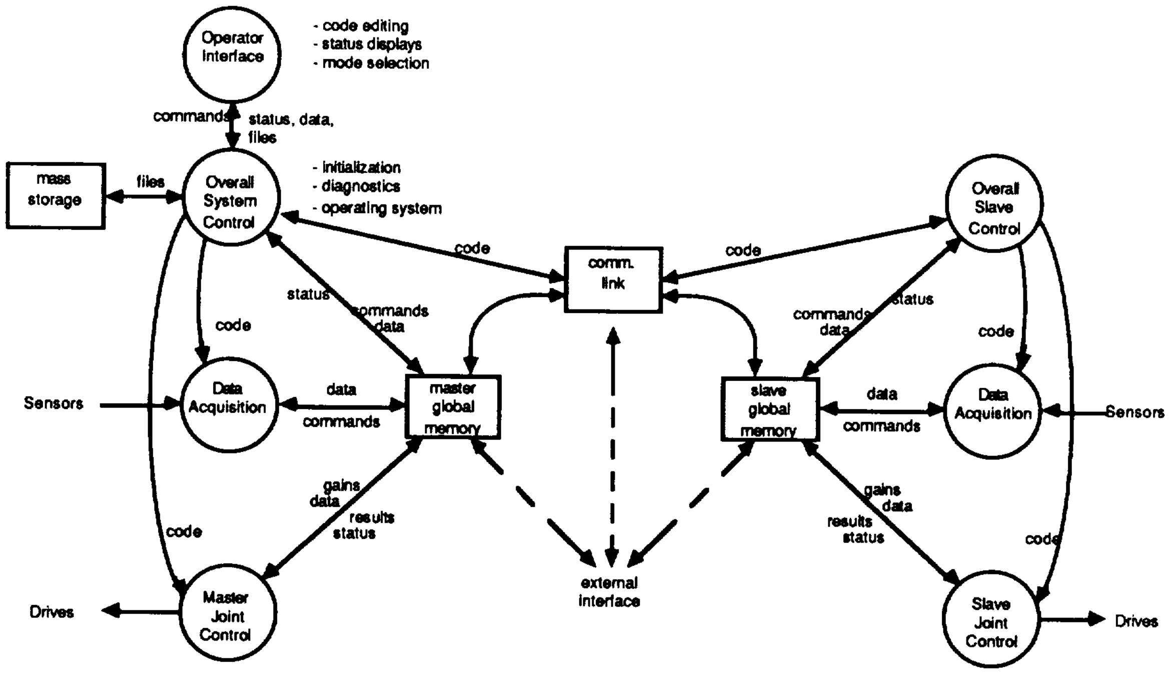


Fig. 8. LTM software architecture.

code for the custom modules. A need for user modification in this code is not expected. All higher-level code in the LTM, in which future user modification can be expected, is written in C. C is a more widely accepted language in the robotics research community, and future code upgrades and maintenance should thus be easier.

The joint level control scheme for the LTM must perform well in two diverse operating modes, a robotic mode and a bilateral, force-reflecting master-slave mode. Performance in either of these modes can be compromised by significant nonlinearities associated with the traction drive pitch-yaw joints, as well as load variations due to changes in arm configuration or payload. The basic approach for addressing these effects in the LTM is to close a torque control loop around the motor drive portion of the drive train using the in-line torque transducer. For the robotic control mode, a proportional-integral control loop for each pitch-yaw joint with decoupled input commands has been implemented. This loop is shown in Fig. 9. For the bilateral, force-reflecting master-slave mode, the pitch-yaw joint control loop shown in Fig. 9, minus the integral term has been implemented in classic bilateral, position-position control fashion.

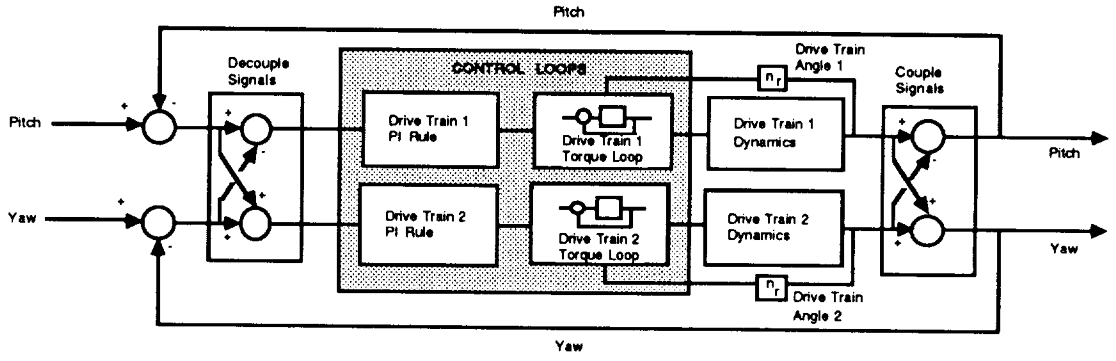


Fig. 9. Robotic control loop block diagram.

STATUS

Mechanical and control system fabrication and assembly of the first master-slave arm pair is complete. The second master-slave arm pair will be completed shortly. Initial operation and testing of the baseline master-slave system, as well as the robotic mode, has been started. It has been shown that force-reflecting servomechanisms utilizing traction drives are feasible, and that distributed electronics and universal cabling allow modularity to be implemented. The performance of this first LTM prototype is expected to confirm the expectation that a telerobotic manipulator system bridging the gap between classic teleoperated manipulators and robotic systems can be built.

REFERENCES

1. Dubey, R. V., Euler, J. A., Magness, R. B., Babcock, S. M., Herndon, J. N., "Control of the Seven-Degree-of-Freedom NASA Laboratory Telerobotic Manipulator," Proceedings of the NASA Conference on Space Telerobotics, January 31-February 2, 1989, Pasadena, California.

2. Lowenthal, S. H., Rohn, D. A., and Steinetz, B. M., "Application of Traction Drives as Servo Mechanisms," 19th Aerospace Mechanisms Symposium, May 1985.

3. Rowe, J. C., Spille, R. F., and Zimmermann, S. D., "Integrated Digital Control and Man-Machine Interface for Complex Remote Handling Systems," Proceedings of the International Topical Meeting on Remote Systems and Robotics in Hostile Environments, March 29-April 2, 1987, Pascoe, Washington.

4. Lumia, R., Fiala, J., and Wavering, A., "NASREM: Robot Control System and Testbed," from Robotics and Manufacturing, ASME Press, 1988.

ROBOTIC CONTROL OF THE SEVEN-DEGREE-OF-FREEDOM NASA LABORATORY TELEROBOTIC MANIPULATOR*

R. V. Dubey, J. A. Euler and R. B. Magness
Mechanical & Aerospace Engineering
The University of Tennessee
Knoxville, Tennessee 37996

S. M. Babcock and J. N. Herndon
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831

ABSTRACT

A computationally efficient robotic control scheme for the NASA Laboratory Telerobotic Manipulator (LTM) is presented. This scheme utilizes the redundancy of the seven-degree-of-freedom LTM to avoid joint limits and singularities. An analysis to determine singular configurations is presented. Performance criteria are determined based on the joint limits and singularity analysis. The control scheme is developed in the framework of resolved rate control using the gradient projection method, and it does not require the generalized inverse of the Jacobian. An efficient formulation for determining the joint velocities of the LTM is obtained. This control scheme is well suited for real-time implementation, which is essential if the end-effector trajectory is continuously modified based on sensory feedback. Implementation of this scheme on a Motorola 68020 VME bus-based controller of the LTM is in progress. Simulation results demonstrating the redundancy utilization in the robotic mode are presented.

1. Introduction

NASA has embarked on an extensive national project to establish a permanent human-occupied space station. To accomplish this project, significantly increased levels of dexterous human-like handling tasks will be required in orbit. This will include space station construction as well as planned and unplanned maintenance operations on the station. In addition, a significant amount of satellite repair and maintenance is expected in the future. To meet the need for sharply increased levels of dexterous handling while decreasing the levels of required human extravehicular activity, NASA has established a goal for significant use of telerobotic hardware in future space activities.

The NASA Langley Research Center has sponsored the Laboratory Telerobotic Manipulator (LTM) Prototype Project at the Oak Ridge National Laboratory to develop prototypical manipulators for use in NASA laboratories to develop and demonstrate telerobotic and robotic capabilities in an earth-based environment [1]. As a result, the LTM must be designed to be a high-quality force-reflecting teleoperator with capabilities for robotic operation. High performance under human control, low required backdriving torque, high velocity and acceleration capability, and good capacity-to-weight ratio are emphasized in the design. To provide the basis for a transition to autonomous robotic operation, features for high-quality robotic operation are also provided. These include good end-effector positioning accuracy and high mechanical and control stiffness. The LTM is also designed for modular maintainability to ease repair and reconfiguration.

The LTM arm, shown in Fig. 1, has seven degrees of freedom that provide kinematic redundancy. The arm is configured from three common pitch/yaw joints which combine to provide shoulder, elbow, and wrist joints. The

*Research performed at Oak Ridge National Laboratory, operated by Martin Marietta Energy Systems, Inc., for the U.S. Department of Energy under Contract No. DE-AC05-840R21400, and sponsored by the National Aeronautics and Space Administration, Langley Research Center.

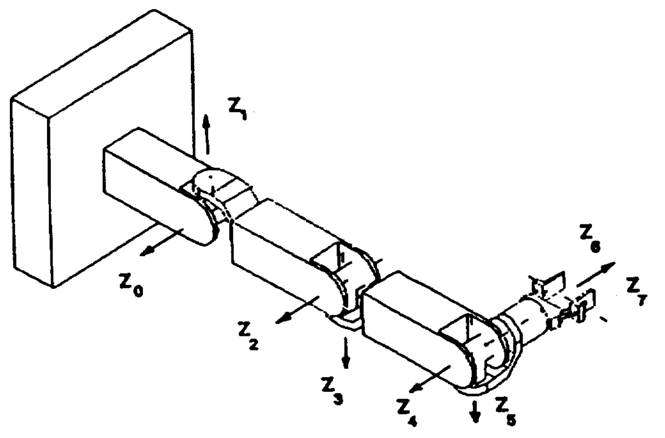


Fig.1 Laboratory Telerobotic Manipulator

Table 1. Denavit-Hartenberg table of link parameters

θ_i (deg)	d_i (m)	α_i (deg)	a_i (m)	θ_i (deg) shown
θ_1	0	-90	0	-90
θ_2	0	90	a_2	0
θ_3	0	90	0	90
θ_4	0	-90	a_4	0
θ_5	0	90	0	0
θ_6	0	90	0	90
θ_7	d_7	0	0	0

Note: a_2 = 23.0 in., a_4= 20.0 in., d_7 = 12.0 in.

Table 2. Motion Range of the LTM
(Note: Zero reference is indicated in Figure 1)

Motion	Range (degrees) with counterbalancing and with cabling.	Range (degrees) without counterbalancing and with cabling
Shoulder pitch	$-45 > \theta_1 > -135$	$+30 > \theta_1 > -135$
Shoulder yaw	$+180 > \theta_2 > -180$	$+180 > \theta_2 > -180$
Elbow pitch	$+120 > \theta_3 > +45$	$+135 > \theta_3 > -30$
Elbow yaw	$+120 > \theta_4 > -120$	$+180 > \theta_4 > -180$
Wrist pitch	$+135 > \theta_5 > -30$	$+135 > \theta_5 > -30$
Wrist yaw	$+180 > \theta_6 > 0$	$+180 > \theta_6 > 0$
Wrist roll	$+180 > \theta_7 > -180$	$+180 > \theta_7 > -180$

interface boundaries of these joints provide inherent modularity. A wrist roll mechanism, mounted on the output of the wrist joint, provides the seventh degree of freedom. Seven degrees of freedom allow the LTM to reorient itself without changing the end-effector position and orientation. This paper describes the robotic control scheme for the LTM for utilizing redundancy to avoid internal singularities and joint limits.

2. Robotic Control of the LTM

Several joints of a robotic manipulator at time-varying rates simultaneously move the end-effector along a specified path defined in terms of end-effector position and orientation as a function of time. In the case of a six-degree-of-freedom manipulator, joint motions required to achieve a specified end-effector motion are unique. However, a seven-or more degree-of-freedom manipulator has more joints than the six independent variables required to completely specify the position and orientation of the end-effector. The kinematic linear equations relating unknown joint velocities to specified end-effector velocity components do not have unique solutions–an infinite number of solutions is possible. Thus we may choose a joint velocity solution that results in "improved" performance of the manipulator while tracking a desired trajectory, and performance may be judged by criteria such as avoiding obstacles or joint limits.

A number of control schemes for determining joint trajectories for redundant manipulators have been suggested by researchers. Most control schemes in the literature determine joint velocities through global or local optimization of various performance criteria. Global optimization schemes are generally iterative and computationally complex; thus, they are currently limited to off-line programming. On-line implementation is essential if the end-effector trajectory is continuously modified based on sensory feedback. Most local optimization schemes are presented in the framework of resolved rate control [2]. Control of the LTM in the robotic mode is achieved by an efficient gradient projection kinematic control scheme developed by Dubey et al. [3]. This scheme avoids computation of the pseudoinverse of the Jacobian, and it results in an efficient formulation for determining joint velocities.

3. Kinematic Optimization Scheme

The kinematic optimization scheme developed by Dubey et al. [3] used to control the LTM can be summarized as follows. A manipulator using n joints to control m independent variables of the end-effector position and orientation ($m \leq 6$) is described by the following kinematic equation:

$$\dot{x} = J\dot{\theta} \ , \tag{1}$$

where $\dot{x}$ is an m-dimensional vector of linear and angular velocities of the end-effector with reference to base coordinates, $\dot{\theta}$ is an n-dimensional vector of joint velocities, and J is an $m \times n$ Jacobian matrix.

If J is a square matrix and has full rank, then the joint velocities required to achieve the desired end-effector motion will be unique and can be evaluated by

$$\dot{\theta} = J^{-1}\dot{x} \ . \tag{2}$$

If J is rectangular with $m < n$, the joint velocities can be computed by

$$\dot{\theta} = J^{+}\dot{x} + (I - J^{+}J)\dot{\phi} \ , \tag{3}$$

where J^{+} is the Moore-Penrose generalized inverse [4] of the Jacobian. If J has a full rank , then

$$J^{+} = J^{T}(JJ^{T})^{-1}. \tag{4}$$

The matrix I in Eq. (3) is an $n \times n$ identity matrix, and the vector $\dot{\phi}$ is an arbitrary n-dimensional joint velocity vector. To optimize a performance criterion $H(\theta)$ using the gradient projection method [5], redundancy is resolved by substituting $k\nabla H(\theta)$ for $\dot{\phi}$ in Eq. (3) and rewriting it as

$$\dot{\theta} = J^{+}\dot{x} + k(I - J^{+}J)\nabla H(\theta). \tag{5}$$

The coefficient k in Eq. (5) is a real scalar constant, and $\nabla H(\theta)$ is the gradient vector of $H(\theta)$.

Let $\dot{\theta} \in R^{7}$ be the joint velocity vector for the seven-degree-of-freedom manipulator. Suppose in the Cartesian workspace the end-effector velocity is described by a six-dimensional vector with reference to the base coordinates, and has three linear and three angular velocity components. The joint velocity vector $\dot{\theta}$ and the end-effector velocity vector $\dot{x}$ are related by Eq. (1), where J is a 6×7 Jacobian matrix. We will assume that the rank of the Jacobian is six, which implies that J is not singular. Thus, it is possible to construct a nonsingular 6×6 matrix J^{*} from any six independent columns of the matrix. In general, by rearranging the columns of J and the corresponding elements of $\dot{\theta}$ in a different order, we can rewrite Eq. (1) as

$$\dot{x} = [\alpha \ J^{*}]\dot{\theta} \ , \tag{6}$$

where α is any column vector of the Jacobian such that the remaining six columns form a nonsingular matrix J^{*}. Rearranging terms in Eq. (5), we obtain the following:

$$\dot{\theta} = J^{+}(\dot{x} - kJ\nabla H) + k\nabla H \ . \tag{7}$$

A suitable selection of k may be based on the hardware bounds on the joint velocities and heuristics. The first term on the right-hand side of Eq. (7) is the least-norm solution of Eq. (1) with $\dot{x}$ replaced by $(\dot{x} - kJ\nabla H)$. As shown in ref.3, the least-norm solution can be obtained from a particular solution $\dot{\theta}_p^{*}$ and a homogeneous solution $\dot{\theta}_h^{*}$ of this equation by subtracting from the particular solution its component along the homogeneous solution. Thus we have

$$\dot{\theta} = \dot{\theta}_p^{*} - \left(\frac{\dot{\theta}_p^{*T} \ \dot{\theta}_h^{*}}{\dot{\theta}_h^{*T} \ \dot{\theta}_h^{*}} \right) \dot{\theta}_h^{*} + k\nabla H \ , \tag{8}$$

where

$$\dot{\theta}_p^* = \begin{bmatrix} 0 \\ J^{*-1}(\dot{x} - kJ\nabla H) \end{bmatrix}, \tag{9}$$

and

$$\dot{\theta}_h^* = \begin{bmatrix} 1 \\ J^{*-1}\alpha \end{bmatrix}. \tag{10}$$

If we assume the wrist to be spherical, with none of the two wrist axes pairs aligned, we can partition J^* as follows:

$$J^* = \begin{bmatrix} J_1^{*3x3} & 0^{3x3} \\ J_2^{*3x3} & J_3^{*3x3} \end{bmatrix}. \tag{11}$$

To determine $J^{*-1}(\dot{x} - kJ\nabla H)$ and $J^{*-1}\alpha$ in Eqs. (9) and (10) respectively, we need only to solve two sets of three simultaneous equations. Thus a simple formulation for determining the joint velocities is obtained.

4. Inverse Kinematics of the LTM

The above control scheme for optimizing a performance criterion using the gradient projection method was applied to the LTM. The LTM is a seven degree-of-freedom manipulator with a spherical wrist. The pitch-yaw-roll spherical wrist is designed so that its singularities occur when the hand is pointing to its sides and at the extremes of motion range, not when it is pointing straight out, as is common in many industrial manipulators. Degrees of freedom of the LTM and the coordinate frames referred to in the Denavit-Hartenberg table (Table 1) for this manipulator are shown in Fig.1.

To simplify the calculations, we will refer the desired end-effector and wrist velocity vectors to the third coordinate frame x_3, y_3, z_3, which is attached to link 3 using the notation used by Paul [6]. This results in a Jacobian that has a much simpler form and thus is more efficient for computation. Let the desired end-effector velocity referred to the third coordinate frame be given by

$$^3\dot{x}_h = [\,^3v_h^T \quad ^3\omega_h^T\,]^T$$

$$^3\dot{x}_h = [^3v_{h_1}\ ^3v_{h_2}\ ^3v_{h_3}\ ^3\omega_{h_1}\ ^3\omega_{h_2}\ ^3\omega_{h_3}]^T, \tag{12}$$

where $^3v_h \in R^3$ and $^3\omega_h \in R^3$ are the linear and angular hand velocity vectors, respectively, referred to the third coordinate frame.

Let the desired wrist velocity vector referred to the third coordinate frame be given by

$$^3\dot{x}_w = [\,^3v_w^T \quad ^3\omega_w^T\,]^T$$

$$^3\dot{x}_w = [^3v_{w_1}\ ^3v_{w_2}\ ^3v_{w_3}\ ^3\omega_{w_1}\ ^3\omega_{w_2}\ ^3\omega_{w_3}]^T, \tag{13}$$

where $^3v_w \in R^3$ and $^3\omega_w \in R^3$ are the linear and angular wrist velocity vectors, respectively, referred to the third coordinate frame. For a given $^3\dot{x}_h$, the terms of $^3\dot{x}_w$ may be obtained by using the following relationships:

$$^3\omega_w = {}^3\omega_h\,, \tag{14}$$

$$ {}^3v_w = {}^3v_h - {}^3\omega_h \times d_7\, {}^3z_h , \tag{15}$$

where ${}^3z_h$ is the unit vector z_h at the hand (Fig. 1) that is referred to the third coordinate frame; ${}^3z_h$ may be shown to be the following:

$$ {}^3z_h = \left[s_4c_6+c_4c_5s_6 \quad s_4c_5s_6-c_4c_6 \quad -s_5s_6 \right]^T , \tag{16}$$

where c_i and s_i represent $\cos\theta_i$ and $\sin\theta_i$ respectively. Let ${}^3J_w$ be the Jacobian relating the joint velocity vector $\dot\theta = [\dot\theta_1, \dot\theta_2, ..., \dot\theta_7]^T$ and the wrist velocity vector ${}^3\dot{x}_w$ such that

$$ {}^3\dot{x}_w = {}^3J_w\dot\theta . \tag{17}$$

The Jacobian ${}^3J_w$ can be shown to be the following:

$$ {}^3J_w = \begin{bmatrix} a_4s_2s_3s_4+a_2c_2s_3 & a_4c_3s_4 & 0 & -a_4s_4 & 0 & 0 & 0 \\ -a_4s_2s_3c_4 & -a_4c_3c_4-a_2 & 0 & a_4c_4 & 0 & 0 & 0 \\ -a_4(s_2c_3s_4+c_2c_4)-a_2c_2c_3 & a_4s_3s_4 & -a_4c_4 & 0 & 0 & 0 & 0 \\ -s_2c_3 & s_3 & 0 & 0 & -s_4 & c_4s_5 & c_4c_5s_6+s_4c_6 \\ c_2 & 0 & 1 & 0 & c_4 & s_4s_5 & s_4c_5s_6-c_4c_6 \\ -s_2s_3 & -c_3 & 0 & 1 & 0 & c_5 & -s_5s_6 \end{bmatrix}, \tag{18}$$

where, as before, c_i and s_i represent $\cos\theta_i$ and $\sin\theta_i$ respectively.

To determine the joint velocities required to follow a desired end-effector velocity and to optimize a given performance criterion using the gradient projection method, we first determine the end-effector velocity ${}^3\dot{x}_h$ referred to the third coordinate frame. Given the end-effector velocity vector $\dot{x}_h \in R^6$ in the base coordinates, we can determine ${}^3\dot{x}_h$ from the following:

$$ {}^3\dot{x}_h = {}^3R_0\dot{x}_h , \tag{19}$$

where ${}^3R_0$ is a 3×3 projection matrix given by

$$ {}^3R_0 = \begin{bmatrix} c_1c_2c_3-s_1s_3 & s_1c_2c_3+c_1s_3 & -s_2c_3 \\ c_1s_2 & s_1s_2 & c_2 \\ c_1c_2s_3+s_1c_3 & s_1c_2s_3-c_1c_3 & -s_2s_3 \end{bmatrix}, \tag{20}$$

and c_i and s_i represent $\cos\theta_i$ and $\sin\theta_i$ respectively. We can now determine wrist velocity vector ${}^3\dot{x}_w$ from Eqs. (14) and (15).

Consider the case when the first column of the Jacobian is taken to be α and the remaining six columns are independent and form the matrix J^*. The elements of $\dot\theta^*_p$ denoted by $\dot\theta^*_{p_i}$, for i = 1 to 7, with $\dot\theta^*_{p_1} = 0$, may be obtained as follows:

$$ \begin{bmatrix} \dot\theta^*_{p_2} \\ \dot\theta^*_{p_4} \end{bmatrix} = \frac{1}{\Delta} \begin{bmatrix} a_4c_4 & a_4s_4 \\ a_2+a_4c_3c_4 & a_4c_3s_4 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}, \tag{21a}$$

where $\Delta = -a_2 a_4 s_4$,

$$\dot{\theta}^*_{p3} = \frac{(a_4 s_3 s_4)\dot{\theta}^*_{p2} - \dot{x}_3}{a_4 c_4}, \tag{22a}$$

$$\begin{bmatrix} \dot{\theta}^*_{p5} \\ \dot{\theta}^*_{p6} \\ \dot{\theta}^*_{p7} \end{bmatrix} = \frac{1}{s_6} \begin{bmatrix} c_4c_5c_6 - s_4s_6 & s_4c_5c_6 + c_4s_6 & -s_5c_6 \\ c_4s_5s_6 & s_4s_5s_6 & c_5s_6 \\ c_4c_5 & s_4c_5 & -s_5 \end{bmatrix} \begin{bmatrix} \dot{x}_4 - s_3\dot{\theta}^*_{p2} \\ \dot{x}_5 - \dot{\theta}^*_{p3} \\ \dot{x}_6 + c_3\dot{\theta}^*_{p2} - \dot{\theta}^*_{p4} \end{bmatrix}. \tag{23a}$$

On the other hand, another column, say column four, may be taken to be α with the remaining six columns forming the matrix $\mathbf{J}^*$, in which case if $\mathbf{J}^*$ is nonsingular we have $\dot{\theta}^*_{p4} = 0$ and :

$$\begin{bmatrix} \dot{\theta}^*_{p1} \\ \dot{\theta}^*_{p2} \end{bmatrix} = \frac{1}{\Delta} \begin{bmatrix} -(a_2 + a_4c_3c_4) & -a_4c_3s_4 \\ a_4s_2s_3c_4 & a_4s_2s_3s_4 + a_2c_2s_3 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}, \tag{21b}$$

where $\Delta = -a_2s_3(a_4c_2c_3c_4 + a_4s_2s_4 + a_2c_2)$,

$$\dot{\theta}^*_{p3} = \frac{(a_4s_3s_4)\dot{\theta}^*_{p2} - \dot{x}_3 - [a_4(s_2c_3s_4 + c_2c_4) + a_2c_2c_3]\dot{\theta}^*_{p1}}{a_4c_4}, \tag{22b}$$

and

$$\begin{bmatrix} \dot{\theta}^*_{p5} \\ \dot{\theta}^*_{p6} \\ \dot{\theta}^*_{p7} \end{bmatrix} = \frac{1}{s_6} \begin{bmatrix} c_4c_5c_6 - s_4s_6 & s_4c_5c_6 + c_4s_6 & -s_5c_6 \\ c_4s_5s_6 & s_4s_5s_6 & c_5s_6 \\ c_4c_5 & s_4c_5 & -s_5 \end{bmatrix} \begin{bmatrix} \dot{x}_4 + s_2c_3\dot{\theta}^*_{p1} - s_3\dot{\theta}^*_{p2} \\ \dot{x}_5 - c_2\dot{\theta}^*_{p1} - \dot{\theta}^*_{p3} \\ \dot{x}_6 + s_2s_3\dot{\theta}^*_{p1} + c_3\dot{\theta}^*_{p2} \end{bmatrix}. \tag{23b}$$

In Eqs. (21) through (23), $\dot{x}_i$ for i = 1 to 6 are the ith elements of the vector $(\dot{\mathbf{x}} - k\mathbf{J}\nabla H)$. The vector $\dot{\theta}^*_h$ with elements $\dot{\theta}^*_{h\,i}$, i = 1 to 7, may similarly be obtained by setting the element $\dot{\theta}^*_{h\,i}$ = 1, where i = 1 for the case when α is taken to be the first column and i = 4 when α is taken to be the fourth column. The remaining elements of $\dot{\theta}^*_h$ may be obtained from Eqs. (21) through (23) by replacing $\dot{\theta}^*_{p\,i}$ by $\dot{\theta}^*_{h\,i}$ and using the ith elements of vector $-\alpha$ for $\dot{x}_i$ for i = 1 to 6.

We have obtained computationally efficient closed-form solutions for the elements of $\dot{\theta}^*_p$ and $\dot{\theta}^*_h$ in Eqs. (21) through (23). The vector $\dot{\theta}$ can now be obtained from Eq. (8). Thus, by this approach we have eliminated the need to determine the generalized inverse of the Jacobian or to numerically solve the six simultaneous equations with six unknowns. Suppose that in the course of execution we have s_4 approaching zero, then α may be chosen to be equal to column 4. On the other hand, if s_3 approaches zero we may choose α to be column 1. In all cases, $c_4 = 0$ and/or $s_6 = 0$ must be avoided, situations achieved by optimizing a suitable performance criterion.

5. Performance Criteria and Singularity Analysis

To avoid joint limits and singular configurations of the LTM we have developed performance criteria to be optimized using the control scheme presented in the last section. The motion ranges of the joints of the LTM, given in Table 2, are given for two cases: (1) with counter-balancing and cabling and (2) without counter-balancing and with cabling (see Fig.1). To stay within the joint limits specified in Table 2, we may choose the following performance criterion:

$$H(\theta) = \Sigma\,(\theta_i - \theta_{imid})^2 \;, \tag{24}$$

where θ_i is the ith joint angle and θ_{imid} is the midpoint value for the θ_i joint angle. Inspection of the above performance criterion shows that if it is minimized, the joint angles tend to stay in their midrange. Each term in the above summation may be weighted according to the range of the corresponding joint motion and its distance from the midpoint.

When a manipulator is in a singular configuration it is unable to move or rotate the end-effector in at least one direction. The joint velocities required to move in this direction are infinitely high. In a configuration close to a singular configuration joint velocities required to move in certain direction(s) are much above the hardware bounds on joint velocities, resulting in inaccurate motion. The workspace of an articulated manipulator (redundant or nonredundant) is filled with singularities at the workspace boundaries as well as inside the workspace. Singularities at the workspace boundaries are usually unavoidable; however, singularities inside the workspace, referred to as internal singularities, are avoidable for manipulators with redundant joints. Because an infinite number of joint configurations results in a given position and orientation of a redundant manipulator within its workspace, it is possible to choose a nonsingular joint configuration. However, in order to avoid singular configurations we need to know the conditions for singularity.

When a manipulator is in a singular configuration, the determinant of JJ^T is zero [7]. Thus the joint coordinates that make the determinant of JJ^T equal to zero would result in singular configurations. However, this condition involves an extremely complicated equation which is difficult to solve. The singularities of a seven-degree-of-freedom arm such as the LTM occur when the rank of the Jacobian J is less than six, implying that the arm is in a configuration in which the end-effector cannot be moved and rotated in a completely arbitrary direction. Because the Jacobian for the LTM is a 6 × 7 matrix, the rank of it may be determined by considering the determinant of all seven 6 × 6 matrices that can be formed from the columns of J. Since this determination is extremely laborious, we utilize the following scheme.

When joint 6 is either 0 or 180°, then joint 5 is collinear with joint 7. In this case the column 5 vector of J is equal to or the negative of the column 7 vector, which implies that the wrist is singular and cannot be moved to an arbitrary new orientation with just the use of the wrist joint angles. Thus, the task of finding the singularities has been broken into two parts. The singularities will be considered first for the case when joints 5 and 7 are not collinear, and then for the case when joints 5 and 7 are collinear. The Jacobian referred to the second coordinate frame ^2J contains the following components:

$$^2J = \begin{bmatrix} -a_4c_2s_3c_4 & a_4s_4 & -a_4s_3c_4 & -a_4c_3s_4 & 0 & 0 & 0 \\ a_4(s_2s_4+c_2c_3c_4)+a_2c_2 & 0 & a_4c_3c_4 & -a_4s_3s_4 & 0 & 0 & 0 \\ -a_4s_2s_3c_4 & -(a_2+a_4c_3c_4) & 0 & a_4c_4 & 0 & 0 & 0 \\ -s_2 & 0 & 0 & s_3 & -c_3s_4 & c_3c_4s_5+s_3c_5 & (c_3c_4c_5-s_3s_5)s_6+c_3s_4c_6 \\ 0 & 1 & 0 & -c_3 & -s_3s_4 & s_3c_4s_5-c_3c_5 & (s_3c_4c_5+c_3s_5)s_6+s_3s_4c_6 \\ c_2 & 0 & 1 & 0 & c_4 & s_4s_5 & s_4c_5s_6-c_4c_6 \end{bmatrix} \tag{25}$$

We now consider the following the cases:

1. Joints 5 and 7 Are Not Collinear (Nonsingular Wrist)

Since the wrist is nonsingular, the LTM can be singular only when the first three rows of the Jacobian have a rank less than 3. This implies that the top left 3×4 submatrix Js of ${}^2\mathbf{J}$ will have a rank less than 3 in a singular configuration. This sub-Jacobian is given by,

$$\mathrm{Js} = \begin{bmatrix} -a_4c_2s_3c_4 & a_4s_4 & -a_4s_3c_4 & -a_4c_3s_4 \\ a_4(s_2s_4+c_2c_3c_4)+a_2c_2 & 0 & a_4c_3c_4 & -a_4s_3s_4 \\ -a_4s_2s_3c_4 & -(a_2+a_4c_3c_4) & 0 & a_4c_4 \end{bmatrix}. \tag{26}$$

Denote by J_{ijk} the Jacobian formed by the ith, jth, and kth columns of the Jacobian represented by Js. If the rank of Js is less than 3, then it follows that,

$$\det(J_{123}) = 0, \quad \det(J_{124}) = 0, \quad \det(J_{134}) = 0, \quad \det(J_{234}) = 0. \tag{27}$$

Based on the above conditions we obtain the singularities corresponding to a nonsingular wrist (see Table 3). Three singular configurations corresponding to a nonsingular wrist are shown in Fig. 2.

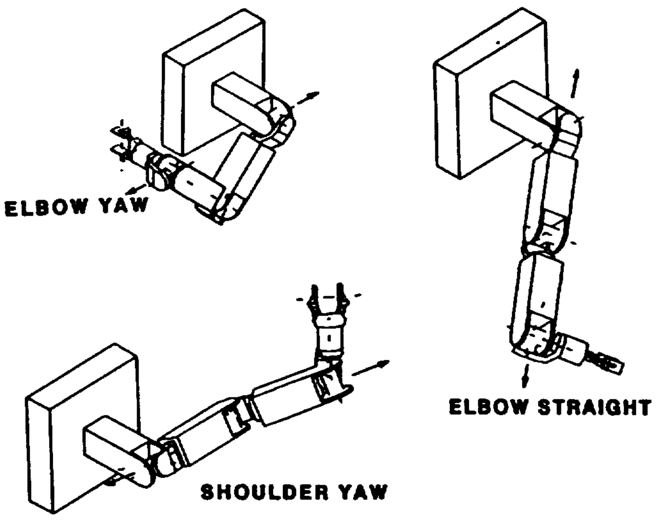


Table 3. Singularities of the LTM (Nonsingular Wrist)

θ_2	θ_3	θ_4	
	90	±90	
	–90	+90	NP2
w1		90	
w2		–90	
	0	0	
	180	0	NP1
	0	180	NP1
	180	180	
±90		0	
±90		180	

$w1 = \mathrm{atan}(-a_2/a_4)$, $w2 = \mathrm{atan}(a_2/a_4)$

NP1 -- Not possible. Link 2 is coincident with link 4.

NP2 -- Not possible. Wrist pitch, θ_3, is greater than –90.

Fig. 2 Singular configurations corresponding to a nonsingular wrist.

2. Joints 5 and 7 Are Collinear (Singular Wrist)

Notice that with joints 5 and 7 collinear ($\theta_6 = \pm 90°$), the two 6-dimensional vectors formed by columns 5 and 7 of Jacobian ${}^2\mathbf{J}$ are parallel, which corresponds to the wrist being in a singular position. Hence, for the LTM to have a singularity, only the sub-Jacobian given by the first six columns of ${}^2\mathbf{J}$ need be considered. This sub-Jacobian will be represented by Jss and can be written in the following form:

$$\mathrm{Jss} = \begin{bmatrix} -a_4c_2s_3c_4 & a_4s_4 & -a_4s_3c_4 & -a_4c_3s_4 & 0 & 0 \\ a_4(s_2s_4+c_2c_3c_4)+a_2c_2 & 0 & a_4c_3c_4 & -a_4s_3s_4 & 0 & 0 \\ -a_4s_2s_3c_4 & -(a_2+a_4c_3c_4) & 0 & a_4c_4 & 0 & 0 \\ -s_2 & 0 & 0 & s_3 & -c_3s_4 & c_3c_4s_5+s_3c_5 \\ 0 & 1 & 0 & -c_3 & -s_3s_4 & s_3c_4s_5-c_3c_5 \\ c_2 & 0 & 1 & 0 & c_4 & s_4s_5 \end{bmatrix}. \tag{28}$$

For the case under consideration, all the remaining singularities for the LTM can be determined by setting the determinant of the Jacobian Jss equal to zero. However, this leads to an equation for which the roots are difficult to determine. As a result, at this time only the singularities will be found that correspond to the occurrence of two sets of collinear joint axes. Since all singularities were found for which joints 5 and 7 were not collinear, then it follows that any singularities that occur due to two sets of collinear joint axes must have one of that set of collinear joint axes due to joints 5 and 7 being collinear. Thus to determine if two sets of collinear joint axes can occur, it is first necessary to find all possible cases where sets of joint axes are collinear. Then it must be determined if those sets can physically exist when joints 5 and 7 are collinear.

In the procedure for doing this we used the notation of Paul [6] in which the mth joint axis lies along the z axis of the (m-1) frame. Thus, if the z axis of frame n is collinear with the z axis of frame m, it means that joint axes m + 1 and n + 1 are collinear. Let ${}^{m}T^{n}$ denote the homogeneous transform which refers frame n of the LTM to frame m of the LTM. To be collinear it follows that ${}^{m}T^{n}$ must be of the form

$$ {}^{m}T^{n} = \begin{bmatrix} r11 & r12 & 0 & 0 \\ r21 & r22 & 0 & 0 \\ 0 & 0 & \pm 1 & p3 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{29} $$

In general, the ${}^{m}T^{n}$ is of the form,

$$ {}^{m}T^{n} = \begin{bmatrix} r11 & r12 & r13 & p1 \\ r21 & r22 & r23 & p2 \\ r31 & r32 & \rho 33 & p3 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{30} $$

Thus, for two joint axes to be collinear, the following seven relations must be satisfied.

$$ \begin{array}{llll} \text{(a): } r13 = 0, & \text{(c): } r31 = 0, & \text{(e): } r33 = \pm 1, & \\ \text{(b): } r23 = 0, & \text{(d): } r32 = 0, & \text{(f): } p1 = 0, & \text{(g): } p2 = 0. \end{array} \tag{31} $$

Notice that Eq. (31) represents a set of dependent equations. Only Eqs. (31c), (31d), (31f), and (31g) are independent; however, the other equations help in the algebraic manipulation. Based on the above conditions, we obtain the singularities of the LTM when the wrist is in a singular configuration (Table 4). Figure 3 shows three of the singular configurations corresponding to a singular wrist. Including the singularities of both cases, that is, singular and nonsingular wrist, we obtain the singularities of the LTM shown in Table 5.

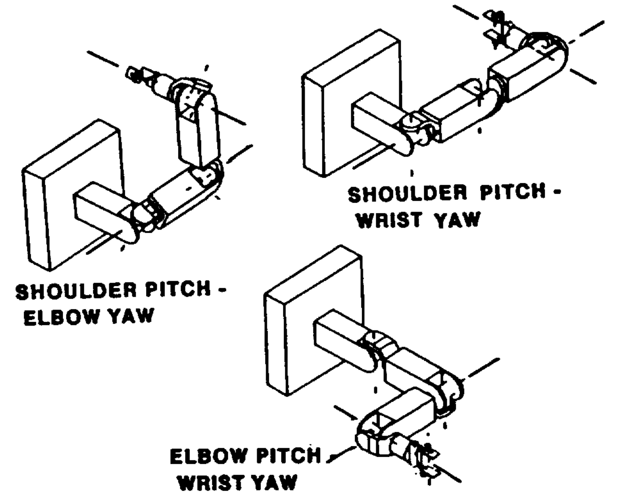


Fig. 3 Three configurations corresponding to a singular wrist.

Table 4. Singularities of the LTM (Singular Wrist)

θ_2	θ_3	θ_4	θ_5	θ_6
±90	90			0,180
		±90	90	0,180

Table 5. Singularities of the LTM (Singular and Nonsingular Wrist)

θ_2	θ_3	θ_4	θ_5	θ_6
	90	±90		
w1		90		
w2		-90		
	0	0		
	180	180		
±90		0		
±90		180		
±90	90			0,180
		±90	90	0,180

$w1 = \text{atan}(-a_2/a_4)$, $w2 = \text{atan}(a_2/a_4)$

To avoid the singular configurations shown in Table 5, we may optimize the following performance criterion:

$$H(\theta) = k_2\sin^2\theta_2 + k_3\cos^2\theta_3 + k_4\sin^2\theta_4 + k_6\cos^2\theta_6\,, \qquad (32)$$

where the k_i are weighting factors that may be chosen based on the range of the joint angle and the distance the joint angle is from its desired value. Inspection of Eq. (32) and Table 5 shows that if $H(\theta)$ is minimized, then joints 4 and 2 will tend to move toward 0° while joints 3 and 6 will tend to move toward ±90°.

6. Real-Time Implementation and Simulation Results

The robotic control scheme for the LTM is being implemented on a 16-MHz VME bus/Motorola 68020 microprocessor. The software is written in C language. Figure 4 is a block diagram of implementation of the kinematic optimization scheme. Control loops are closed around the joint servos, and the optimizing inverse kinematics algorithm is implemented in an open-loop fashion. The complete algorithm runs at 100 Hz, which includes the optimizing inverse kinematics, forward kinematics, joint-to-motor transformation calculations, and joint velocity and position limit checking.

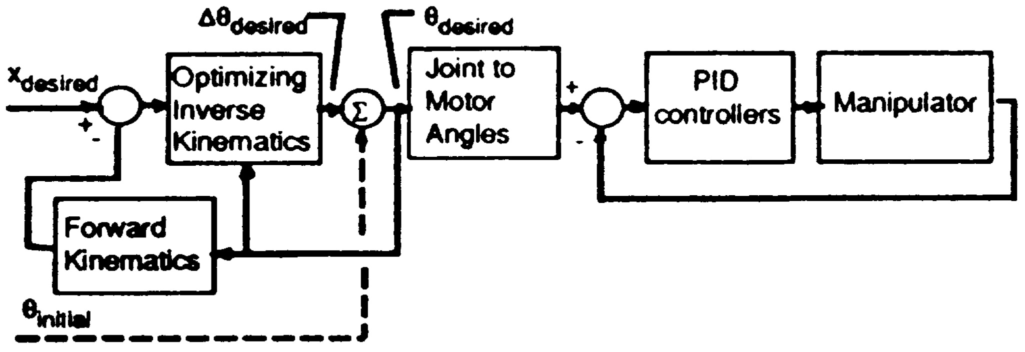


Fig. 4 Real-time implementation block diagram.

Graphical simulations to test the control software were performed for different cases. In each case the LTM end-effector follows a specified straight-line trajectory while maintaining a fixed orientation . In the first case, the performance criterion $H(\theta)$ in Eq. 24 is minimized so that the LTM avoids running into joint limits. In the second case, singularities are avoided by optimizing performance criterion $H(\theta)$ in Eq. 32. Comparison is made in each case with the joint trajectories resulting from the least-norm solution, which does not utilize the null space of the Jacobian to optimize a performance criterion.

Simulation results presented in Figures 5 and 6 are for the case when the performance criterion is optimized to avoid joint angle limits. Each figure shows the LTM trajectory along with a plot of joint angles as a function of time. In Fig. 5, only the least-norm solution is used to follow the desired end-effector trajectory. In this case joint 5 hits its lower limit before the end point is reached. Figure 6 demonstrates the use of redundancy to avoid the limits on joint angles. In this case the LTM reaches the desired end point along a specified trajectory without reaching a joint limit.

Figures 7 and 8 present the case when the performance criterion is optimized to avoid high joint velocities due to an internal singularity. The LTM is started in a near singular configuration in both figures. The least-norm solution shown in Fig. 7 produces high joint velocities and, therefore, the LTM is commanded to stop; however, Fig. 8 shows that if the null space of the Jacobian is utilized to avoid singularities, the desired end point is reached without getting too close to the singular configuration, thus avoiding high joint velocities.

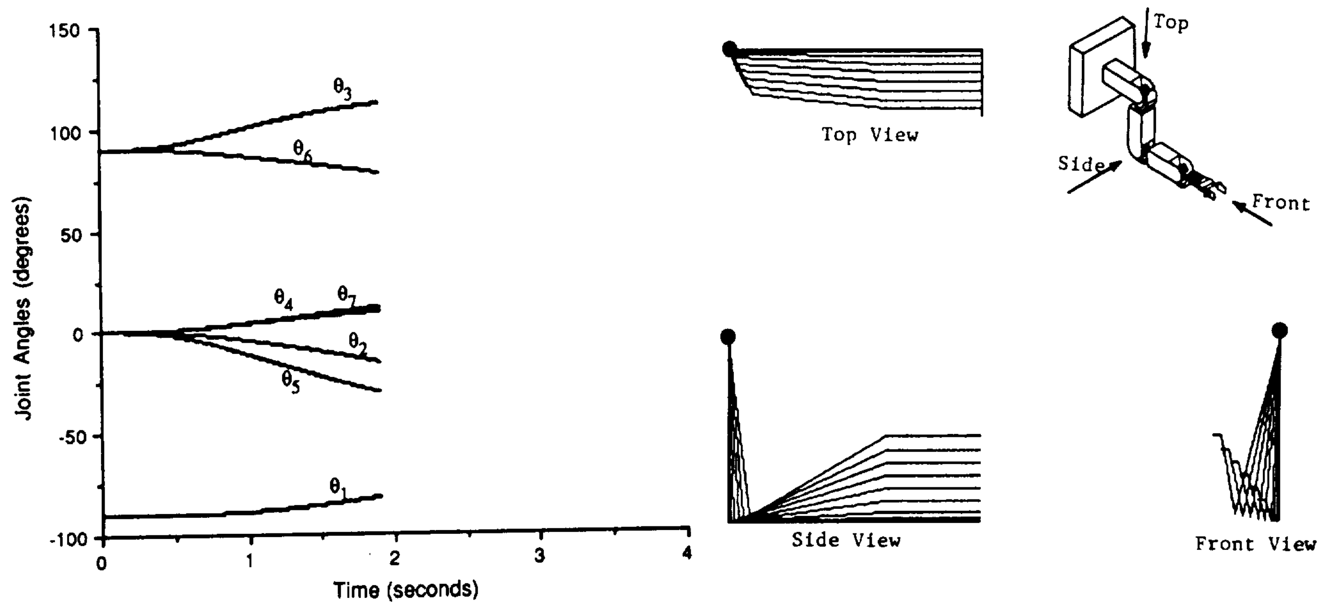


Fig. 5 End-effector motion along a straight line with a fixed orientation . (Least-norm solution - reaches joint 5 limit)

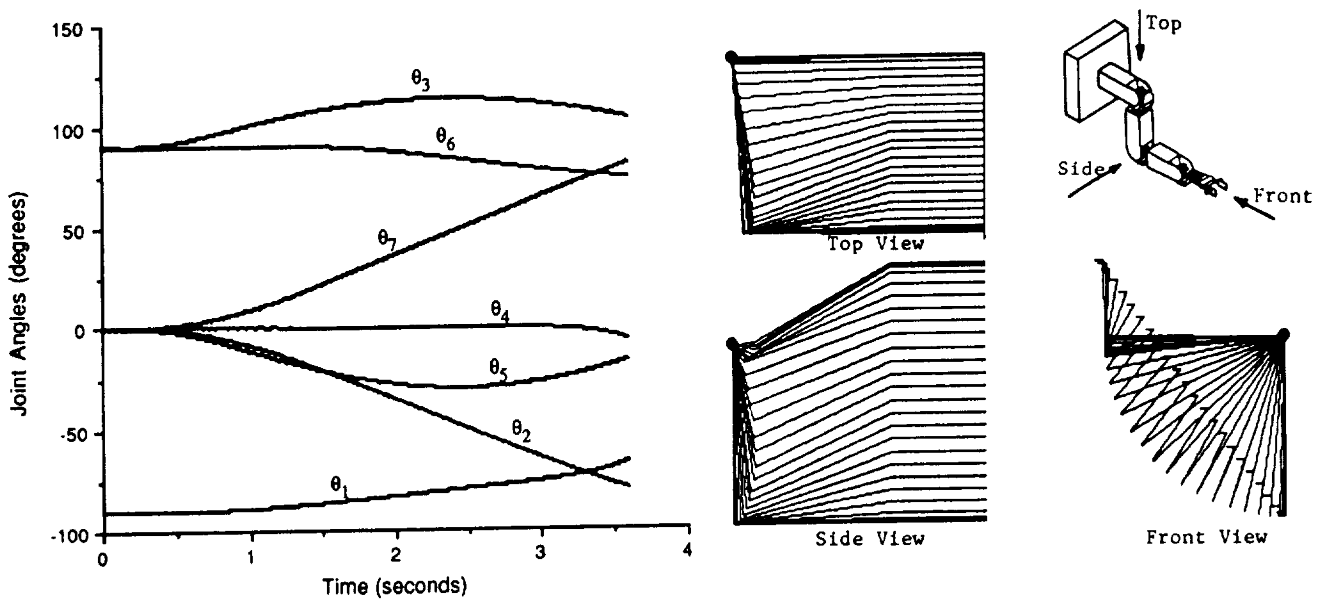


Fig. 6 End-effector motion along a straight line with fixed orientation. (Redundancy utilized to avoid joint limits)

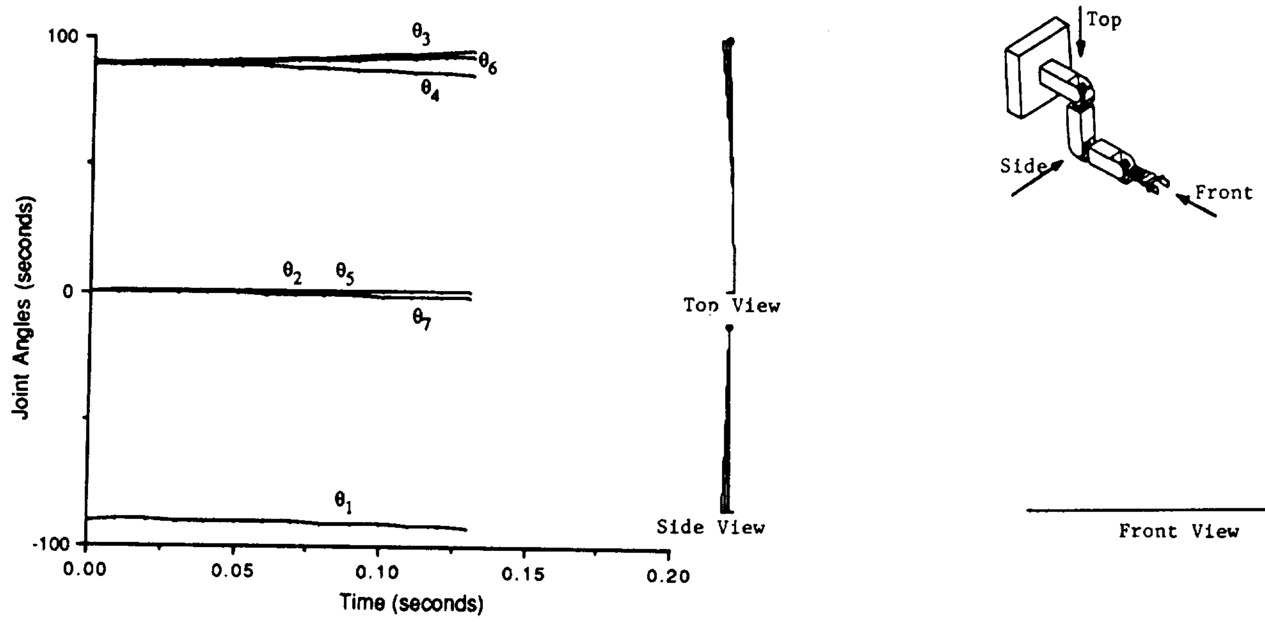


Fig. 7 End-effector motion along a straight line with a fixed orientation .
(Least-norm solution-reaches velocity limit)

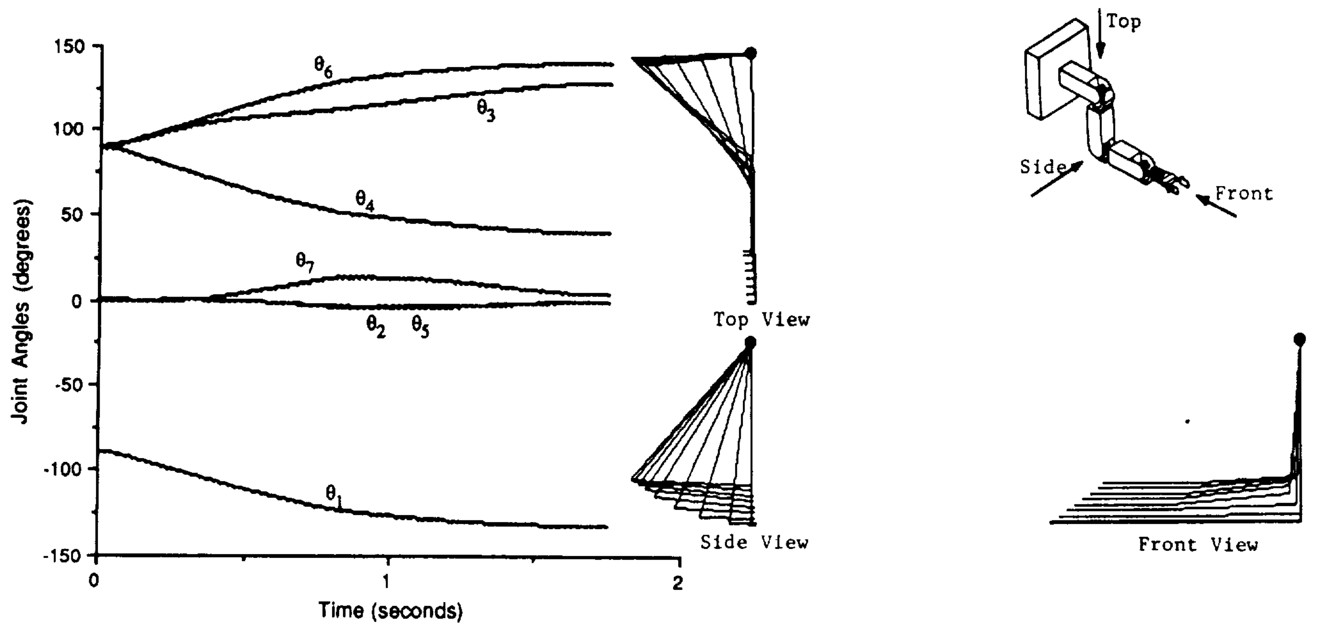


Fig. 8 End-effector motion along a straight line with a fixed orientation .
(Redundancy utilized to avoid singularities)

7. Conclusions

A computationally efficient, robotic control scheme for the seven-degree-of-freedom LTM was presented. This scheme determines the joint velocities required to follow a specified end-effector trajectory while optimizing a given performance criterion using the gradient projection method. LTM kinematics was analyzed to determine its internal singularities. Performance criteria to avoid joint angle limits and singularities were obtained. Feasibility and effectiveness of the control scheme were demonstrated by simulations to avoid joint angle limits and singularities. Real-time implementation of the control scheme is in progress. Future work includes the use of redundancy to avoid obstacles and minimize joint torques, simultaneous optimizations of multiple performance criteria, and extension of the robotic control scheme to telerobotic control with force reflection.

References

[1] Herndon, J.N., Babcock, S.M., Butler, P.L., Costello, H.M., Glassell, R.L., Kress, R.L., Kuban, D.P., Rowe, J.C., Williams, D.M., "The Laboratory Telerobotic Manipulator Program," Proceedings of the NASA Conference on Space Telerobotics, January 31-February 2, 1989, Pasadena, California.

[2] Whitney, D. E., "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators," ASME J. Dynamic Systems, Measurement, and Control, 94, No. 4, 303-309 (1972)

[3] Dubey, R.V., Euler, J.A., and Babcock, S.M., "An Efficient Gradient Projection Optimization Scheme for a Seven-Degree-of-Freedom Redundant Robot with Spherical Wrist," Proc. of the IEEE International Conference on Robotics and Automation Philadelphia, April 1988, pp. 28-36.

[4] Albert, A., Regression and the Moore-Penrose Pseudo-Inverse, Academic Press, 1972.

[5] Liegeois, A., "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms," IEEE Trans. Systems, Man, Cybern., SMC-7, No. 12 (1977)

[6] Paul, R.P., Robot Manipulators: Mathematics, Programming and Control, The MIT Press, 1981.

[7] Yoshikawa, T., "Manipulability of Robotic Mechanisms," The Int. J. of Rob. Research," Vol. 4, No. 2, Summer 1985, 3-9.

N90-29871

The Control of Space Manipulators Subject to Spacecraft Attitude Control Saturation Limits

S. Dubowsky, E. E. Vance[*], M. A. Torres[**]

Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

The motions of robotic manipulators mounted on spacecraft can disturb the spacecraft's positions and attitude. These disturbances can surpass the ability of the system's attitude control reaction jets to control them, for the disturbances increase as manipulator speeds increase. If the manipulator moves too quickly the resulting disturbances can exceed the saturation levels of the reaction jets, causing excessive spacecraft motions. This paper presents a method for planning space manipulator,s motions so that tasks can be performed as quickly as possible without saturating the system's attitude control jets.

1. Introduction

This paper presents a method that enables space manipulator motions to be planned so that tasks can be performed in minimum time, without saturating the system's attitude control jets.

Future space missions are expected to use robotic manipulators mounted on spacecraft to construct space stations and repair satellites. However, the motions of such manipulators can disturb the position and attitude of their spacecraft. While control techniques have been proposed for space manipulators which permit their spacecraft to move in response to manipulator motions [1], for many missions even relatively small unplanned spacecraft motions may be undesirable. Although these motions can be controlled using the spacecraft's attitude control reaction jets, disturbances increase as manipulator speeds increase. Even with the use of recently developed methods for planning space manipulator motions to minimize the disturbances [2,3], these disturbances can exceed the saturation levels of the spacecraft reaction jet system and result in excessive spacecraft motions. Therefore, motion planning for space manipulators must consider the limits of the reaction jets if space manipulators are to be able to perform their tasks quickly, in minimum time, without excessive spacecraft motions.

A number of methods have been developed to plan the minimum time motion of *fixed based* manipulators [4-9]. However these methods do not consider the problem of the motion planning for *space manipulators* with their spacecraft control systems saturation constraints . The technique presented here considers the saturation limits of both a manipulator's joint actuators and those of the reaction jets. It can be applied to any rigid, non-redundant space manipulator system whose actuator and spacecraft capabilities can be specified as a function of the state of the system. It provides both the optimal position and velocity command profiles for the system and the optimal open-loop actuator

[*] Now at the Center for Naval Analyses, Alexandria, VA.

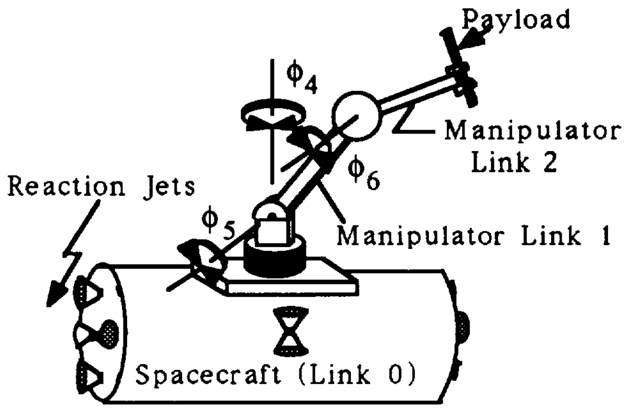
[**] NASA Fellow.

and reaction jet forces and torques required for a given task. These forces and torques can be used as feedforward signals by manipulator and spacecraft closedloop control systems to reduce dynamic control system errors. The technique can also be used with conventional planning methods to aide in planning nonoptimal manipulator motions that will not exceed the system's capabilities.

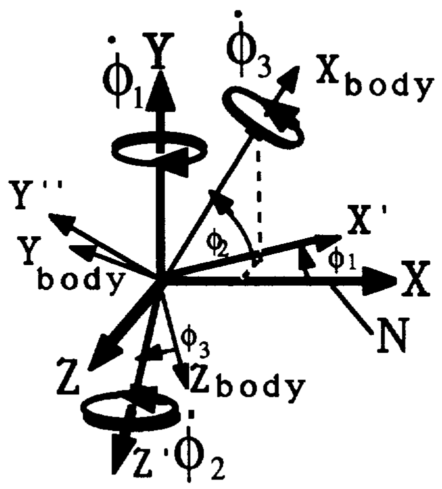
Results demonstrate the effectiveness of the method for planning minimum time space manipulator motions. They also show that the technique can be used to design the lightest weight system to perform a given set of tasks in a specified amount of time.

2. The System

The technique is illustrated by its application to a simple system consisting of a three degree-of-freedom (DOF) revolute manipulator mounted on a spacecraft with six DOF, see Figure 1. The system has a total of nine DOF. The spacecraft is equipped with six reaction jets which can counteract the disturbance forces and moments generated by the manipulator's motion.



a. System and Manipulator Joint Variables.



b. Spacecraft Rotation Variables.

Figure 1. System Model and Variables

The spacecraft's six DOF are represented by the variables X, Y, Z, ϕ_1, ϕ_2 and ϕ_3 which define its position and orientation with respect to the inertial coordinate frame N, also shown in Figure 1. A body-fixed coordinate frame (X_{body},Y_{body},Z_{body}) is attached to the spacecraft at its center of mass. The angle ϕ_1 is the rotation of the spacecraft about the Y_{body} axis, ϕ_2 is the rotation about the Z_{body} axis, and ϕ_3 is the rotation about the X_{body} axis, as shown in Figure 1b. The three manipulator joint motions, ϕ_4, ϕ_5 and ϕ_6, are shown in Figure 1a.

3. The Dynamic Model

The planning algorithm requires a full nonlinear dynamic model of the system. These equations may be formulated in any convenient manner. Here a Lagrangian formulation was used to develop the the dynamic equations for the system shown in Figure 1 with X, Y, Z, ϕ_1, ϕ_2, ϕ_3, ϕ_4, ϕ_5 and ϕ_6 as generalized coordinates. The resulting dynamic equations may be written in the vector form [10]:

$$\mathbf{M}_{\phi}\ddot{\underline{\phi}} + \dot{\underline{\phi}}\ \overset{<}{\mathbf{C}}_{\phi}\ \dot{\underline{\phi}} = \underline{\tau} \tag{1}$$

where
$\underline{\phi}$ is a 9 element vector of generalized coordinates,
$\mathbf{M}_{\phi}$ is a 9x9 mass matrix,
$\overset{<}{\mathbf{C}}_{\phi}$ is a 9x9x9 Coriolis tensor, and
$\underline{\tau}$ is a 9 element vector of generalized forces and moments.

The elements of the Coriolis tensor, can be calculated from:

$$\overset{<}{C}_{\phi ijk} = \frac{\partial M_{\phi ij}}{\partial \phi_k} - \frac{1}{2}\frac{\partial M_{\phi kj}}{\partial \phi_i} \tag{2}$$

This nonlinear matrix equation was used in an independent dynamic simulation of the system to verify the results of the planning algorithm. When the objective of the optimization is to maintain a stationary spacecraft, a simplified form of Equation (1) can be used in the algorithm. It is obtained by setting the time derivatives of the spacecraft's generalized coordinates to zero. This simplification must be done after the complete equations of motion have been derived; setting these variables equal to zero before the Lagrangian differentiation leads to errors [10]. The resulting simplified equations have the form:

$$\begin{bmatrix} m_{17} & m_{18} & m_{19} \\ m_{27} & m_{28} & m_{29} \\ m_{37} & m_{38} & m_{39} \\ m_{47} & m_{48} & m_{49} \\ m_{57} & m_{58} & m_{59} \\ m_{67} & m_{68} & m_{69} \\ m_{77} & m_{78} & m_{79} \\ m_{78} & m_{88} & m_{89} \\ m_{79} & m_{89} & m_{99} \end{bmatrix} \begin{bmatrix} \ddot{\phi}_4 \\ \ddot{\phi}_5 \\ \ddot{\phi}_6 \end{bmatrix} + \begin{bmatrix} \dot{\phi}_4 \\ \dot{\phi}_5 \\ \dot{\phi}_6 \end{bmatrix}^T \begin{bmatrix} \begin{bmatrix} c_{77} & c_{78} & c_{79} \\ c_{78} & c_{88} & c_{89} \\ c_{79} & c_{89} & c_{99} \end{bmatrix}_1 \\ \vdots \\ \begin{bmatrix} c_{77} & c_{78} & c_{79} \\ c_{78} & c_{88} & c_{89} \\ c_{79} & c_{89} & c_{99} \end{bmatrix}_9 \end{bmatrix} \begin{bmatrix} \dot{\phi}_4 \\ \dot{\phi}_5 \\ \dot{\phi}_6 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \\ \tau_5 \\ \tau_6 \\ \tau_7 \\ \tau_8 \\ \tau_9 \end{bmatrix} \tag{3}$$

where the elements of the simplified mass matrix and Coriolis tensor are subsets of the original full mass matrix and Coriolis tensor respectively.

4. The Planning Algorithm

The minimum time planning technique presented here is based on a well known algorithm developed for optimizing the motions of conventional manipulators along fixed paths [5], which has also been extended to non-fixed manipulator paths [6]. This algorithm is based on the fact that the minimum time motion of a manipulator along its path is achieved when its acceleration or deceleration is at its maximum at every point along the path. The algorithm finds the switching points between the maximum acceleration and deceleration regions of the path using a function called the Limit Curve.

The Limit Curve, $\dot{S}_m(S)$, is generally plotted in the S - $\dot{S}$ phase plane and is the plot of the maximum velocity that the manipulator may have at any distance S along the path without exceeding the system's capabilities. To find the Limit Curve for a space manipulator, including the constraints imposed by the system's reaction jets Equations (1) must be transformed from an equation in terms of n generalized coordinates to an equation in terms of a scalar path coordinate, such as S, the distance along the path [5]. For a non-redundant space manipulator, the prescribed motion of the manipulator and the spacecraft may be written as a vector function of the generalized coordinates in the form:

$$\underline{\mathbf{P}}(S) = \underline{\mathbf{R}}(\underline{\phi}) \tag{4}$$

where $\underline{\mathbf{P}}$ is generally a 12 element vector representing the position and orientation of the manipulator end-effector and spacecraft, generally given in inertial coordinates. Only a three element $\underline{P}$ vector is required for the system shown in Figure 1 because the motion of the base is nominally stationary and the manipulator has only three DOF. Using Equation (4) and its derivatives, it is possible to transform the equations of motion, Equation (1), into an equation of the form [10]:

$$\underline{m}(\underline{\phi})\, \ddot{S} + \underline{b}(\underline{\phi})\, \dot{S}^2 = \underline{\mathbf{T}} \tag{5}$$

The elements of the $\underline{\mathbf{T}}$ vector in Equation (5) are the joint actuator torques and the forces and moments acting at the spacecraft center of mass. The time optimal algorithm requires that the constraints due to the limits of the the manipulator's actuators and the spacecraft attitude control jets must be stated for the $\underline{\mathbf{T}}$ vector as a function of the state of the system. It is therefore necessary to transform the dynamic equations into a form where the generalized force vector, called $\underline{\mathbf{T}}^*$, consists of both the reaction jet forces (F_1 through F_6) and the manipulator actuator torques (T_7, T_8, and T_9), since the saturation constraints are imposed on these forces and torques. The numbering and locations of the reaction jet forces are shown in Figure 2.

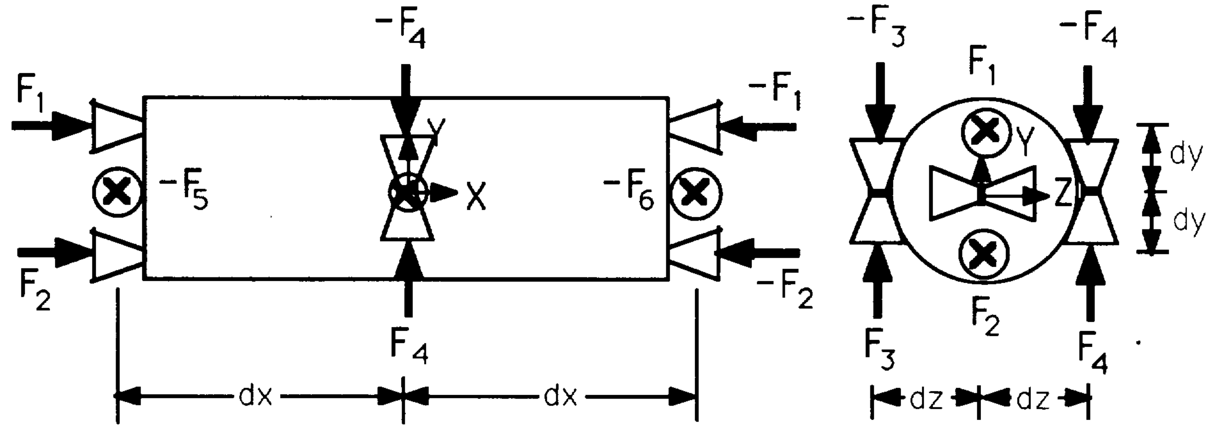


Figure 2. Reaction Jet System

From fundamental mechanics the following transformation may be written:

$$\underline{T}^{*} = \begin{bmatrix} 1/2 & 0 & 0 & 0 & -1/(2dy) & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 1/(2dy) & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & 1/(2dz) & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & -1/(2dz) & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/(2dx) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & -1/(2dx) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \underline{T} \tag{6}$$

Using Equation (6), Equation (5) is then transformed into the form:

$$\underline{m}^*(\underline{\phi})\,\ddot{S} + \underline{b}^*(\underline{\phi})\,\dot{S}^2 = \mathbf{T}^* \tag{7}$$

In this form, the limits on both the capabilities of the reaction jets' forces and the joint motors' torques may easily be taken into account in the formulation of the Limit Curve. The limits on these torques and forces may be expressed as any function of the state of the system:

$$T^*_{imin}(\underline{\phi}, \dot{\underline{\phi}}) \leq T^*_i \leq T^*_{imax}(\underline{\phi}, \dot{\underline{\phi}}) \tag{8}$$

The Limit Curve is found by noting that for each generalized force, a path acceleration value can be calculated as a function of the path position and velocity from Equation (8), or:

$$\ddot{S} = \frac{T^*_i - b^*_i\dot{S}^2}{m^*_i} \tag{9}$$

Note that b^*_i and m^*_i are elements of the vectors $\underline{m}^*$ and $\underline{b}^*$ respectively. From Equation (9), and the limits on forces and torques, it is possible to calculate the range of path accelerations permitted by each joint actuator and reaction jet. The range of accelerations allowed by the complete system is then defined by the limits:

$$\ddot{S}_{min} = \max_i \left(\frac{T^*_{imin} - b^*_i\dot{S}^2}{m^*_i} \right)$$

(10)

$$\ddot{S}_{max} = \min_i \left(\frac{T^*_{imax} - b^*_i\dot{S}^2}{m^*_i} \right)$$

The maximum velocity allowed by the system at any point along the path, $\dot{S}_m(S)$, occurs when the range of allowable $\ddot{S}$ decreases to zero, or $\ddot{S}_{max} = \ddot{S}_{min}$. The function $\dot{S}_m(S)$ plotted in the phase plane defines the Limit Curve which is used by the algorithm to find the optimal switching points as described in detail in reference [5]. Any conventionally planned trajectory lying above the optimal trajectory in the phase plane will violate the constraints imposed by the joint motors and the reaction jets.

This technique has been implemented in a computer software package with extensive computer graphics to aid the planner in visualizing the results of the optimization.

5. Examples

This section describes the application of the planning technique to the example system shown in Figure 1, whose parameters are given in Table I. The system's masses were chosen so that the manipulator's motions would produce significant disturbances on the spacecraft.

Table I. Space Manipulator Parameters.

	Spacecraft	Link 1	Link 2
Mass	80. kg	4. kg	4. kg
Length	3. m	1. m	1. m
Diameter	1. m	0.5 m	0.5 m
Principle Moments of Inertia:			
about X	20. kg-m^2	0.01 kg-m^2	0.01 kg-m^2
about Y	70. kg-m^2	0.34 kg-m^2	0.34 kg-m^2
about Z	70. kg-m^2	0.34kg-m^2	0.34 kg-m
Maximum Joint Motor Output:		15 N-m	
Maximum Reaction Jet Output:		10 N	
Reaction Jet Locations:	dx = 1.25 m	dy = 0.4 m	dz = 0.4 m

The manipulator path for the case discussed is shown in Figure 3. The Limit Curve and optimal trajectory for this case are shown in Figure 4, along with a conventionally planned trajectory. The optimal trajectory required to complete this maneuver is 3.739 seconds, a significant improvement compared to approximately 5.4 seconds required by the conventional plan which uses constant velocity and accelerations. Figure 5 shows that for the optimal trajectory none of the manipulator joint actuators are used to their full capacity; the maximum torque capabilities are shown as hash marks on the vertical axes of each plot. Hence the manipulator's speed is governed by the capabilities of the reaction jets which are at their bounds during the motion, as may be seen in Figure 6. These suggest that this manipulator might be designed with smaller and less powerful motors, which would both reduce system weight and improve performance.

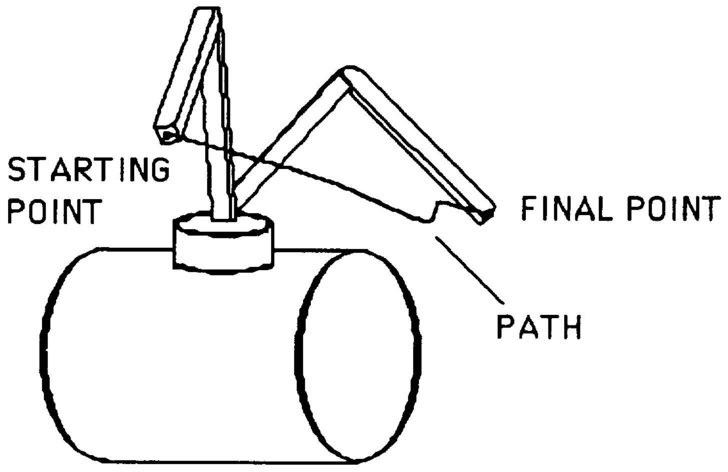


Figure 3. A Three Dimensional View of a Manipulator Path.

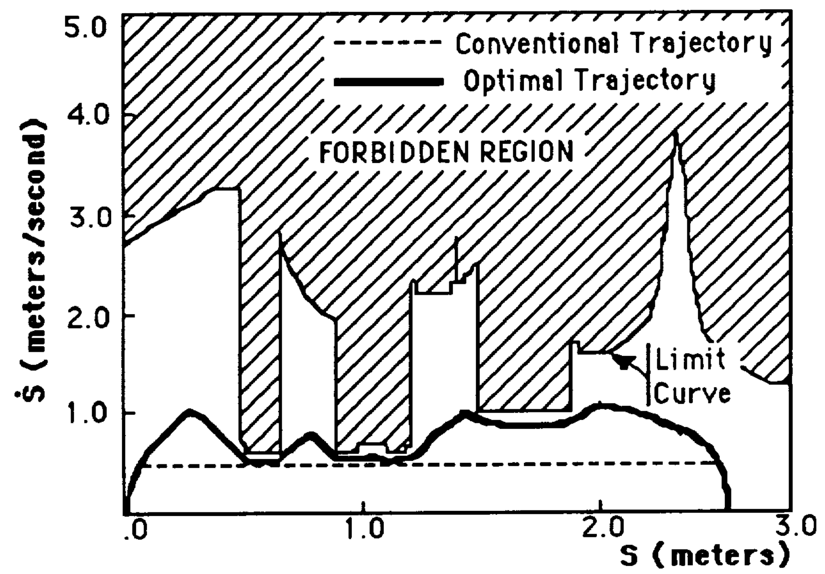


Figure 4. Limit Curve, Optimal and Conventional Trajectories for Example System and Path.

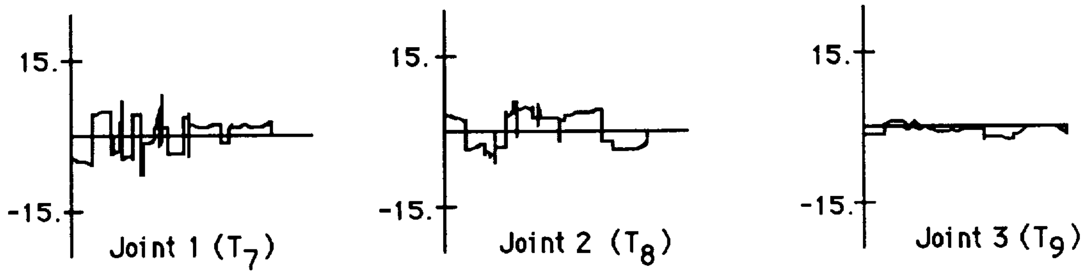


Figure 5. Manipulator joint motor torque profiles (N-m) as a function of Path Distance, S (m).

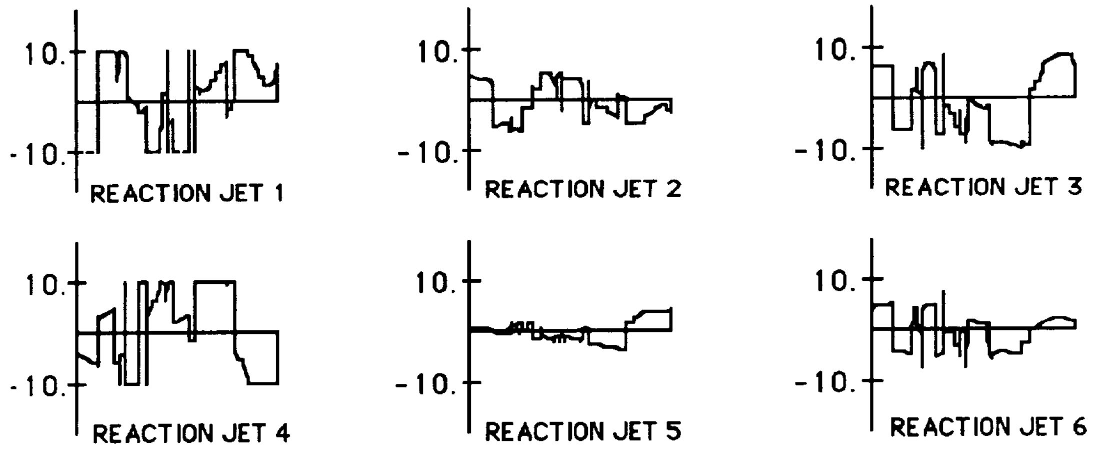


Figure 6. Reaction Jet Force Profiles (N) as a function of Path Distance, S (m).

An independent dynamic simulation was used to verify the results obtained by the algorithm [11]. It showed that the manipulator followed its prescribed path and the spacecraft remained virtually stationary when the joint torques and reaction jet forces calculated with the algorithm were used as dynamic feedforward signals to drive a full nonlinear model of the system: only very small errors were observed due to slight differences between the models and the integration techniques used by the two programs. For example, the simulation showed linear spacecraft displacements of the less than .0006 meters. In real systems modelling errors can lead to undesired spacecraft motions, even with dynamic feedforward. These can easily be corrected by the spacecraft's attitude control system. Figure 7 shows simulation results for the case where the properties of the system used in the planning algorithm were in error by a few percent and a relatively simple PD attitude control system was used to compensate for the errors. The figure shows the feedforward reaction forces and the small contribution required from the closedloop controller to reduce the motions of the spacecraft essentially to zero.

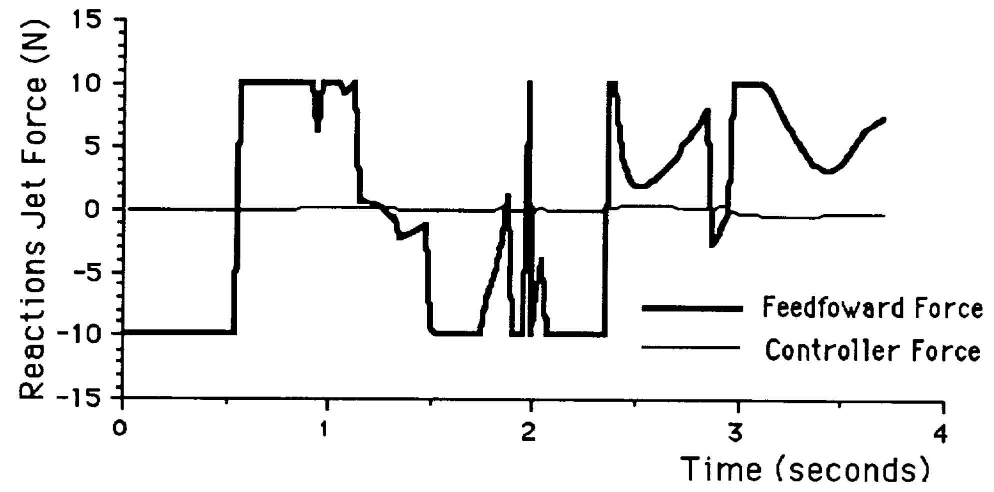


Figure 7. Reaction Jet Forces - Openloop and Control components for System with Modelling Errors.

The importance of the reaction jet forces in holding the spacecraft stationary during the manipulator's motion can be seen in Figures 8a and 8b, which are simulation results for the case where the feedforward signals to the reaction jets are set to zero. Such large linear and angular displacements would be unacceptable in most missions. In most systems the spacecraft's closedloop attitude control system would reduce these disturbance-induced displacements to some degree. However, the simulation results obtained in this study show that trying to control manipulator-disturbed spacecraft motions with feedback control alone can lead to substantial errors, particularly when the attitude control system's bandwidth is limited by system structural resonances and controller sampling times. Based on these results one can conclude that for many systems manipulator disturbances are sufficiently large to require dynamic feedforward compensation in addition to closedloop attitude control.

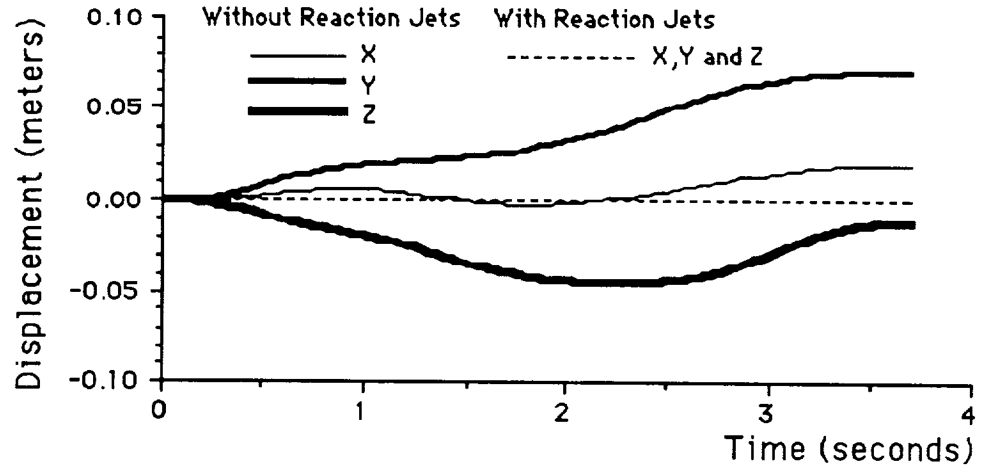


Figure 8 a. Spacecraft Linear Displacements With and Without Reaction Jet Forces (Openloop).

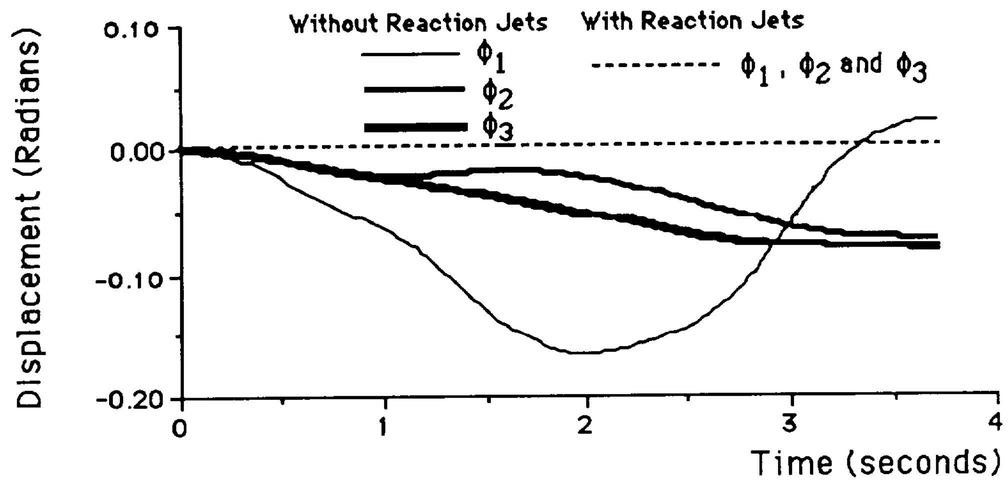


Figure 8 b. Spacecraft Rotations With and Without Reaction Jet Forces (Openloop).

The simulation results obtained in the study also clearly show that saturation of the reaction jet system should be avoided, whether or not manipulator motions are planned in a time optimal manner. Figure 9 shows that the linear motions of the spacecraft became relatively large when the reaction jet forces required to hold the spacecraft during the manipulator's motions exceeded the reaction jet capabilities by 20 percent. The rotational motions also became large. This clearly points out the need to consider the saturation limits of the spacecraft's attitude control system when planning the motions of its manipulator.

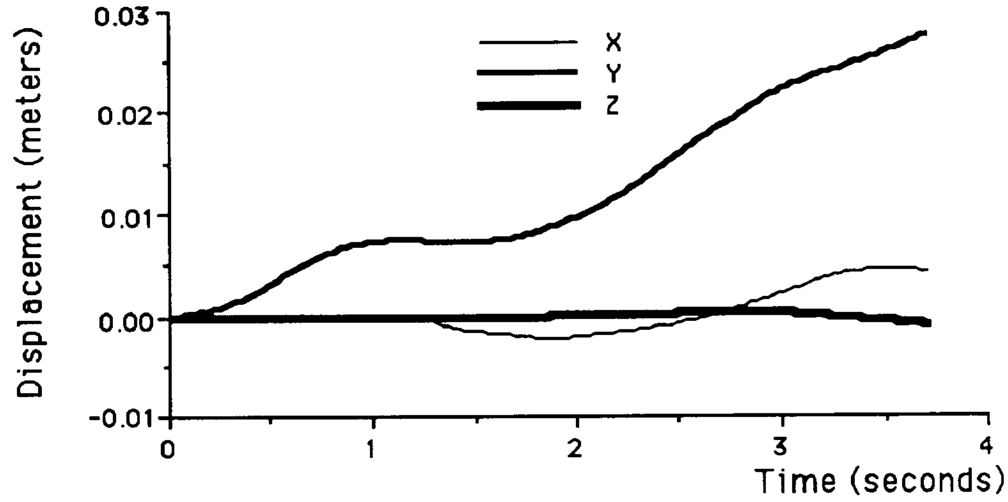


Figure 9. Linear Spacecraft Displacement for System With Reaction Jet Saturation.

6. Conclusions

This paper presents a method for planning the time optimal motions of space manipulators. It considers the constraints of the forces and moments acting on the spacecraft, as well as the constraints of the manipulator joint motors, to calculate a minimum time velocity trajectory for the manipulator. The algorithm has been verified by an independent simulation. The results obtained in the study show that the saturation of a space manipulator system's attitude control jets can be an important problem which should be considered in planning the motions of the manipulator. The technique developed in this paper, combined with a simple attitude control system to compensate for modelling errors, maybe an effective technique for dealing with this problem. The results obtained in this study also suggest that dynamic feedforward techniques may be an important part of any space manipulator control system.

7. Acknowledgement

This work was supported by NASA Langley Research Center Automation Branch.

8. References

[1] Hollars, M.G., Cannon, R.H., and Alexander, H.L. "Experiments in Advanced Control Concepts for Space Robotics" Tenth Annual AAS Guidance and Control Conference, Keystone, CO, January-February 1987.

[2] Dubowsky, S. and Vafa, Z., "A Virtual Manipulator Model for Space Robotic Systems," Proceedings of NASA Workshop on Space Telerobotics, Jet Propulsion Laboratory, Pasadena,

[3] Vafa, Z. and Dubowsky, S.,"Minimization of Spacecraft Disturbances in Space Robotic Systems," Eleventh Annual AAS Guidance and Control Conference, Keystone, CO, January-

[4] Kahn, M.E. and Roth, B., "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains," Journal of Dynamic System, Measurement and Control, pp 164-172,

[5] Bobrow, J.E., Dubowsky, S., and Gibson, J.S., "Time-Optimal Control of Robotic Manipulators Along Specified Paths," The International Journal of Robotics Research, Vol. 4,

[6] Dubowsky, S., Norris, M.A. and Shiller, Z., "Time Optimal Trajectory Planning for Robotic Manipulators with Obstacle Avoidance: A CAD Approach," Proceedings of the 1986 IEEE Conference on Robotics and Automation, San Francisco, CA, April 1986.

[7] Geering, H., Guzzella, L., Hepner, S.A.R. and Onder, C.H., "Time-Optimal Motions of Robots in Assembly Tasks," IEEE Transactions on Automation Control, pp. 512-518,

[8] Meier, E.B., "An Efficient Algorithm for Bang-Bang Control Systems Applied to a Two-Link Manipulator," PhD Dissertation, Department of Mechanical Engineering, Stanford University,

[9] Shiller, Z. and Dubowsky, S., "Global Time Optimal Motions of Robotic Manipulators in the Presence of Obstacles," IEEE Conference on Robotics and Automation, Philadelphia, PA,

[10] Vance, E.E., "Time Optimal Trajectory Planning for Mobile Manipulators," SM Thesis, M.I.T., Cambridge, MA, 1988.

[11] Torres, M.A., "Dynamic Simulation of Manipulators in Space," Technical Report, Department of Mechanical Engineering, M.I.T., Cambridge, MA, 1987.

N90-29972

SYSTEM ARCHITECTURES FOR TELEROBOTIC RESEARCH

F. Wallace Harrison

Automation Technology Branch
NASA Langley Research Center
Hampton, Virginia

ABSTRACT

NASA currently supports several activities related to the definition and creation of telerobotic systems. The effort and investment required to create architectures for these complex systems can be enormous; however, the magnitude of process can be reduced if structured design techniques are applied. A number of informal methodologies supporting certain aspects of the design process are available. More recently, prototypes of integrated tools supporting all phases of system design from requirements analysis to code generation and hardware layout have begun to appear. This paper describes activities related to system architecture of telerobots at Langley Research Center including current activities which are designed to provide a methodology for the comparison and quantitative analysis of alternative system architectures.

1. INTRODUCTION

Much effort is presently being directed within NASA and other government organizations to creating architectures for telerobotic systems [1,2,3]; however, there remains a great deal of confusion over a precise definition of and the scope of the activities associated with the term system architecture. For example, how is system architecture related to computer and network architectures and operating, I/O and other systems? Does a system architecture define the organization of a system or does the organization of a system define a system architecture? We offer these definitions as a basis for the discussions in this paper. A system is an arrangement of things so related or connected as to form a complex or unitary whole. Further, we state that a system must possess a finite set of data, rules, facts, and principles organized and arranged in a regular, orderly manner so that a useful purpose is served. Architecture is the science and the art of construction and design; thus, system architecture is the term used to describe activities which ensure that systems are designed and constructed to meet these definitions. This paper presents a philosophy of telerobotic system construction as reflected in an approach to building telerobotic systems at Langley Research Center (LaRC) in the Intelligent Systems Research Laboratory (ISRL).

Section 2 of this paper is a general discussion of the activities related to system architecture in ISRL. Section 3 describes the Telerobotic System Simulation (TRSS), a real-time telerobotic simulation and run-time system for the investigation of telerobotic technologies. Section 4 discusses a Capability-based Architecture for Robotics (CBAR), a new architecture for building evolutionary, structured capabilities into telerobot systems. Section [5] briefly discusses current activities in system architecture designed to provide the tools for the comparison and quantitative analysis of alternative system architectures. We conclude by summarizing the lessons we have learned to date.

2. Intelligent Systems Research Laboratory

Telerobotics consists of a huge number of highly specialized and interrelated disciplines and technologies. The objective of automation research in the Automation Technology Branch (ATB) at LaRC is to advance technology in specific technology areas (mechanisms, controls, sensors, and operator interface) required for space-based assembly, servicing, and inspection systems [4]. However, meaningful progress in most of these areas depends on having at least a base-level capability in and understanding of the range of telerobotic technologies. The Teleoperator and Robotics System Simulation (TRSS) and the Intelligent Systems Research Laboratory (ISRL) have been developed to provide these base-level capabilities and is structured to allow and promote evolutionary development of telerobotic technologies. Development of TRSS began in 1981 by a small group of researchers and programmers with the objective of investigating the effects of transport delays on operator performance [5]. The graphical simulation ran on Control Data Corp., Cyber 175, used a simulated five degree-of-freedom manipulator, and used displays and controls in a general purpose aircraft simulator.

Parallel to the TRSS development, ISRL was established as a tool to investigate planning systems for robotic systems in a more realistic setting than a purely graphical simulation could provide. Early experiments were directed toward understanding the relationship between perception, reasoning, and manipulation in a simple blocksworld environment for autonomous robots as described in reference [6].

ISRL is organized as a distributed, hierarchical collection of teleoperator and robotic hardware. Each major subsystem consists of a user programmable controller and data communications hardware. In addition to their primary function each subsystem can be used as a program development system or as a general purpose computational element. Primary real-time communications occurs on a 250,000 byte per second packet-switching global bus conforming to the IEEE 488-1978 standard. Device drivers have been written to support demand-driven, priority-based network access and shared file system access. The current network configuration is illustrated in figure 1. Primary subsystems consist of two PUMA manipulators, a vision and laser ranging system, and an operator interface. Interfaces to a Symbolics 3670 computer, a Control Data Corporation Cyber 175 and a Redifusion Poly 2000 high-speed graphics subsystem are provided.

Manipulator and controller. The two digitally-controlled PUMA robotic arms driven by direct current servo systems, provide manipulator functions in ISRL. The PUMA, typically used in "pick-and-place" industrial applications, is a six degree-of-freedom (DOF) anthropomorphic manipulator. It has been augmented with a parallel jaw gripper and a six DOF force torque sensor. In its factory configuration, control is provided by a hierarchical controller composed of a master and six slave microprocessors, each slave providing low-level proportional, integral, derivative (PID) control of one joint of the manipulator. The master controller is an LSI 11-03 microprocessor with 4000 to 32000 bytes of random access memory for user program storage and 28000 to 30000 bytes of read-only memory containing the VAL operating system.

While the controller and VAL are adequate for many industrial applications, its structure and programming do not allow the flexibility necessary in a research environment. To obtain this flexibility, the master controller was replaced with a general purpose computer (LSI 11-73); and, using VAL as a reference, new software was generated with the necessary capability. Computational and input/output facilities are enhanced with the addition of a special-purpose peripheral device drivers, while more general capabilities, such as manipulator initialization, position and rate control, coordinate transformations, and extended I/O are provided by a library of FORTRAN functions.

End Effector. Grasping functions are provided by a microprocessor-controlled parallel jaw end effector having integral force, proximity, base overload, and crossfire detectors. Finger force and torque about the X and Y axis can be sensed. Proximity sensors are simple binary, infrared reflectivity-based emitter-detectors. Cross-fire detectors detect the presence of a work piece between the jaws by interruption of a light beam.

One of the first telerobotic studies in ISRL was an active compliance task 5]. The finger force/torque sensors sensed constraint forces during close tolerance peg insertion and fed this data to control and display modules via the data acquisition system. A simple computer graphics display indicated the magnitude and direction of the binding forces and torques. Using the display the operator could readily command the arm to move to null any disturbing forces. A subsequent modification allowed the operator to select a mode in which the force and torque data were fed directly to the control system to null the force and torques automatically.

Vision. Current ISRL image acquisition and analysis approaches partition this problem area between man and machine, giving each responsibility for that which it does best. Man serves as the basic mechanism for image interpretation and understanding, while the machine vision system performs image acquisition/enhancement/compression and determines the location of objects. Image acquisition is provided by a Data Cube imaging system interfaced to a Micro-Vax II general-purpose processor. Current efforts have centered on determining the three-space location and orientation of labeled objects using a single camera, and research on a multifunction recognition operator for telerobotic vision [7].

The operator is responsible for moving the camera to acquire a labeled object in the field of view. Once acquired, simulation modules are provided for vision-based control of the manipulator. Vision-based control was demonstrated in a recent satellite servicing simulation. After acquiring a label (in this case, a pattern of light emitting diodes) associated with the simulated experiment module, the vision system determined the relative location of the module with respect to the camera and the end effector, and then commanded the manipulator to move to the module under control of the vision system [8].

A coherent laser scanning system has recently become available in ISRL. Laser scanning systems promise high accuracy ranging with television-like displays over wide ranges of ambient illumination [9]. Current efforts are concentrated on generating accurate representations of the manipulator environment for path planning and collision avoidance.

In 1983, TRSS was interfaced to ISRL. Kinematics of the simulation were converted to those of a PUMA 560 and the capability for force control was added. In 1985, automatic control based on vision sensing was added. All TRSS control strategies, based on resolved-motion rate control, are developed as shared control. That is, all automatic control schemes must continuously share control with the operator; thus the operator is free at any time to "help" the automatic system. Conversely, the automatic system is free to aid the operator in task completion. The system requires no complicated semaphores, signaling mechanisms, or logic to switch between automatic and teleoperator control. The most significant feature of the TRSS control strategy is that the reference signal inputs of the manipulator control system is formed as linear combination of outputs from any number of control/sensor modules. The most obvious advantage of this approach is that sensors and associated control modules may be distributed, both spatially and temporally, across multiple processors. Multirate control is almost trivial to implement and the system has some intrinsic fault tolerance. If sensor communications fails, motion based on that sensor output stops.

3. Telerobotic System Simulation

In 1986, a requirement for task-referenced control of multiple manipulators was generated. To facilitate the implementation of these capabilities, the simulation was ported from the CYBER

175 to a VAX/11-750 located in ISRL and a critical examination of existing TRSS code (a single thread of code executed once each clock cycle) and system organization was conducted as part of this activity. The major conclusion of this examination was that the simulation needed to be more modular in organization. An objective in TRSS is that it provide basic capabilities in specific technology areas that are of marginal interest to a researcher. Modularization hides the implementation details of these uninteresting, but interacting, modules so that a researcher need only to understand its behavior and interface to utilize its capabilities. Another conclusion was that top-down design was inappropriate for the target environment. Providing capabilities where none existed before is often the prime objective of a research project. This implies that fixed requirements specifications and hardware architectures are often not available and that even the high-level organization of the simulation may change frequently. TRSS should provide mechanisms to make the reorganization of the simulation possible. A third conclusion was an extraordinarily high percentage of development time was being spent by research personnel on hardware interfaces and communications software. The Teleoperator and Robotics Testbed (TART) was conceived and implemented in response to these limitations. TART consists of a baseline of standard modules and interfaces, termed capabilities, and a logical organization which defines the operating characteristics of a telerobotic system. Capabilities facilitate design, coding, and testing of algorithms and aid the integration of alternative algorithms and software from other sources.

The TART system architecture, as illustrated in figure 2, consists of six layers (L1 through L6), each supported by the functions and capabilities of the layer below. To the programmer, each boundary defines an abstract machine on which the functions of the next higher level are defined. Layers accept commands from and provides feedback to the layer above. The lowest level of TART, the sensor/actuator layer L6, is defined and fixed by the communications protocols and physical characteristics of the collection of devices available in ISRL (see discussion above). Servo level control of manipulators, camera systems, and end effectors and processing of raw sensor data is provided at L5. In some cases, this processing at this layer is provided by vendor-supplied equipment. In other cases, such as machine vision, algorithms are provided by ATB researchers. Inputs to the sensor processing and servo control layer from L4 are the commanded positions and orientations and/or their derivatives for manipulators, grippers, pointing systems, and other devices in their local coordinate system. L5 provides preprocessed sensor and control states to the next higher layer. The electrical voltages and currents to drive motors and actuators are output to L6. If data rates require the interpolation of set points, this is provided in L5.

The communications layer, L4, maps the input and output from the servo/sensor processes to a consistent unambiguous representation. The data structures for all similar devices are required to be identical in form with data scaled to the same units. Thus L4 isolates higher levels from the eccentricities of the underlying hardware and communications protocol. For example, all Cartesian force/torque sensors have a uniform representation which includes a signal indicating overload conditions. If a particular force/torque sensor does not generate this signal in hardware, it must be synthesized in software. Data conversion from device measurement units to laboratory units is also done at L4. The communications layer is maintained by programming support personnel who have experience in dealing with real-time communications. The virtual telerobot architecture provided by L4 functions much like the set of registers available to machine language programmers of computer systems. Just as one does not have to understand the microprogramming and internal data paths of a computer to program it, L4 isolates the researcher from much of the low level programming of the telerobot.

The capabilities required to make the devices in ISRL perform as a system are provided in the coordination and control layer L3. Commands received from L2 are distributed among a number of control capabilities (vision, force, etc.) based on the task to be performed. Tasks that require more than one resource, ie. multiarm control or a compliant grasp, are coordinated in L3 by simultaneously enabling multiple control capabilities with appropriate gains. Coordinate transformations from world to local device coordinates are performed here. Since the capabilities at this level, such as force or vision control, are defined with respect to the TART virtual architecture, the programmer can focus on algorithm development.

Task primitives (move arm(s), close end effector, etc.) are executed by the task level, L2. Capabilities in L2 are activated by commands from L1 and enable appropriate control modules in L3 by setting variables in global memory. L2 then monitors the system as the task is performed and reports success or failure to L1. Each task primitive is implemented as a subroutine and maintained in a library which is directly linked with the TRSS operator interface program or other applications. Levels L3 and L2 taken together provide a capability similar to the intrinsic functions of a compiler.

L1 provides the interface between external systems, both man and machine, and the remainder of TART. Displays for monitoring system status and a command interpreter to decompose high level commands (move to or grasp an object) into sequences of task primitives are provided. L1 provides the remainder of the capabilities found in the typical compiler or interpreter. New task primitives and sequences of commands can be executed, debugged, and evaluated in the operator interface command language before being programmed as an L2 module. For a discussion of the use of TART by external system see [6].

These six levels, their functional descriptions and their logical organization, define the TART system architecture. It should be noted that any implementation is a compromise among a number of competing requirements (speed, ease of use, cost, etc.). The principle function of the TART implementation architecture is to support telerobotics research, and ease of use and reduced cost are emphasized at the expense of performance. In general, layers 4 and above are implemented on a single MicroVax II microprocessor under the VMS operating system. The virtual machine architecture is implemented as a installed shareable image making its data structures global to all processes. Each layer of the architecture is implemented as one or more VMS processes and each process, once started, is responsible for its own scheduling using the mechanisms offered by the VMS operating system. Every capability is implemented as a separate process. To the researcher, this means that refinements and alternatives to algorithms can easily be investigated by stopping the current process and starting a new. However, the requirement that each capability in the TART architecture be a separate process is a major source of overhead and the global accessibility of the virtual architecture can lead to abuse of the TART design philosophy and subtle programming errors. The Capability-Based Architecture for Robotics (CBAR), is being developed in response to these problems.

4. A Capability-Based Architecture for Robotics (CBAR)

The system architecture of CBAR is based on a more formal definition and representation of a capability. The motivation for creating the abstraction of the capability is related to human limitations in the ability to manage and understand information and control flow in the design of complex systems. To overcome this limitation, capabilities have clearly defined function and operate on a small set of interface data structures with control flow limited to that implied by changes in each capability's data structures. No explicit command/response mechanism is required. CBAR encourages the use of modularization in the design process and is amendable to both top-down (recursive application of the abstraction) and bottom-up (utilization of existing abstractions) design practices. At all levels of the system design, both function and interface are represented in only sufficient detail to understand the immediate design and implementation problem. Both the designer's and implementer's ability to efficiently create systems that behave properly and are easy to maintain and document is enhanced.

A capability, C, is defined by its set of data structures and an associated transformation and is represented in figure 3 where P is called the planning data structure, Q is the query data structure, R is the output data structure, S is the sensor data structure, T is a capability transformation, and A is a set of attributes. Functionally, a capability transforms S, the world as seen through its sensor data structure, to P, the world as we wish it, by transitioning through a series of states Q while producing output R to effect changes on its environment.

To better understand the representation, consider a control engineer's model of a simple feedback controller capability (see figure 4). The controller has some state (Q) and an algorithm (T) which forms an output (R) to control a plant based on a set point (P) and feedback from sensors (S). A more general interpretation can be taken from planning systems. Here a particular situation or configuration is a problem state (Q) which is derived from sensor data (S) and previous states. An operator (T) transforms a given state into another state while producing output (R). A solution to a problem is a sequence of operators that transforms an initial state into a goal state (P). Complex functional capabilities can be generated by repeated application of the capability abstraction as illustrated in figure 5. The behavior of a system at any given time is described by the set of active capabilities and their interconnections and can be represented as a lattice.

The flow of data in the CBAR lattice is bidirectional and nonstationary. Goals flow from the top of the lattice to the bottom through the P data structures with increasing levels of detail. State information flows from bottom to top through the Q data structures with decreasing detail. The form of the lattice is determined by the active set of goals and states.

Two degenerate capabilities that are of interest are best explained by an example. Consider the feedback controller discussed above used to control a plant consisting of an amplifier-driven motor and position encoder. The problem, depicted in figure 6, is to generate a set of motor currents to drive the motor through amplifier A to a state sensed through encoder E. C3, called an input capability, transforms (not necessarily linearly) from sensor E internal units to a more convenient representation (angular displacement, velocity, etc.) and makes the coding of C1, the control algorithm, much more tractable. C2, an output capability, functions in a similar manner but generates actuator control signals.

Dynamic reconfiguration of the lattice is accomplished through contact and disconnect mechanisms. A capability ready to execute is said to be contacted. Capabilities are contacted by a name, C, assigned as they are created (currently the file name qualified by subroutine name) with a desired set of access rights. The access rights are compared with the current accessibility attributes and the contact is either allowed or denied. If a contact is allowed, the capabilities exchange information regarding the location of their data structures and a count of the number of contacts is incremented.

Links in the lattice are broken when a capability is no longer required via a disconnect mechanism and a capability with no contacts is said to be disconnected. A capability receiving a disconnect signal decreases its contact count and, if zero, terminates execution.

Accessibility attributes refer to the permissible read and write access modes applied to the data structures of a capability and are placed to the right of the directed line segments. Any capability always has read access to its own planning data structure (P) and write access to its query data structure (Q); however, external access to P and Q are strictly enforced. Read access (RA·) means that only a single external routine can read a capability's Q data structure and implies that a capability can be contacted by only a single external source. For example, in figure 6 capability C1 has exclusive ownership of C2 preventing other processes from controlling the amplifier drive signal. Write access (WA) means that only a single external process can write to a capabilities P data structure, but is not sufficient in itself to insure sole ownership. Read shared (RS) and write shared (WS) access means that multiple external routine have access to a capabilitiy's P and Q data structures respectively. In figure 6, capability C3 is contacted by C1 with RS access implying that other external routines may access and monitor the position and rate of the motor.

Consider an expanded three level implementation of the previous motor controller example in figure 7. Levels L2 and L3 are identical in form and function to figure 6; however, an additional layer L1 has been added to monitor the controller and provide a status indication to external routines. The sensor capability, C3, has been contacted with RS access to allow simultaneous access to its Q data structure by both C1 and C4. Note that C4 could have monitored position and rate through Q3 but this results in a communications delay, less reliability, and more compute overhead.

5. CURRENT WORK

Predicting the performance of architectures for complex systems during early design and development is a difficult and demanding task. Typically, accurate quantitative measures are unavailable until late in the design process leaving many key decisions to rest on the experience and prejudices of the designer. Until the subjective elements of the process can be significantly reduced, the design and implementation of system architectures will remain more an art than a science. Today, it would be unthinkable to attempt the design of a computer system without appropriate computer-aided design tools. The quality and quantity of tools for the VLSI design have increased dramatically since their introduction in the 1960's. More recently, similar tools have been introduced for software design; however, few tools to aid the integration of both hardware and software in the creation of complex system (such as telerobots) exist. The Architecture Design and Assessment System (ADAS), a set of computer-aided design tools supporting system design from initial concept through hardware/software implementation, has been developed by Research Triangle Institute (RTI) [10]. ADAS is a collection of tools that can be used to identify pathologies early in the design process so that alternatives can be created and analyzed. LaRC and RTI are currently modeling TRSS and the TART architecture using ADAS with the objective of developing a methodology for the comparative analysis of telerobotic architectures.

Using the ADAS graphical interface, the hardware and software designs of three systems: TRSS/TART, TRSS/CBAR, and the FTS/NASREM are being modeled and analyzed. TRSS/TART, because of the availability of detailed performance data, will be used to validate the modeling and analysis techniques. ADAS is also an integral element of the CBAR design and implementation. The design will be captured as data flow graphs and mapped to hardware. All design decisions will be evaluated via simulation to pinpoint bottlenecks and determine the best partition between hardware and software. As details of the NASREM/FTS architecture are made available, functional simulations will be conducted to validate the design.

The CBAR implementation architecture is currently under development. A simplified representation of TRSS control architecture in CBAR form is being used as a model for the first prototype. Major design decisions are being evaluated using the Architectural Design and Assessment System to provide quantitative performance analysis before implementation.

6. CONCLUDING REMARKS

At LaRC, architectural endeavors have evolved into two distinct activities. System architecture activities are concerned with the functional characteristics and the logical organization of a system. As such, it is principally a conceptual and philosophical activity that results in a system design specification, a detailed understanding of what the system is to do and how it is logically organized. The implementation architecture phase translates the system design specification into a detailed implementation plan by mapping the system architecture onto the hardware and software subsystems. The partitioning has important practical implications. Partitioning the architectural design makes each phase more tractable by limiting the number of degrees of freedom in each step; and, properly done, new device technologies and algorithms can be incorporated into the system without scrapping or compromising the entire design specification. For example, several families of computers have been designed which share common system architecture with performance ranges of 1-100 and yet most software written and compiled and linked on one machine can run on any other family member.

The role of the system architect in a research environment is at best difficult. The system architect is typically concerned with topics such as requirements engineering, design specification, implementation architectures, testing, validation, and maintenance. It is sufficient to say that to most researchers, these topics are not of paramount importance, yet the difference between a project's success and failure is often correlated with the quality of its supporting systems engineering. Certainly, the utility of the project's results to others is enhanced when sufficient attention is given.

What are some of the attributes of a "good" system architecture? First, there is a consistent design philosophy. Well-defined software and hardware structures with a carefully designed minimized rule set are the result and enable programmers to create efficient and maintainable applications. High-level languages and adequate documentation are absolute requirements. We maintain that the current debate over the need for a "standardized" telerobot control system architecture is unnecessary and distracting. With imagination, thought, and commitment we can build systems which instead of stifling creativity, will encourage it; instead of locking us into proprietary systems, will facilitate their introduction; and instead of being rigidly confining, will be easily extendable.

BIBLIOGRAPHY

[1] Albus, James S., McCain, Harry G, Lumia, Ronald: "NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)," National Bureau of Standards, NBS Technical Note 1235, July 1987.

[2] Matijevic, J. and Dolinsky, S.: "Functional Requirements for the Telerobotic Testbed Project," Jet Propulsion Laboratory, JPL internal document D-3693, December 1988.

[3] Harrison, F. Wallace, and Pennington, Jack E.: "System Simulations Supporting NASA Telerobotics," presented at the Workshop on Space Telerobotics, Pasadena, California, January 1987.

[4] Sliwa, Nancy E.: "Telerobotic Research at Langley Research Center," presented at the Space Operations Automation and Robotics Conference, Houston, Texas, August 1987.

[5] Pennington, Jack E.: "A Rate Controlled Teleoperator Task with Simulated Time Delays,"NASA Langley Research Center, TM-85653, September 1983.

[6] Orlando, Nancy E.: "An Intelligent Robotics Control Scheme," presented at the American Controls Conference, San Diego, California, June 1984.

[7] Goode, Plesent W, and Cornils, Karin: "Monovision Techniques for Telerobots," presented at the Workshop on Space Telerobotics, Pasadena, California, January 1987.

[8] Cornils, Karin and Goode, Plesent W.: "Location of Planar Targets in Three Space from Monocular Images," presented at the Goddard Conference on Space Applications of Artificial Intelligence and Robotics, May 1987.

[9] Goodwin, F. E.: "Coherent Laser Radar 3-D Vision Sensor," Proceedings of Sensors '85, November 1985.

[10] Ingogly, W. F., McLin, D. M., Morrill, R. R., Frank, G. A., and Franke, D. L.: "The Evaluation of 1750A Hardware Implementation Using ADAS," NASA Langley Research Center, CR-178247, January 1987.

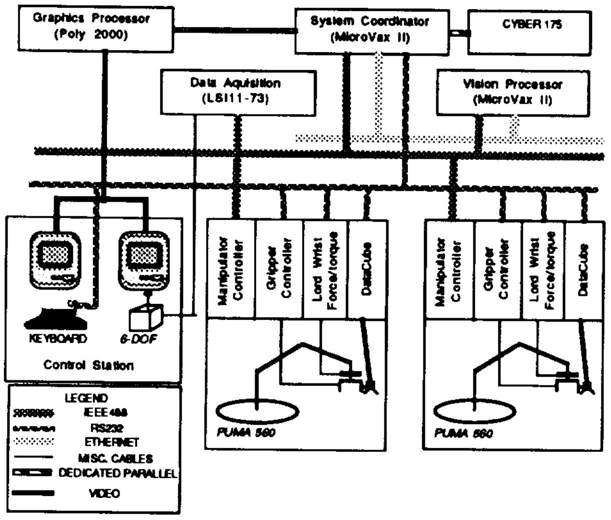


Figure 1. ISRL Physical Layout.

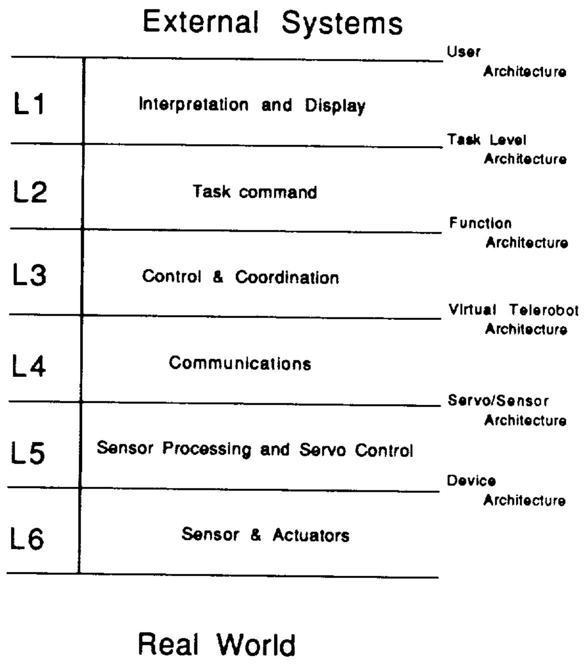


Figure 2. The TART System Architecture.

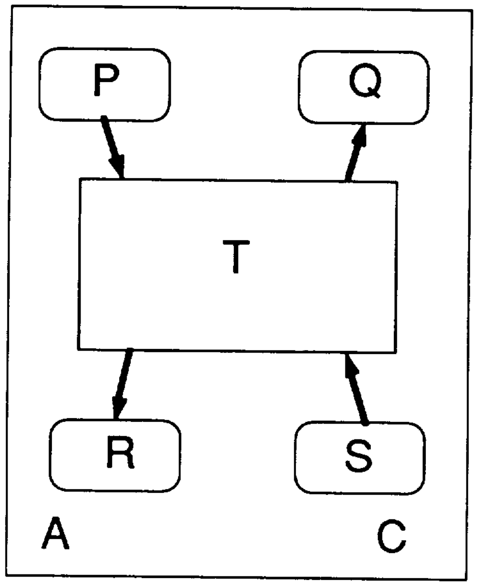


Figure 3. Capability Representation.

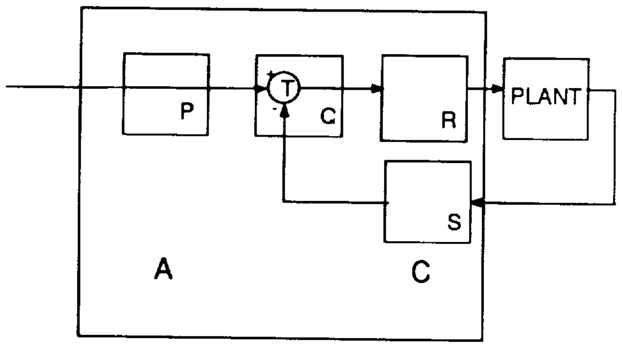


Figure 4. Control system interpretation of a capability.

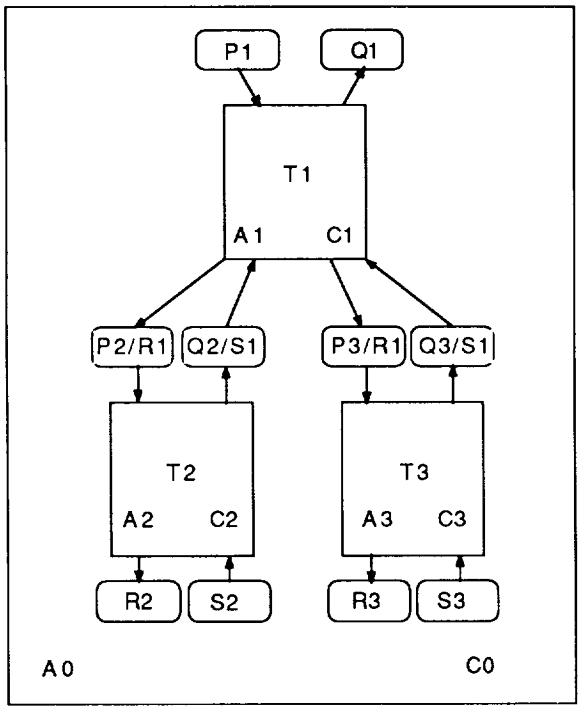


Figure 5. Hierarchical Capability Representation.

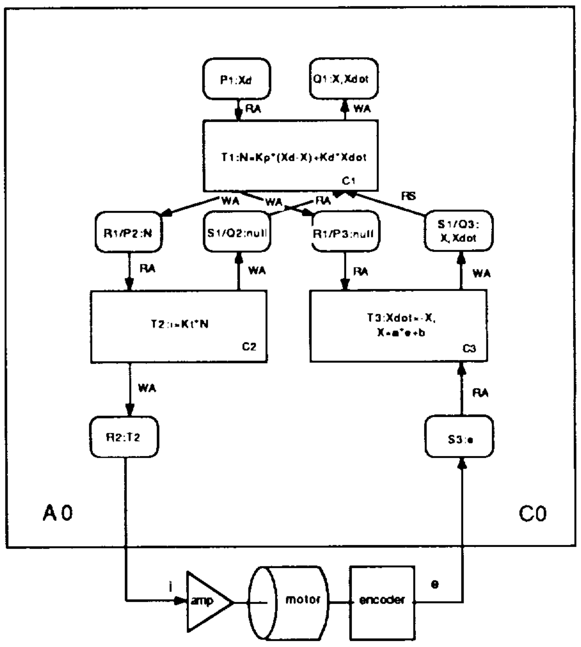


Figure 6. Simple control system.

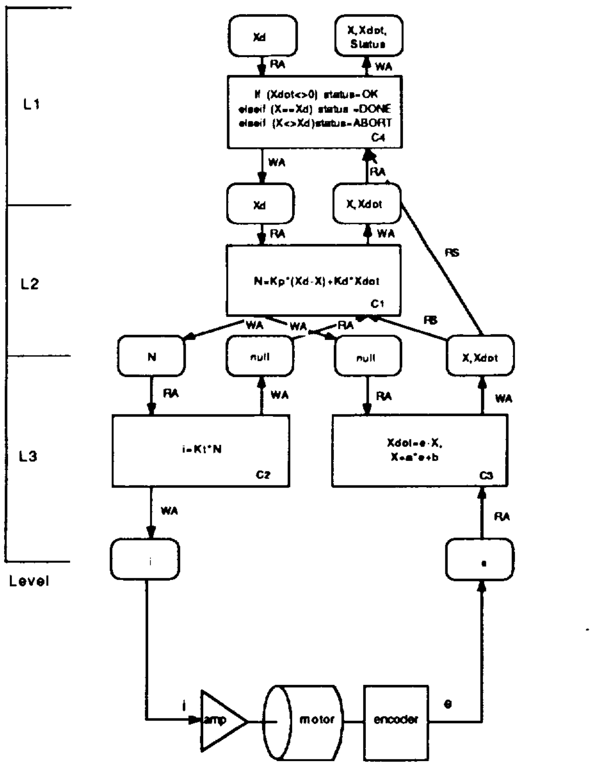


Figure 7. Expanded control system design.

N90-29873

COMPARISON OF JOINT SPACE VERSUS TASK FORCE LOAD DISTRIBUTION OPTIMIZATION FOR A MULTIARM MANIPULATOR SYSTEM

Donald I. Soloway
NASA Langley Research Center
Hampton, Virginia 23665-5225

Thomas E. Alberts
Old Dominion University
Norfolk, Virginia

Abstract

It is often proposed that the redundancy in choosing a force distribution for multiple arms grasping a single object should be handled by minimizing a quadratic performance index. The performance index may be formulated in terms of joint torques or in terms of the Cartesian space force/torque applied to the body by the grippers. The former seeks to minimize power consumption while the latter minimizes body stresses. Because the cost functions are related to each other by a joint angle dependent transformation on the weight matrix, it might be argued that either method tends to reduce power consumption, but clearly the joint space minimization is optimal. In this paper, a comparison of these two options is presented with consideration given to computational cost and power consumption. Simulation results using a two arm robot system are presented to show the savings realized by employing the joint space optimization. These savings are offset by additional complexity, computation time and in some cases processor power consumption.

1. Introduction

Some of the recent developments in multiple arm manipulation of a commonly grasped body include the work of Hayati [1], Alberts [2], and Carnignan [3]. The common thread between these papers is that each employs the minimization of some form of quadratic performance index to choose an appropriate load distribution. Hayati proposed an extension of Mason's [4] hybrid position/force control in which the inertia of each arm is artificially extended to include a portion of the payload inertia. From a practical point of view, the method may be difficult to implement effectively, because it requires precise knowledge of the inertial properties of the arms and of the jointly manipulated objects as well as the solution of inverse dynamics. Alberts closes a force feedback loop around a kinematic resolved rate controller for multiple arms, thus realizing the Damping Control Method. As in Hayati's work the problem of redundancy in determining the distribution of load among the manipulators is handled by minimizing a quadratic cost function in task-space force and torque. This tends to minimize internal forces in the body while maintaining control over a prescribed force and torque interaction with the external environment. Alberts formulation does not consider the closed chain dynamics of the manipulators and payload, but rather each manipulator is viewed as an actuator with an independent control system. Carnignan used a quadratic cost function to minimize joint space torques including those due to manipulator kinetics, but due to the inclusion of the manipulator kinetic effects, undesirable internal forces may be produced in the body. This method is computationally more expensive than task space optimization.

It can be argued that minimizing a quadratic cost function in joint space has greater power efficiency than a minimization in task space, but at what computational cost? This paper compares the task space versus joint space cost functions with respect to power efficiency and computational cost. The development is to be based on Alberts' task space cost function and a joint space cost function developed along similar lines. The computational cost of using the joint space minimization scheme is similar to Carnignan's method. The torques required to compensate for manipulator kinetics are not included in the minimization so as to avoid imposing unnecessary internal forces and torques within the manipulated body.

2. Power Cost Calculation

In a multiarm robotic system, minimization of a joint torque based quadratic cost function would tend to minimize the dissipated power used by the motors under static conditions.[1]

A minimization scheme to establish an "optimal" force distribution could be formulated in either joint space or task space. From the standpoint of power used to drive the joints, it would be optimal to formulate the minimization in joint space. In this paper the question considered is that of how much power really can be saved by optimizing in joint space as opposed to task space.

Based upon a task space quadratic performance index

$$Q_t = \underline{\Gamma}_t^T W_t \underline{\Gamma}_t \tag{1}$$

the task space optimal load distribution force equation [2] is

$$\underline{\Gamma}_t = \Lambda_t \underline{F} \tag{2}$$

where

$$\Lambda_t = W_t^{-1} H^T [H W_t^{-1} H^T]^{-1} \tag{3}$$

and $\underline{\Gamma}_t$ is a vector of wrench vectors (as in Equation 15, Appendix I) such that each components $\underline{F}(i)$ of $\underline{\Gamma}_t$ is a wrench vector applied to the body at the grasp point of the $\underline{i}$th manipulator and subscript t denotes task space. H is a matrix of Jacobian transformations that depends upon the locations of the grasp points of each arm relative to the applied external force $\underline{F}$. The weight

[1]To truly minimize power consumption under dynamic conditions the cost function should include a term representing the product of joint torque and joint velocity. This is discussed further later in reference to Equation (9).

matrix is given by $W_t = \text{diag}(v_1, v_2, \ldots, v_N)$ where N is the number of arms and v_i is a diagonal weight matrix reflecting the relative cartesian end effector force and torque capabilities of the ith arm.

Using a similar development (Appendix I) but, based upon a joint space quadratic performance index

$$Q_j = \tau^T W_j \tau \tag{4}$$

the joint space optimal load distribution force equation is

$$\underline{\Gamma}_j = \Lambda_j \underline{F} \tag{5}$$

where

$$\Lambda_j = J^{-T} W_j^{-1} J^{-1} H [H J^{-T} W_j J^{-1} H^{-T}]^{-1} \tag{6}$$

with $W_j = \text{diag}(w_1, w_2, \ldots, w_N)$ and w_i is a diagonal matrix whose elements are the squares of the reciprocals of the relative joint torque capabilities of the ith arm. Subscript j denotes joint space.

The cost function is an indication of power used since torque (τ) is proportional to current and current squared is proportional to power dissipated. To compare power consumption on an equal basis the cost of task space optimized load distribution was evaluated in terms of the joint space cost function. Thus, for evaluative purposes, the torque-squared cost of executing the task space optimization scheme can be expressed as

$$Q_{tj} = \tau_t^T W_j \tau_t \tag{7}$$

where $\tau_t = J^T \Lambda_t \underline{F}$

and the cost associated with joint space optimization is given by Equation (4), that is $Q_{jj} = Q_j$. The cost resulting from the joint space formulation will always be less than that of the task space formulation according to the above criterion.

The power used by a motor is

$$\text{Power} = I_a^2 R + \frac{K_E}{K_T} \omega T m + \frac{K_E}{K_T} \omega T_l \tag{8}$$

where I_a is the armature current
R is the armature resistance
K_E is the back emf constant
K_T is the torque constant
ω is the angular speed of the rotor
T_l is the torque of the load

$T_m = T_f + D_\omega$ where T_f is constant friction and D is the viscous damping coefficient.

The current I_a is related to the dynamcs of the motor by

$$I_a = \frac{1}{K_T} [(J_m + J_l)\frac{d\omega}{dt} + T_m + T_l] \tag{9}$$

where J_m is the motor inertia and J_l is the load inertia. Observe that in the joint space minimization presented here (7) minimizes the power due to the term $I_a{}^2R$ in (8) but does not account for the power associated with ωT_l in the last term of (8). From a practical point of view it appears that the contribution of this term will normally be small. In simulations conducted, in which the load velocity was 0.1 m/s, the ωT_l term resulted in power difference of less than 1%.

The percent difference in power is defined as follows

$$P_N = \frac{P_t - P_j}{P_j} * 100 \tag{10}$$

where P_t and P_j represent the power used by task and joint space respectively.

The power difference is

$$P_D = P_t - P_j \quad \text{where } P_D \text{ is in watts} \tag{11}$$

A simulation[2] was used with the system parameters of the NASA LTM[3] (given in Appendix III) to compute the power used by the two-arm LTM system in moving a 40 lb. payload on Earth. The center of gravity of the object was at the grasp point of one of the arms. The two arms moved horizontally without rotating the object. For this trajectory P_D and P_N were plotted in figures 1 and 2 respectively. The initial P_J = 1093 watts.

[2]The simulation is a combination of ROBSIM (a full dynamics robot simulator developed by NASA LaRC) and user coded. Robsim simulated the robotics dynamics and the user code calculated the load distributed joint torques and costs.

[3]LTM is an acronym for the Laboratory Telerobotic Manipulator developed for NASA LaRC by Oak Ridge National Laboratory. The LTM is a seven-degree-of-freedom arm employing differential friction drive joints. For simplicity the differential joint drive are treated as conventional gear driven joints.

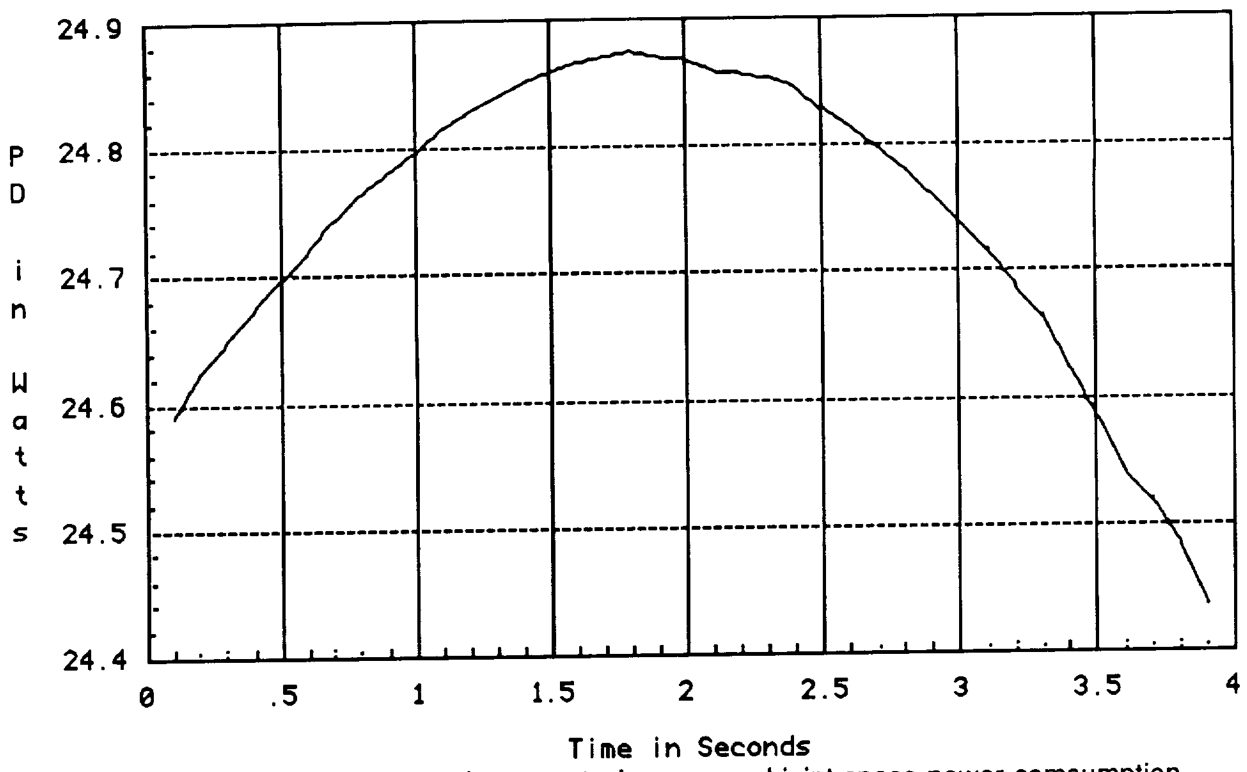


Figure 1. The difference between task space and joint space power comsumption.

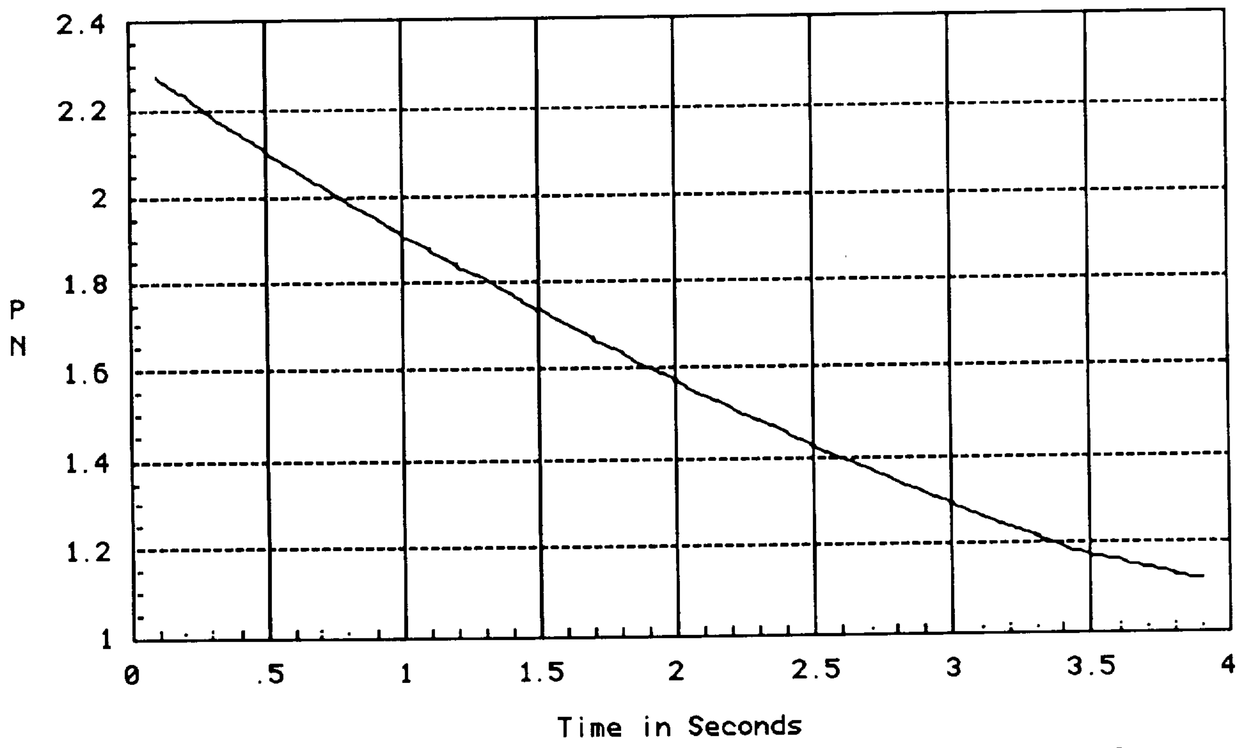


Figure 2. The percentage of power saved with respect to the total power used in joint space.

3. Computational Cost Calculation

In a real-time computer program it is desirable to keep computationally expensive operations to a minimum. The number of operations it takes to calculate task space and joint space load distribution are determined below.

For this analysis the following assumptions are made:

1. The manipulators grasp points are fixed while the body is in motion, thus H remains a constant matrix.
2. W does not change over the motion of the body.
3. All manipulators have the same number of degrees of freedom.

In task space Λ is constant thus requiring only a matrix vector multiply to calculate Γ_t. Λ is a 6N x 6 matrix where N is the number of manipulators grasping the body, and F is a 6 x 1 vector needing:

36N	multiplies;
30N	additions

to calculate Γ_t.

Joint space will have the same operations as above plus the operations to calculate Λ. The total number of operation needed to calculated τ_j as derived in Appendix II is:

N(48n + 72) + 195	multiplies
71	divides
N(48n + 30) + 206	additions

With the need to calculated the jacobian inverse the total number of operations to calculate Γ_j as derived in Appendix II is:

N(60n + 267) + 195	multiplies
71N + 71	divides
N(58n + 236) + 206	additions

Currently the most powerful space qualified procesor is the Harris 80C86. Based on using this processor with a 5 mhz clock and the number of arms and degrees of freedom (DOF), the results Table I, II, and III are obtained. The entries in the tables represent the number of times the computation can be executed in 1 second.

	Number of Arms		
Degrees of Freedom	2	3	4
6	17.5	12.4	9.54
7	16.4	11.5	8.86
8	15.3	10.7	8.27

Joint Space with Jacobian Inverse Calculated
Table I

Degrees of Freedom	2	3	4
6	29.8	21.9	17.3
7	27.2	19.8	15.6
8	25.0	18.1	14.2

Joint Space without Jacobian Inverse Calculated
Table II

Degrees of Freedom	2	3	4
6	413	275	206
7	413	275	206
8	413	275	206

Task Space
Table III

The computational cost of task space optimization is small enough to execute the code on an existing processor whereas joint space optimization would likely require a separate processor. In this case the power required to operate additional processing equipment must be considered. The power requirement of an 64K 80C86 board based on Harris radiation hardened components is 30 watts.

4. Summary

It has been shown that an apparently significant amount of power can be saved by employing the joint space optimized load distribution. In the example presented the largest savings over task space optimization realized was about 25 watts. This can be viewed as an extreme case. It is important to note, however, that the joint space optimization represents a substantial increase in computational burden. If this results in a need for additional processors, the power required to operate them might offset the savings realized by the optimization scheme.

In the future, space qualified processors will be available that are much more powerful and possibly more power efficient than those now used. In the case where a surplus of efficient computing power is available, the joint space optimization may prove to be the method of choice.

5. Joint Space Optimal Load Distribution

Assume that a desired net cartesian space wrench vector $\underline{F}$ acting on the body with known point of application p is specified. The wrench vector is made up of cartesian space force $\underline{f}$ and moment $\underline{m}$ vectors such that

$$\underline{F} = \begin{bmatrix} \underline{f} \\ \underline{m} \end{bmatrix} \tag{12}$$

This wrench could be to counteract gravitational loading or inertial reactions due to body acceleration, or to apply a force to the external environment through the jointly manipulated object. Now consider a system of N manipulators, where $\underline{F}(i)$ denotes a cartesian space wrench applied to the jointly manipulated body by the end effector of arm i. The following conditions must be satisfied in order to establish equilibrium:

$$\underline{f} = \sum_{i=1}^{N} \quad (13) \qquad \text{and} \qquad \underline{m} = \sum_{i=1}^{N} \left(\underline{m}(i) + \underline{r}_p(i) \times \underline{f}(i) \right) \tag{14}$$

where $\underline{f}(i)$ and $\underline{m}(i)$ are the force and moment vectors, respectively, that make up $\underline{F}(i)$

$$\underline{F}(i) = \begin{bmatrix} \underline{f}(i) \\ \underline{m}(i) \end{bmatrix} \tag{15}$$

and $\underline{r}_p(i)$ is a vector drawn from the point of application p of the force $\underline{f}$ to the grasp point for manipulator i. The minimization procedure operates on a quadratic function Q in the vector τ, with positive definite symmetric weighting matrix W

$$Q = \tau^T W \tau \tag{16}$$

where

$\tau = J^T \Gamma$, J^T is the jacobian transpose of the manipulator

$$\underline{\Gamma} = \begin{bmatrix} \underline{F}(1) \\ \cdot \\ \cdot \\ \underline{F}(N) \end{bmatrix} \quad \text{and} \quad W = \text{diag}\,(w_1, w_2, \dots, w_N) \tag{17}$$

Conditions (13) and (14) are expressed in matrix form as

$$\left[\begin{array}{c|c}I_3 & 0 \\ \hline [r_p(1)X] & I_3\end{array}\;\vdots\;\begin{array}{c|c}I_3 & 0 \\ \hline [r_p(2)X] & I_3\end{array}\;\vdots\;\cdots\;\vdots\;\begin{array}{c|c}I_3 & 0 \\ \hline [r_p(N)X] & I_3\end{array}\right]\begin{bmatrix}F(1)\\ \vdots \\ F(N)\end{bmatrix} = F \tag{18}$$

where, I_3 is a 3 by 3 identity matrix and $[\underline{r}_p(i)X]$ is a matrix that operates on $\underline{f}(i)$ to form the cross product $\underline{r}_p(i)X\ \underline{f}(i)$. A more compact expression for (18) is

$$HJ^T\ \tau = \underline{F} \tag{19}$$

Using a Lagrange multiplier λ to append equation (19) to (16) the augmented cost function is obtained.

$$\tilde{Q} = \tau^T W\tau - \lambda^T(HJ^{-T}\tau - \underline{F}) \tag{20}$$

The optimal solution must satisfy

$$\frac{\partial\tilde{Q}}{\partial\tau} = 0, \text{ that is } \ 2\tau^T W - \lambda^T HJ^{-1} = 0 \tag{21}$$

Rearranging and applying equation (19) to equation (21)

$$\tau = \frac{1}{2}\ HW^{-1}J^{-1}H^T\lambda \ \text{ which yields } \ \lambda = 2(HJ^{-T}W^{-1}J^{-1}H^T)^{-1}\underline{F} \tag{22}$$

Now upon eliminating λ between equations (21) and (22) the final expression for joint torques is obtained.

$$\tau = W^{-1}J^{-1}H^T(HJ^{-T}W^{-1}J^{-1}H^T)^{-1}\underline{F} = \Lambda\underline{F} \tag{23}$$

Appendix II

To count the operations in Λ it is necessary to first simplify Λ into two matrices A and B, such that

where $\quad \Lambda = AB^{-1}$, $A = J^{-T}W_j^{-1}J^{-1}H^T$ and $B = HA$

Calculating A:

$$\begin{bmatrix} J_i^{-1} & J_2^{-1} & \cdot & 0 \\ & 0 & \cdot & J_N^{-1}\end{bmatrix}, \ W_j^{-1} = \begin{bmatrix} W_1^{-1} & W_2^{-1} & \cdot & 0 \\ & 0 & \cdot & W_N^{-1}\end{bmatrix} \text{ and } \ H^T = \begin{bmatrix} H_1^T \\ \vdots \\ H_N^T\end{bmatrix}$$

where J_i^{-1} is the jacobian inverse for manipulator i,

W_i^{-1} is the weight matrix inverse for manipulator i,

and H_i^T is the transform grasp point to controlled force F for manipulator i.

so

$$A = J^{-T}W_j^{-1}J^{-1}H^T = \begin{bmatrix} J_1^{-1}W_1^{-1}J_1^{-T}H_1^T & & 0 \\ & \ddots & \\ 0 & & J_N^{-1}W_N^{-1}J_N^{-T}H_N^T \end{bmatrix}$$

and

$$B = HA = \begin{bmatrix} H_1J_1^{-1}W_1^{-1}J_1^{-T}H_1^T & & 0 \\ & \ddots & \\ 0 & & H_1J_N^{-1}W_N^{-1}J_N^{-T}H_N^T \end{bmatrix}$$

The cost to calculate $J_i^{-T}W_i^{-1}J_i^{-1}H_i^T$ is:

$$J_i^{-1}H_i^T = \begin{bmatrix} n \times 6 \\ \text{full} \\ \text{matrix} \end{bmatrix} \left[\begin{array}{ccc|ccc} 1 & & 0 & o & z & -y \\ & 1 & & -z & o & x \\ 0 & & 1 & y & -x & o \\ \hline & & & 1 & & 0 \\ & 0 & & & 1 & \\ & & & 0 & & 1 \end{array}\right]$$

at a cost of n*(6 multiplies + 6 additions)

$$W_i^{-1}[J_i^{-1}H_i^T] = \begin{bmatrix} n \times 6 \\ \text{full} \\ \text{matrix} \end{bmatrix}$$

at a cost of n*6 multiplies

$$J_i^{-T}[W_i^{-1}J_i^{-1}H_i^T] = \begin{bmatrix} 6 \times 6 \\ \text{full} \\ \text{matrix} \end{bmatrix}$$

at a cost of n36 multiplies + (n-1)36 additions. The total cost of A is N[48n multiplies + (48n-36) additions].

Calculating B:

$$B_i = H_i A_i = \begin{bmatrix} 6 \times 6 \\ \text{full} \\ \text{matrix} \end{bmatrix} \begin{bmatrix} 6 \times 6 \\ \text{full} \\ \text{matrix} \end{bmatrix}$$

at a cost of 36 multiplies 30 additions. Using triangular factorization to count the operation for B^{-1}, the cost is 195 multiplies 71 divides 206 additions. Total cost for Λ_j is

$N(48n + 36) + 195$	multiplies
71	divides
$N(48n) + 206$	additions

To calculate the cost for a jacobian inverse a generalized inverse formulation is used. $J_i^{-1} = J_i^T[J_iJ_i^T]^{-1}$ where J_i is the $6 \times n$ jacobian for the ith manipulator and n is the degree of freedom of the manipulator. The cost is $12n + 6$ multiplies 71 divides and $10n + 206$ additions giving a total cost for Λ_j with calculation of the jacobian inverse as

$N(60n + 231) + 195$	multiplies
$71N + 71$	divides
$N(58n + 206) + 206$	additions

Appendix III

6. LTM System Parameters

Motor constants (for all 7 joints)

Torque Constant K_T	8.5	oz-in/A
Back emf constant K_E	6.3	V/KRPM
Armature resistance R	2.5	OHM
Armature inertia J_m	0.0015	oz-in-sec^2
Viscous Damping D	0.3	oz-in-KRPM
Static Friction T_f	0.8	oz-in

Gear ratio from motor shaft to joint (assuming conventional gear driven joints)

Joint 1	522
Joint 2	522
Joint 3	522
Joint 4	522
Joint 5	121
Joint 6	121
Joint 7	25

Denavit and Hartenberg parameters

Denavit-Hartenberg Parameters				
joint	d	a	a	θ
1	0	-90°	0	$\theta_1 + 90°$
2	0	90°	23"	θ_2
3	0	-90°	0	$\theta_3 - 90°$
4	0	90°	20"	θ_4
5	0	-90°	0	θ_5
6	0	90°	0	$\theta_6 + 90°$
7	5.9"	0	0	$\theta_7 - 90°$

Joint Inertia Matrix

Link Number	Inertia Matrix KG-M	Orientation Matrix
1	$\begin{matrix} 0.029 & 0.0 & 0.0 \\ 0.0 & 0.0145 & 0.0 \\ 0.0 & 0.0 & 0.0145 \end{matrix}$	$\begin{matrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -1.0 \\ 0.0 & 1.0 & 0.0 \end{matrix}$
2	$\begin{matrix} 0.029 & 0.0 & 0.0 \\ 0.0 & 0.2989 & 0.0 \\ 0.0 & 0.0 & 0.2989 \end{matrix}$	$\begin{matrix} 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ 1.0 & 0.0 & 0.0 \end{matrix}$
3	$\begin{matrix} 0.029 & 0.0 & 0.0 \\ 0.0 & 0.0145 & 0.0 \\ 0.0 & 0.0 & 0.0145 \end{matrix}$	$\begin{matrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -1.0 \\ 0.0 & 1.0 & 0.0 \end{matrix}$
4	$\begin{matrix} 0.029 & 0.0 & 0.0 \\ 0.0 & 0.2296 & 0.0 \\ 0.0 & 0.0 & 0.2296 \end{matrix}$	$\begin{matrix} 0.0 & -1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ -1.0 & 0.0 & 0.0 \end{matrix}$
5	$\begin{matrix} 0.0163 & 0.0 & 0.0 \\ 0.0 & 0.0082 & 0.0 \\ 0.0 & 0.0 & 0.0082 \end{matrix}$	$\begin{matrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -1.0 \\ 0.0 & 1.0 & 0.0 \end{matrix}$
6	$\begin{matrix} 0.0163 & 0.0 & 0.0 \\ 0.0 & 0.0269 & 0.0 \\ 0.0 & 0.0 & 0.0269 \end{matrix}$	$\begin{matrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ 0.0 & -1.0 & 0.0 \end{matrix}$
7	$\begin{matrix} 0.0182 & 0.0 & 0.0 \\ 0.0 & 0.0099 & 0.0 \\ 0.0 & 0.0 & 0.0099 \end{matrix}$	$\begin{matrix} 0.0 & -1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ -1.0 & 0.0 & 0.0 \end{matrix}$

where the orientation matrix is referenced to the base of LTM.

7. Bibliography

1. Hayati, S.: "Hybrid Position/Force Control of Multiarm Cooperating Robots." IEEE Conference on Robotics and Automation, April 1986.

2. Alberts, T., and Soloway, D.: "Force Control of Multiarm Robot System." IEEE Conference on Robotics and Automation, August 1988.

3. Carnignan, Craig R., and Akin, David L.: "Cooperative Control of two arms in The Transport of an Inertial Load in Zero G." IEEE Conference on Robotics and Automation, August 1988.

4. Mason, M. T.: "Compliance and Force Control for Computer Controlled Manipulators." IEEE Trans. on Systems, Man Cybernetics, v. 11, June 1981, pp. 418-432.

5. Denavit, J., and Hartenberg, R. S.: "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices." ASME Journal of Applied Mechanics, June 1955, pp. 215-221.